University of São Paulo
"Luiz de Queiroz" College of Agriculture

Frame by frame completion probability of an American football pass

Gustavo Pompeu da Silva

Dissertation presented to obtain the degree of Master in Science. Area: Statistics and Agricultural Experimentation

Piracicaba
2022

Gustavo Pompeu da Silva
Bachelor in Statistics

Frame by frame completion probability of an American football pass

Advisor:
Prof. Dr. **RAFAEL DE ANDRADE MORAL**

Dissertation presented to obtain the degree of Master in Science. Area: Statistics and Agricultural Experimentation

Piracicaba
2022

Dados Internacionais de Catalogação na Publicação
DIVISÃO DE BIBLIOTECA - DIBD/ESALQ/USP

Silva, Gustavo Pompeu da
Frame by frame completion probability of an American football pass / Gustavo Pompeu da Silva. – – Piracicaba, 2022 .
36 p.

Dissertação (Mestrado) – – USP / Escola Superior de Agricultura "Luiz de Queiroz".

1. Aprendizado de máquinas 2. National Football League 3. Random forest 4. Software R . I. Título.

ACKNOWLEDGEMENTS

I would like to thank my mom, dad and all my family for all the support during my whole life and specifically during these 2 years of my Master's, it was only possible because of them.

Very special thanks to my supervisor Rafael Moral for all the affection, knowledge and incentive to write this dissertation in english and to participate in CASI 2021, that for sure helped me grow a lot, and for always being available when needed.

Thanks to all my professors at USP/ESALQ that gave their all to learn how to teach classes online in 2020 all of a sudden.

Thanks to my undergraduate supervisor Eduardo Monteiro from UnB for the incentive for me to go to Piracicaba and get a Master's degree.

Thanks to all my friends that helped me in one way or the other during these years, all my longtime friends from NBA VIP, NFL VIP and other groups, who are all the crazy people like me that are in love with sports and fantasy games, and thanks to all my very good friends from school and college, I don't need to cite you by name, you all know who you are. I love you all.

Finally, thanks to CAPES for providing the financial support during my Master's (proc. no. 88887.483407/2020-00), I would not have been able to conclude my Master's without the funding they provided and I thank them deeply.

CONTENTS

RESUMO

Probabilidade de completar um passe no futebol americano quadro por quadro

Futebol americano é um esporte cada vez mais popular, com uma audiência crescente em muitos países do mundo. Nos Estados Unidos existe a *National Football League* (NFL), onde toda jogada ofensiva pode ser uma corrida ou um passe, e nessa dissertação o interesse está nos passes. Nem todas as jogadas de passe são iguais, vários fatores podem influenciar a probabilidade de completar um passe, como separação do recebedor para o defensor mais próximo, distância entre o passador e o recebedor, formação ofensiva, placar do jogo e vários outros. Quando se tenta prever a probabilidade de completar um passe, é essencial saber quem é o alvo do passe. Usando medidas de distância entre os jogadores e a bola, é possível calcular probabilidade empíricas e prever com alta acurácia quem será o alvo. A grande questão é: quão provável é um passe ser completado em uma partida da NFL enquanto a bola está no ar? Foi desenvolvido um algoritmo de aprendizado de máquinas para responder a essa pergunta baseado nos fatores mencionados. Usando dados da temporada de 2018 da NFL, foram obtidas probabilidades condicionais e marginais de completar passes, baseadas em um modelo de floresta aleatoória. Foi feito um procedimento em dois estágios: primeiro, calcularam-se as probabilidades de cada jogador ofensivo ser o alvo do passe, depois, dado que o jogador é o alvo, é prevista a probabilidade do passe ser completado baseado no modelo de floresta aleatória. Por último, a probabilidade geral do passe ser completado pode ser calculada usando a lei da probabilidade total.

Palavras-chave: Aprendizado de máquinas, National Football League, Random forest, Software R

ABSTRACT

Frame by frame completion probability of an American football pass

American football is an increasingly popular sport, with a growing audience in many countries in the world. The most watched American football league in the world is the United States' National Football League (NFL), where every offensive play can be either a run or a pass, and in this dissertation, it is the pass that matters. Not all pass plays in the NFL are created equal, many factors can affect the probability of a pass completion, such as receiver separation from the nearest defender, distance from receiver to passer, offensive formation, game score, among many others. When predicting the completion probability of a pass, it is essential to know who the target of the pass is. By using distance measures between players and the ball, it is possible to calculate empirical probabilities and predict very accurately who the target will be. The big question is: how likely is it for a pass to be completed in an NFL match while the ball is in the air? We develop a machine learning algorithm to answer this based on the aforementioned predictors. Using data from the 2018 NFL season, we obtained conditional and marginal predictions for pass completion probability based on a random forest model. This is based on a two-stage procedure: firstly, we calculate the probability of each offensive player being the pass target, then conditional on the target, we predict completion probability based on the random forest model. Finally, the general completion probability can be calculated using the law of total probability. We present animations for selected plays and show the pass completion probability frame by frame.

Keywords: Machine learning, National Football League, R Software, Random forest

1 INTRODUCTION

American football is a team sport played by two teams of eleven players on a rectangular field with goalposts at each end. The game is divided in plays; in each play one team is the offense and have possession of the ball, and the other is the defense. The offense tries to advance down the field by running or passing the ball, while the defense aims to stop the offense's advancement while trying to take control of the ball themselves. The most common ways of scoring points is by advancing the ball to the opponent's end zone for a touchdown, or by kicking the ball through the opponent's goalposts for a field goal. The team with more points at the end of the game wins.

The National Football League (NFL) is the most popular professional league of American football in the world. It is based in the United States and currently consists of 32 teams, divided in two conferences of four divisions each. The NFL is the most profitable professional sports league in the United States, having generated revenue of 15.26 billion U.S. dollars in 2019 (Gough, 2020). Each season is concluded with the Super Bowl, where the champions of each conference play against each other, and is one of the largest events of the year in the United States, with a growing audience around the world. Super Bowl LV, which was played on February 7th, 2021, had an average viewership of almost 100 million in the United States plus an estimated 30 to 50 million viewers around the world. Although it is still a small viewership when compared to the biggest events in the world like the FIFA World Cup Finals, it is a number that is growing every year (Richter, 2021).

The main difference between this paper and similar works on the topic, such as that of Yurko *et al.* (2020), is that we focused specifically on passing plays in the NFL, and present more detailed results. We used data from the 2018 NFL season to model and predict probabilities of pass completion, as well as to estimate probabilities for each eligible player of the offense to be the target at each play.

We started by cleaning the data and creating the variables we were interested in including in our model (mostly distance measures) to obtain empirical probabilities for each of the offensive players to be the target of the play in every frame, from the moment of the pass until the final outcome. Using statistical models, we obtained the probabilities of pass completion given that a specific player was the target and then, through the law of total probability we were able to estimate the probability of pass completion for the play as a whole (Zwillinger and Kokoska, 1999).

In Chapter 2, we introduce the data set, along with the mathematical definition of the metrics we created. We also present an exploratory data analysis for contextualization, and the statistical modeling tools used. In Chapter 3, we present and discuss our results. Finally, in Chapter 4 we make our final considerations about our work and draw some conclusions and insights about how our results can be useful and how they may be improved even further in future works.

2 MATERIALS AND METHODS

The `R` software (R Core Team, 2021) was used to read the data, build the models, generate the plots and every other computational implementation needed. Many packages were used to obtain the results needed; to manipulate the data we used the `tidyverse` suite of packages (Wickham *et al.*, 2019), to easily implement the cross-validation for the models we used the `caret` package (Kuhn, 2020), to fit the random forests we used the `randomForest` package (Liaw and Wiener, 2002), and to create animations of the plays we used the `gganimate` package (Pedersen and Robinson, 2020).

2.1 NFL data

The data utilized in this paper was obtained from the Kaggle analytics competition NFL Big Data Bowl 2021 (Kaggle, 2020), and is available at `https://www.kaggle.com/c/nfl-big-data-bowl-2021/data`. The competition used NFL's Next Gen Stats data that includes the position and speed of every player on the field during each play. The data contains tracking, play, game, and player information for all possible passing plays during the 2018 regular season, except from three games of week 1, for a total of 253 games. Passing plays are considered to be the ones on which a pass was thrown, the quarterback was sacked, or any one of five different penalties was called (defensive pass interference, offensive pass interference, defensive holding, illegal contact, or roughing the passer).

The data is hierarchical by nature, consisting of game data, play data within each game and tracking data within each play. Besides that we also have player data. The utilized variables from each data level are:

– Game data: game identifier code, and the three-letter abbreviation codes of the home and visitor team;

– Player data: player identification number (unique across players), player name, and player position group (e.g. quarterback (QB), running back (RB), linebacker (LB), wide receiver (WR), defensive back (DB), safety (S), tight end (TE) and defensive lineman (DL), totaling 8 categories);

– Play data: game identifier code, play identifier code, play description, game quarter (categorical, 1 to 5, with 5 representing overtime), down (categorical, 1 to 4), distance needed for a first down (in yards), three-letter abbreviation codes of possession team and which side of the field is the line-of-scrimmage, yard line at line-of-scrimmage, formation used by possession team (e.g. shotgun, wildcat, jumbo, "I" formation, pistol, singleback and empty backfield, totaling 7 categories), number of defenders in close proximity to line-of-scrimmage, number of pass rushers, dropback categorization of quarterback (e.g. designed rollout left, designed rollout right, traditional, scramble, scramble rollout right, scramble rollout left and unknown, totaling 7 categories), home and visiting team scores prior to the play (in points), time on clock of play (in minutes and seconds), NFL categorization of the penalties that occurred on the play, and outcome of the passing play (C: Complete pass, I: Incomplete pass, S: Quarterback sack, IN: Intercepted pass, totaling 4 categories);

– Tracking data: Player position along the long axis of the field (0 - 120 yards), player position along the short axis of the field, (0 - 53.3 yards), tagged play details (moment of ball snap, pass release, pass catch, tackle, etc., totaling 41 categories), player identification number, player name and jersey number, player position group (QB, RB, LB, WR, DB, S, TE and DL), team of corresponding player, frame identifier for each play (starting at 1), game identifier code, play identifier code, and direction to which the offense is moving (left or right).

The variable that indicates the direction to which the offense is moving was used to flip the coordinates x and y when the direction was left, so the plays always align with the direction of the offense's target end zone. A tutorial post on the Kaggle competition forum (Bliss and Lopez, 2020) contained initial code to read and merge the databases and flip coordinates (`https://www.kaggle.com/tombliss/tutorial`).

2.1.1 Data manipulation

First and foremost, all the plays that presented any problem in the database were removed from the data. For example: unusual pass plays like fake punts or fake field goals that did not have a specific offense formation would have a missing value in this variable, or even some plays had clear problems such as not having tracking information of the ball or some players. All plays whose result was a sack were also removed, because although they are considered passing plays by the NFL, a pass does not actually take place. Plays that had a penalty in them were also removed because most of them would have missing values for several variables and we cannot be sure what the penalty is during plays, since the referees only announce the penalties after the play is over.

The play description variable was very important. It describes what happened in the play, for example: "(15:00) M.Ryan pass short right to J.Jones pushed ob at ATL 30 for 10 yards (M.Jenkins).". This contains the name of the player that passed the ball, the player that received the pass and can have other characteristics of the play. From these descriptions, using string manipulation we were able to extract the name of the passer in every play and the name of the target in most of the plays. There were, however, some incomplete passes which did not state the name of the target on the play. In these cases the eligible receiver closest to the ball at the moment of the play being considered an incomplete pass was considered the target. Moreover, plays that were not meant to be attempted passes to a target (e.g. spikes – when the quarterback simply throws the ball to the ground for the clock to stop, and throwaways – when the quarterback is very pressured by the defense and throws the ball away to avoid a sack) could be easily identified because key words such as "spiked" or "threw away" would be present in the text. These plays were also removed from the data.

For the remaining plays we used the variable that tags the play details on every frame (moment of ball snap, pass release, pass catch, tackle, etc.) to determine in which one the forward pass begins and in which one there is an outcome to the play, such as completed pass, incomplete pass or interception. Finally, we filtered only the frames in between these events.

The original database contained 18 309 388 rows, each representing a player within a frame within a play, with rows representing defensive players and the ball on every frame. The total number of frames was 1 247 711. After applying all the filters mentioned above, the database was reduced to 1 012 954 rows, with a maximum of 5 rows for the same frame, representing only the possible receivers of a play, and a total of 203 091 frames.

2.2 Statistical Theory

In this Section we will cover the basic theoretical explanation behind the main statistical topics used throughout this paper.

2.2.1 Logistic Regression

Following Dobson and Barnett (2018), when we want to model a binary variable, which in our case is the completion or incompletion of a pass, we can use a generalized linear model. Let $P = \frac{Y}{n}$, represent the proportion of successes Y obtained within a total number of independent trials n. We assume that $Y \sim \text{Binomial}(n, \pi)$, which yields $E(Y) = n\pi$, where π is the probability of success. We have $E(P) = \pi$, and we can model the probability π as

$$g(\pi) = \boldsymbol{x}^\top \boldsymbol{\beta}, \tag{2.1}$$

where $g(\pi)$ is a monotone and differentiable link function, $\boldsymbol{x}$ is a vector of explanatory variables and $\boldsymbol{\beta}$ is a vector of parameters.

For the logistic regression model, we have that the canonical link function is $g(\pi) = \text{logit}(\pi) = \log\left(\frac{\pi}{1-\pi}\right)$, yielding

$$\log\left(\frac{\pi}{1-\pi}\right) = \boldsymbol{x}^\top \boldsymbol{\beta}, \tag{2.2}$$

which can be rewritten as

$$\pi = \frac{\exp(\boldsymbol{x}^\top \boldsymbol{\beta})}{1 + \exp(\boldsymbol{x}^\top \boldsymbol{\beta})}. \tag{2.3}$$

This is called a logistic regression model.

2.2.1.1 Other link functions

As an alternative to logistic regression, we can use other link functions as $g(\pi)$. In this paper we used two others that are very often used in the literature: the probit and the complementary log-log link. The probit link takes the form

$$g(\pi) = \Phi^{-1}(\pi),$$

where $\Phi(.)$ represents the cumulative distribution function of the standard normal distribution. The complementary log-log link takes the form

$$g(\pi) = \log[-\log(1-\pi)],$$

and is based on the cumulative distribution function of the Gumbel distribution.

These are all binomial generalized linear models, which are easily implemented in `R` by using the `glm()` function.

2.2.2 Random Forests

Random forests (RF) were introduced by Breiman (2001) as an extension of regression or classification trees (Breiman *et al.*, 1984), which are based on decision trees. Each individual regression tree is built from the original sample by bootstrap resampling and is grown based on m randomly selected features. A tree is created by a iterative process that simply partitions the data in several regions by doing a recursive binary splitting. In each step the predictor and its cutpoint are selected such that splitting the space into new regions leads to the greatest possible reduction in the residual sum of squares, and this process is repeated until a stopping criterion is reached. The prediction of the tree will be the average of the response variable on the region the observation is assigned.

A classification RF is an ensemble learning method that builds many classification trees and evaluates their overall performance together. By doing this we avoid the impact of outlier trees because the volume of trees we are building is very large. The best value for m can then be determined via cross-validation.

Suppose we have the training data $\{(\boldsymbol{X}_1, y_1), ..., (\boldsymbol{X}_n, y_n)\}$, where $\boldsymbol{X}_i$ is a vector of the features, and y_i is the response variable. We want to find a function $f : X \to Y$. If M is the total number of features, then a RF:

(i) selects n observations randomly from the original sample, with replacement, which forms a bootstrap sample;

(ii) for each subset, selects m random features from the overall M features.

For a binary response variable, a class prediction is produced for each tree, and the RF prediction probability is the proportion of one of the classes on these predictions. The number of trees in the forest is also a value that can be changed, but as the default value in the package `randomForest` (Liaw and Wiener, 2002) on `R` is 500, and testing higher values didn't produce better results, we decided to maintain this number in our work.

2.2.3 Discriminant Analysis

Discriminant analysis is a multivariate classification technique that aims to separate observations in groups and allocate new ones in one of the predefined groups. Discriminant analysis is exploratory by nature. The goal of this technique is to describe algebraically the differential features of the observations. Discriminant functions are found, which separate the populations as best as possible (Johnson and Wichern, 2007).

We used two types of discriminant analysis: linear and quadratic. They both require that predictors follow a multivariate normal distribution. The Linear Discriminant Analysis (LDA) has an assumption that the predictor data from each class may have different means, but the same covariance structure, while in the Quadratic Discriminant Analysis (QDA) they are allowed have different covariance structures.

LDA is a generalization of the Fisher Linear Discriminant. It can be used for more than two classes, but in this paper we only have the need for two. We obtain the decision boundary between the two classes and every new observation is allocated to one of the classes based on the vector $\boldsymbol{x}$ of explanatory variables associated with them.

The probability of a observation belonging to group k conditional on observing $\boldsymbol{x}$ satisfies:

$$P(k \mid \boldsymbol{x}) \propto \pi_k f(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{2.4}$$

where π_k is the probability of an observation to belong to class k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ represent the vector of means and the covariance matrix of class k respectively. But in LDA we have to assume that both classes have the same covariance structure, which means that when maximizing Equation 2.4, we obtain:

$$\delta_k(\boldsymbol{x}) = \log(\pi_k) + \boldsymbol{x}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k - \frac{1}{2} \boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k. \tag{2.5}$$

For each class k, these are called the linear discriminant functions. For two classes k and l, the decision boundary is where $\delta_k(\boldsymbol{x}) = \delta_l(\boldsymbol{x})$. For a new observation, we calculate these values and see which is greater, and that is the class to which it will be allocated.

For QDA, the difference is that each class is allowed have a different covariance structure, which means we get different functions known as quadratic discriminant functions:

$$\delta_k(\boldsymbol{x}) = \log(\pi_k) - \frac{1}{2} \log|\boldsymbol{\Sigma_k}| - \frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma_k}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_k). \tag{2.6}$$

The conclusions are the same as in LDA. QDA provides more flexible (non linear) boundaries than LDA, but by doing that requires the estimation of more parameters.

In `R`, we used the functions `lda()` and `qda()` from package `MASS` (Venables and Ripley, 2002) to implement LDA and QDA, respectively.

2.2.4 Cross-Validation

Cross-validation, also known as out-of-sample testing, can be any of several similar model validation techniques for assessing how the results of a statistical analysis will generalize to an independent data set (Stone, 1974). These techniques are mainly used in settings where the goal is prediction, to give an insight on how the model will perform when applied to an independent dataset (i.e., an unknown dataset). The goal of cross-validation is to avoid overfitting and selection bias (Cawley and Talbot, 2010). It works by:

- Separating the data in several partitions;
- In each step a model is trained with most of the data;
- The model is tested with the remainder of the data by computing the predictions of these observations that weren't used to train it;
- In the end, all observations will be left out to be tested on exactly one step, thus giving us predictions of all observations without using them in the model that predicted it.

In this thesis, we have one realization of the response variable for each frame in each play. However, since the frames that belong to the same play are dependent, it is best to consider each play as group of frames, and to use the Leave Group Out Cross-Validation (LGOCV) technique, using plays as the grouping factor. This means that all frames within a play were held out in one of the folds. It is necessary to split the data in k-folds, and as a general rule, most authors and empirical evidence suggest that $k = 5$ or $k = 10$ are the most preferred values.

In summary, the goal of cross-validation is predictive power, rather than estimation of a particular structural or causal parameter. Since the method uses out-of-sample comparisons rather than in-sample goodness-of-fit measures, it ensures that we obtain unbiased comparisons of the fit.

2.2.5 Receiver Operating Characteristic curves

A receiver operating characteristic (ROC) curve is a graphical technique that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. To produce a ROC curve, the sensitivities (True Positive Rates) and specificities (True Negative Rates) for different values of a continuous test measure are first tabulated. This results, essentially, in a list of various test values and the corresponding sensitivity and specificity of the test at that value. Then, the graphical ROC curve is produced by plotting true positive rate on the y-axis versus false positive rate on the x-axis for the various values tabulated (Hoo *et al.*, 2017).

A ROC curve similar in shape to the identity line $y = x$ produces false positive results at the same rate as true positive results. Therefore, we expect a diagnostic test with reasonable accuracy to have a ROC curve in the upper left triangle above the $y = x$ identity line. The area under the curve (AUC) is a global measure of the ability of a test to discriminate whether a specific condition is present or not present (Zou *et al.*, 2007). An AUC of 0.5 represents a test with no discriminating ability (i.e., no better than chance), while an AUC of 1 represents a test with perfect discrimination.

In this paper we chose to evaluate the models through the AUC, computed with the trapezoidal rule (Robin *et al.*, 2011), as it is one of the most widely used metrics for evaluation of binary classification problems. The AUC is a robust overall measure to evaluate the performance of score classifiers because its calculation relies on the complete ROC curve and thus involves all possible classification thresholds (or cut-off points) (Hanley and McNeil, 1982).

In `R` we used package `pROC` (Robin *et al.*, 2011) to calculate the values and package `plotROC` (Sachs, 2017) to generate the plots.

2.3 Distance measures

It is noteworthy to mention that any field in the NFL is divided in yards, which is the standard distance measure for everything in the NFL, so all the distances we calculated in this paper are also in yards.

The first distance measure we wanted to calculate was the distance of a point (x_0, y_0) to the line created by the points (x_1, y_1) and (x_2, y_2), representing the movement of the ball or a player from one frame $t-1$ to another frame t. We refer to this distance as d.

Let $\boldsymbol{v}$ be a vector perpendicular to the line formed by (x_1, y_1) and (x_2, y_2), and given by

$$\boldsymbol{v} = \begin{bmatrix} y_2 - y_1 \\ -(x_2 - x_1) \end{bmatrix}. \tag{2.7}$$

Then let $\boldsymbol{r}$ be a vector from the point (x_0, y_0) to (x_1, y_1)

$$\boldsymbol{r} = \begin{bmatrix} x_1 - x_0 \\ y_1 - y_0 \end{bmatrix}. \tag{2.8}$$

We can calculate the distance d, which is given by projecting $\boldsymbol{r}$ onto $\boldsymbol{v}$ (WEISSTEIN, 2021)

$$d = |\boldsymbol{v} \cdot \boldsymbol{r}| = \frac{\left|(x_2 - x_1)(y_1 - y_0) - (x_1 - x_0)(y_2 - y_1)\right|}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}. \tag{2.9}$$

In Figure 2.1 we visualize the points and the vectors used to calculate the distance d. In `R` we created a function to calculate and return the distance d when informing the coordinates of the three points, using the formula given on Equation 2.9.

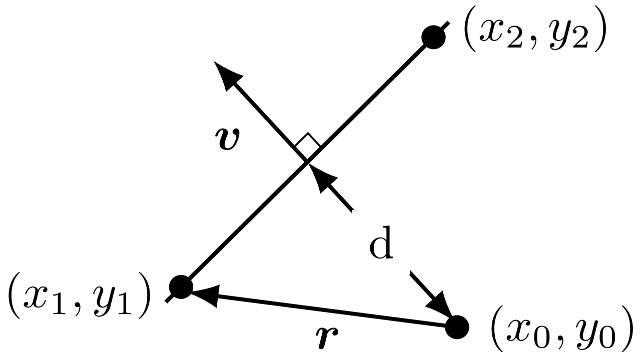


Figure 2.1. Visual representation of point-line distance d

To calculate the distance between two points on a plane, we used the Euclidean distance between two points, which for any set of points (x_0, y_0) and (x_1, y_1) is given by

$$h = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}. \tag{2.10}$$

In R, h was calculated through the function `PointDistance` from package `raster` (HIJMANS, 2021), which uses the formula on Equation 2.10. Finally, the last distance-based measure we used was the distance difference for a player or for the ball between frames t and $t-1$, which we refer to as b:

$$b = h_t - h_{t-1}. \tag{2.11}$$

2.4 Statistical modeling

We mainly separated our work into two big modeling issues, one to determine probabilities of the offensive players to be the target of the play, and the other to determine the probabilities of the pass to be completed given that a specific player is the target.

2.4.1 Target prediction

To calculate distances between players and the ball, we used the variables that describe their coordinates in the field in each frame, and applied them to the equations described in Section 2.3. There were some cases where the ball coordinates in the previous frame were identical to the current frame; in these situations we considered the coordinates of the previous frame that had different coordinates to calculate the distance from point (player) to line (ball direction) described in Equation 2.9. We refer to this distance as $d^{(1)}$. In the case of the location of a player being exactly on top of the line created by the ball's coordinates we set the distance to 0.01 to avoid issues of having a distance being 0.

In some plays the ball was still going backwards on the first frames despite the frames being after the event of a "pass forward", most likely because of an error in the database or because the passer was moving backwards when releasing the ball. In these cases we discarded the first few frames until the ball really started moving forward.

From Equation 2.11, we define $d^{(3)}$ as the distance b, which represents the distance difference between players and ball from previous to current frame, resulting in a negative number if the distance from a player to the ball decreased in current frame, and a positive number if the distance increased. A characteristic of this distance is that it can have negative values, so we created a standardised version $d^{(2)}$ such that $d^{(2)} \geq 1$, defined as

$$d_{ij}^{(2)} = \begin{cases} d_{ij}^{(3)} + 1 + \left|\min(\boldsymbol{d}_j^{(3)})\right| & \text{if } \min(\boldsymbol{d}_j^{(3)}) > 0 \\ d_{ij}^{(3)} + 1 - \min(\boldsymbol{d}_j^{(3)}) & \text{otherwise} \end{cases}, \tag{2.12}$$

where $d_{ij}^{(2)}$ and $d_{ij}^{(3)}$ represents the distance $d^{(2)}$ and $d^{(3)}$ for player i on frame j respectively, and $\boldsymbol{d}_j^{(3)}$ is the vector of all observations of $d^{(3)}$ on frame j. The use of $d^{(2)}$ avoids problems arising from values equal to or close to zero.

We calculated the probability of a player i being the target for every frame j, considering each frame independent, even those in the same play, because what was used to determine these probabilities were only the aforementioned distances, and since from one frame to another on the same play these distances don't change much, the probabilities for the same player on the same play end up following a natural dependent pattern. Using the distance variables $d^{(1)}$ and $d^{(2)}$, we calculated an empirical probability of the form

$$P_k(T = i \mid j) = \frac{\min(\boldsymbol{d}_j^{(k)})}{d_{ij}^{(k)}} \times \frac{1}{\sum_i^n \frac{\min(\boldsymbol{d}_j^{(k)})}{d_{ij}^{(k)}}} = \frac{1}{d_{ij}^{(k)} \times \sum_{i=1}^n (d_{ij}^{(k)})^{-1}}, \tag{2.13}$$

where $k = 1, 2$, $i = 1, \ldots, n$ represents the players who are the potential targets in the play of frame j, $d_{ij}^{(k)}$ is the distance measure $d^{(k)}$ of player i in frame j, and $\boldsymbol{d_j^{(k)}}$ represents the vector of values of $d^{(k)}$ for all potential targets on frame j. This formula guarantees that the sum of the probabilities will be 1 for every frame, and for example, considering $k = 1$, guarantees that the closest player to the line projected by the ball will have the highest probability of being the target, while when $k = 2$, the player that got closer to the ball from previous to current frame will have the highest probability of being the target.

To increase the accuracy of our method, we created a weighted combination of probabilities based on $d^{(1)}$ and $d^{(2)}$, written as

$$f(W) = WP_1(T = i \mid j) + (1 - W)P_2(T = i \mid j), \tag{2.14}$$

where the weight $W \in [0, 1]$. The objective is to give more importance to one measure or the other in each frame, depending on some characteristics of the plays. If we consider that these two metrics have the same importance in every situation we will have $W = 0.5$.

We used four different approaches to determine these weights, considering the order statistics of a metric for one player per frame in the whole database. Let $d^{(4)}$ be the Euclidean distance h between the players and the ball. The four weights used were:

- $W^{(1)}$, based on the $d^{(3)}$ of the player with the lowest $d^{(1)}$;
- $W^{(2)}$, based on the $d^{(2)}$ of the player with the lowest $d^{(1)}$;
- $W^{(3)}$, based on the $d^{(4)}$ of the player with the lowest $d^{(1)}$;
- $W^{(4)}$, based on the $d^{(4)}$ of the player with the lowest $d^{(2)}$.

The values of $W^{(r)}$, $r = \{1, 2, 3, 4\}$, are the same for every player i on the same frame, but different for every frame j. Therefore we use the indexing $W_j^{(r)}$ to refer to weight of type r calculated for frame j. To represent the values of these weights mathematically, we define the matrix

$$\mathbb{D}^{(k)} = \begin{bmatrix} d_{11}^{(k)} & d_{12}^{(k)} & d_{13}^{(k)} & \dots & d_{1m}^{(k)} \\ d_{21}^{(k)} & d_{22}^{(k)} & d_{23}^{(k)} & \dots & d_{2m}^{(k)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{n1}^{(k)} & d_{n2}^{(k)} & d_{n3}^{(k)} & \dots & d_{nm}^{(k)} \end{bmatrix}, \tag{2.15}$$

with $d_{ij}^{(k)}$ representing the distance measure $d^{(k)}$ ($k = \{2, 3, 4\}$) of player $i = 1, ..., n$ in frame $j = 1, ..., m$, where $m = 203091$ represents the total number of frames in all of our database. Let $\mathbb{Z}^{(s)}$ ($s = \{1, 2\}$) be a matrix whose elements are

$$z_{ij}^{(s)} = \begin{cases} 1 & \text{if } \min_i(\boldsymbol{d}_j^{(s)}) = d_{ij}^{(s)} \\ 0 & \text{otherwise} \end{cases}, \tag{2.16}$$

where $\boldsymbol{d}_j^{(s)}$ is the j-th column of $\mathbb{D}^{(s)}$. We also define

$$\boldsymbol{d}_*^{(k,s)} = \operatorname{diag}\{(\mathbb{D}^{(k)})^\top \mathbb{Z}^{(s)}\}, \tag{2.17}$$

where $\operatorname{diag}(\mathbf{X})$ represents the main diagonal of the square matrix $\mathbf{X}$. The indices k and s are both used to represent distances $d^{(1)}$ to $d^{(4)}$ that are used for matrices $\mathbb{D}^{(k)}$ and $\mathbb{Z}^{(s)}$, but they are different because weights $W^{(r)}$ depend on two different distances, as explained on the description of the weights listed above.

Now let $\boldsymbol{U}$ be the ordered vector of dimension u containing the unique values in $\boldsymbol{d}_*^{(k,s)}$, and $\boldsymbol{a}$ be a vector of the same dimension u such that the l-th element of $\boldsymbol{a}$ is $a_l = \frac{l-1}{u-1}$, then

$$W_j^{(r)} = a_l, \text{ for the } l \text{ corresponding to the match } U_l = d_{*j}^{(k,s)}. \tag{2.18}$$

To make Equation 2.18 more clear, suppose we have 7 frames and $\boldsymbol{d}_*^{(k)} = \{3, 3, 2, 6, 6, 1, 8\}$, then $\boldsymbol{U} = \{1, 2, 3, 6, 8\}$ and $\boldsymbol{a} = \{\frac{0}{4}, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, \frac{4}{4}\}$, then $\boldsymbol{W}^{(r)} = \{\frac{2}{4}, \frac{2}{4}, \frac{1}{4}, \frac{3}{4}, \frac{3}{4}, \frac{0}{4}, \frac{4}{4}\}$, because we are seeking the l-th element of $\boldsymbol{a}$ for the l that corresponds to the match $U_l = d_{*j}^{(k,s)}$.

For each different weight, the values of k and s in these equations will depend on which distance measures the weights are based on. For $W^{(1)}$ we have $k = 3$ and $s = 1$ because it is based on the $d^{(3)}$ of the player with the lowest $d^{(1)}$. Then for $W^{(2)}$ we have $k = 2$ and $s = 1$, for $W^{(3)}$ we have $k = 4$ and $s = 1$, and finally for $W^{(4)}$ we have $k = 4$ and $s = 2$.

Substituting $W_j^{(r)}$ in Equation 2.14 we get the probabilities of every player being the target on frame j based on weights of type r. In practice, what this means for weight $W^{(1)}$ is that the lower the $d^{(3)}$ of the player with the lowest $d^{(1)}$ is in a frame, we give more importance to the probability $P_1(T = i \mid j)$, from Equation 2.13, which means that if the player closest to the line projection of the ball is also getting much closer to the ball itself, we will give much more importance to the probability calculated from the distance of player to line projection of the ball. Similarly, the higher this distance is, we give more importance to the probability $P_2(T = i \mid j)$, which means that if the player closest to the line projection of the ball is actually getting farther away from the ball itself, we will give much more importance to the probability calculated from the distance difference between the current and previous frames. The same line of thought applies for the other weights, but for $W^{(4)}$ it is inverted, meaning that in Equation 2.14, for this type of weight, we have to use $f(1 - W^{(4)})$, that is because it is the only one that generates a distance of the player with the lowest $d^{(2)}$ and not the lowest $d^{(1)}$.

Now that we have a probability of every player being the target in every frame of every play, if we consider the player with highest probability in each frame to be the predicted target, we can calculate the accuracy of our method. We do so using each one of the proposed weights. A problem with these probabilities is that even in situations where a player clearly does not have a chance to be the target anymore, the probability associated to this player will not be zero. To fix some of the most obvious cases of this problem, we made an adjustment. We considered that a player is very close to ball when he has $d^{(1)}$ and $d^{(4)}$ lower than 2 yards; when this happens, the probabilities of all the other players that are not within the same distances are added to his probability, and theirs are set to zero. In the extreme rare cases that there is more than one player very close to the ball, the probabilities are "transferred" to the one that has the highest probability of being the target. Table 2.1 shows the accuracy before and after applying the aforementioned adjustment.

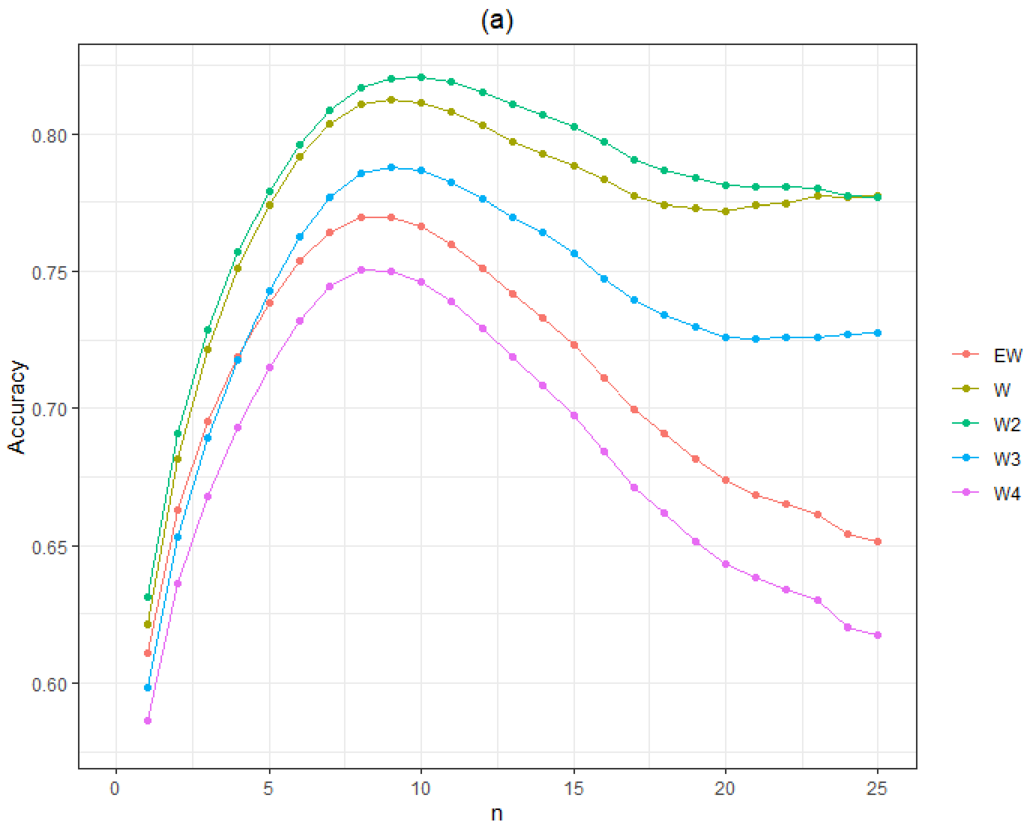
Table 2.1. Accuracy of target prediction using different weights, before and after the adjustment of transferring probabilities when there is at least one player very close to the ball. *EW* represents equal weights ($W = 0.5$ in Equation 2.14)

Weight	Accuracy before adjustment	Accuracy after adjustment
EW	81.25%	82.67%
$W^{(1)}$	84.15%	85.48%
$W^{(2)}$	85.66%	86.68%
$W^{(3)}$	84.34%	85.23%
$W^{(4)}$	81.06%	81.89%

Of all plays in the data, the number of frames analysed varies from 1 to 46, but is mostly concentrated below 20. 75% of the plays consist of 16 or fewer frames, and 95% of the plays have 25 or fewer. With this information, we decided to analyse the accuracy considering the frames, both from the beginning of plays and from the end of plays, to see specifically if the weights tested would have different performances in these situations.

In Figure 2.2, we can see that looking from the beginning of the plays, weight $W^{(2)}$ seems to provide the best accuracy for all frames. But when we look at the last frames of the plays, we see that weight $W^{(3)}$ has a better accuracy in the last 12 frames.

This characteristic propelled us to create a new weight combining $W^{(2)}$ and $W^{(3)}$, giving more importance to $W^{(2)}$ in the beginning of plays but changing to give more importance to $W^{(3)}$ as the play



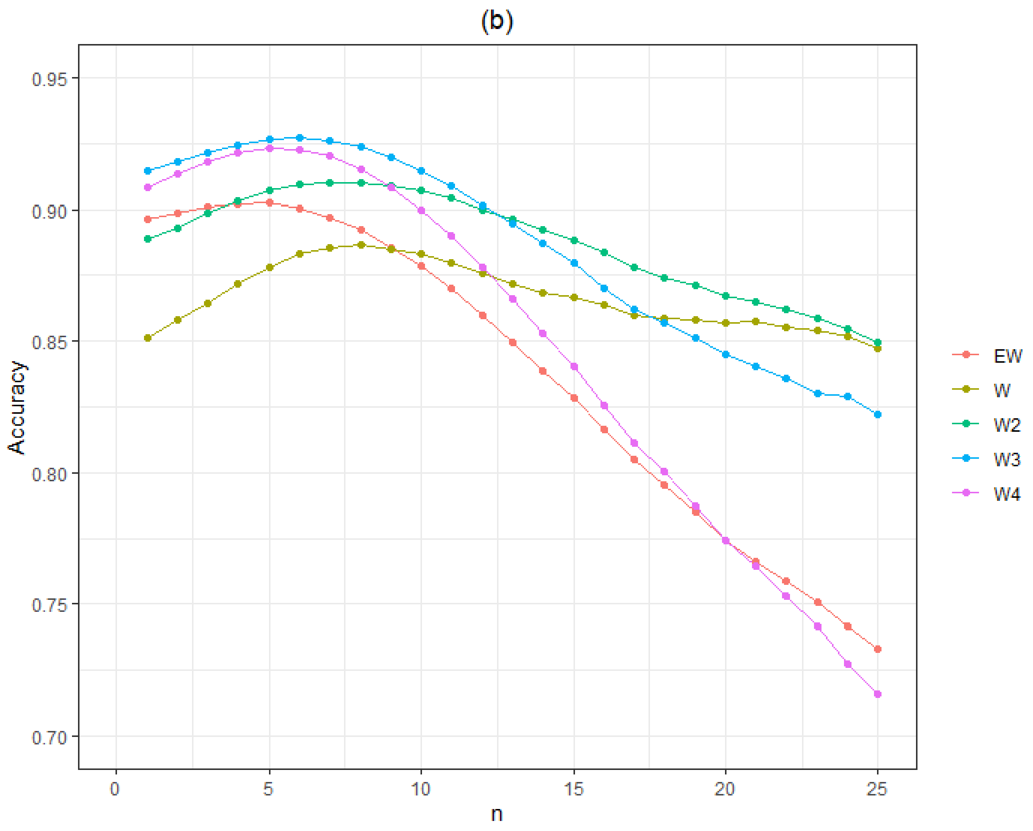


Figure 2.2. Accuracy of Target prediction in (a) first or (b) last n frames of plays, i.e., proportion of frames that the player with highest probability of being the target really was the target. *EW* represents equal weights ($W = 0.5$ in Equation 2.14)

extends. We used a logistic model to create these new weights. The values to evaluate the model were 1 to 46, which are the minimum and maximum of frames in a play, with asymptote 1, inflection point of the curve being the mean of number of frames in the plays, which was 13.34183, and for the scale parameter we performed a grid search with accuracy as our target function, and 2.57 was the value that maximized the accuracy, so that was the value for the parameter. We called these new weights $W^{(2,3)}$, defined as

$$W_t^{(2,3)} = \frac{1}{1 + e^{\frac{(13.34183-t)}{2.57}}}, \tag{2.19}$$

with $t = 1, ..., 46$ representing the number of the frame in a play.

Therefore, the final formula we used to calculate the probability of a player i to be the target of a play in a frame j is

$$P(T = i \mid j, t, \mathbb{D}^{(1)}, \mathbb{D}^{(2)}, \mathbb{D}^{(4)}) = W_t^{(2,3)} f(W_j^{(3)}) + (1 - W_t^{(2,3)}) f(W_j^{(2)}). \tag{2.20}$$

This new and final way to compute the probabilities achieved an accuracy of 86.92%, which was better than the ones obtained through either of $W^{(2)}$ and $W^{(3)}$, as expected. The difference between t and j is that t is specific for each play, varying from 1 to 46 (maximum of frames evaluated in a play), and j is a generic count of frames on the whole database, from 1 to 203148 (total number of frames on the database).

2.4.2 Completion probability

Now that we have probabilities of each eligible offensive player being the target, we can calculate the conditional probability of the pass to be completed given that a player is the target. Using the law of total probability we can then compute the probability of a pass being completed for each frame of every play. We write

$$P(C) = \sum_{i=1}^{n} P(C \mid T = i) P(T = i), \tag{2.21}$$

where $P(C)$ is the completion probability, n is the number of players that can be the target on a given play, $P(T = i)$ is the probability of player i being the target, calculated through Equation 2.20, and $P(C \mid T = i)$ is the completion probability given that player i is the target. The computation of $P(C \mid T = i)$ is the focus of this section.

The response variable for the models we tested was a binary variable with levels “I” and “C”, representing the result of the play to be an incomplete or a complete pass, respectively. Pass interceptions were considered incomplete passes. We used 32 explanatory variables, listed below:

- Play data:
  - Game quarter (factor, with levels 1 to 5, the last one representing overtime);
  - Down (factor, with levels 1 to 4);
  - Distance needed for a first down (numeric);
  - Formation used by possession team (factor, with 7 categories);
  - Number of defenders in close proximity to line-of-scrimmage (numeric);
  - Number of pass rushers (numeric);
  - Dropback categorization of quarterback (factor, with 7 categories);
  - Time on clock of play, in seconds (numeric);
  - Yard line at line-of-scrimmage from 1-99 (numeric);

  - Offensive team score prior to the play (numeric);
  - Defensive team score prior to the play (numeric);
  - Indicator if the offensive team is playing at home (logical);
  - Distance from passer to given target at the moment of pass (numeric);
- Player data:
  - Player position of the given target (WR, RB, TE, QB and defensive player (DEF)) (factor, with 5 categories);
  - Player position of the closest defensive player (LB, DB, S, DL and offensive player (OFF)) (factor, with 5 categories);
  - Player position of the second closest defensive player (LB, DB, S, DL and OFF) (factor, with 5 categories).
- Frame data:
  - Distance from given target to line projection created by the ball (numeric);
  - Distance from given target to the ball (numeric);
  - Distance difference in current and previous frame from given target to the ball (numeric);
  - Distance from closest defensive player to line projection created by the ball (numeric);
  - Distance from closest defensive player to the ball (numeric);
  - Distance difference in current and previous frame from closest defensive player to the ball (numeric);
  - Distance from second closest defensive player to line projection created by the ball (numeric);
  - Distance from second closest defensive player to the ball (numeric);
  - Distance difference in current and previous frame from second closest defensive player to the ball (numeric);
  - Distance from closest defensive player to line projection created by the given target (numeric);
  - Distance from closest defensive player to the given target (numeric);
  - Distance difference in current and previous frame from closest defensive player to the given target (numeric);
  - Distance from second closest defensive player to line projection created by the given target (numeric);
  - Distance from second closest defensive player to the given target (numeric);
  - Distance difference in current and previous frame from second closest defensive player to the given target (numeric);
  - Distance from given target to the nearest sideline (numeric).

The closest defensive player mentioned was considered to be the closest defensive player to the line projection of the given target, and the second closest defensive player is the closest defensive player to the given target in Euclidean distance. When the same player is the closest by both metrics, we considered the second closest defensive player to be the second closest in Euclidean distance.

In Figure 2.3, we demonstrate the correlation between some of these explanatory variables and the completion or incompletion of a pass, where the y axis in all plots represent the completion percentage,

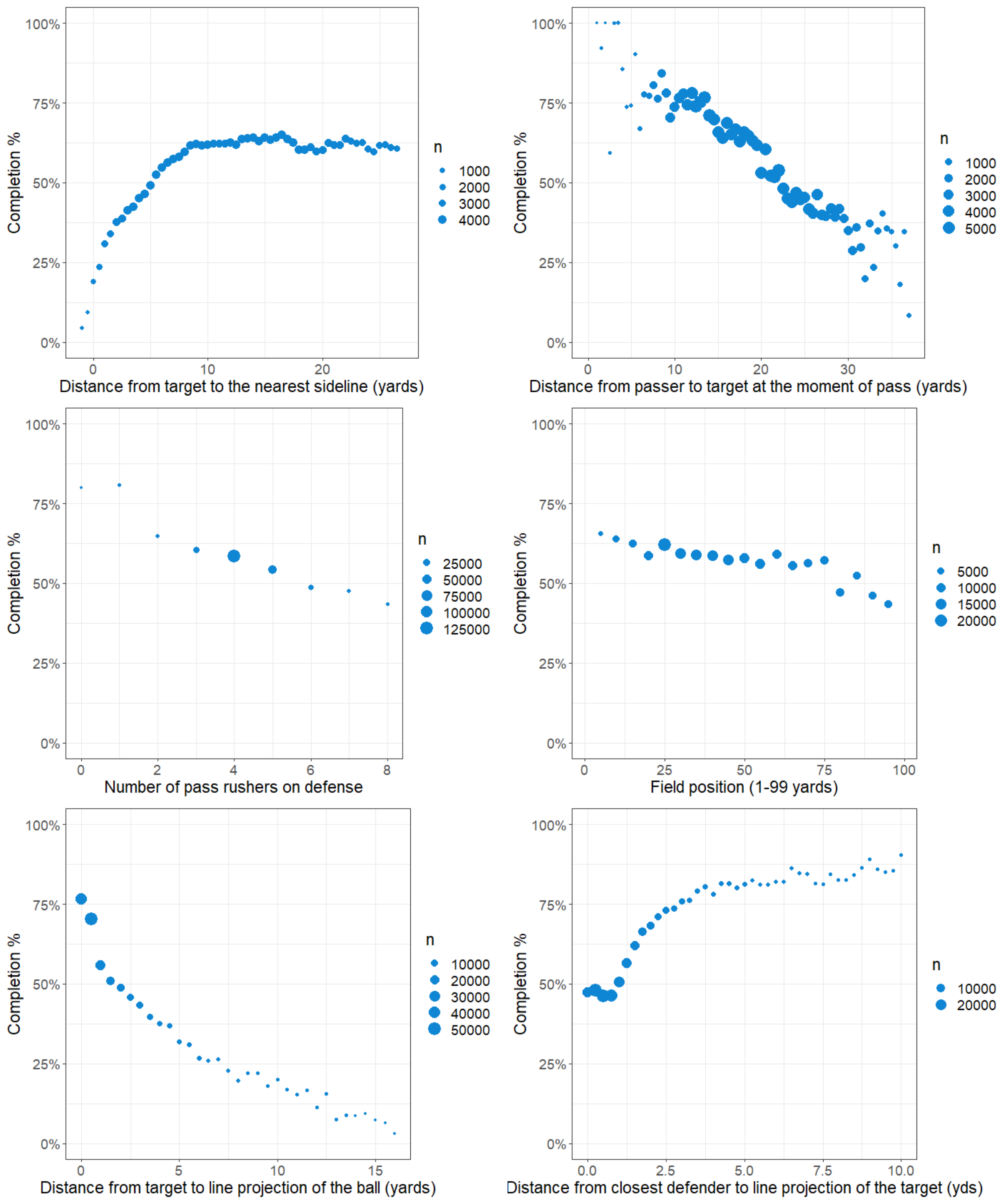


Figure 2.3. Plots of six different variables versus the completion percentage of passes, showing the correlation between explanatory and response variables. The continuous variables displayed on the x axis were discretized in intervals, and the size of the points correspond to the number of frames belonging to each interval

i.e., the percentage of frames corresponding to the metric in the x axis where the result of the play was a completed pass.

We tested several different models: Random Forests, Logistic Regression, Binomial Regression with Probit link, Binomial Regression with complementary-log-log link, Linear Discriminant Analysis and Quadratic Discriminant Analysis. The objetive was to know which one would have the best overall performance. To test these models we used Leave Group Out cross-validation (LGOCV) with 5 or 10 folds, using plays as the grouping factor, meaning that all frames within a play were held out in one of the folds. To test the models, only the data corresponding to the real target of the plays were used, meaning we have one observation per frame.

Table 2.2. Comparison between all the different models tested, sorted by AUC (area under the ROC curve) in descending order, as well as computational time taken to run the cross-validation procedures. "cloglog"is the complementary log-log link

Method	**Folds**	**AUC**	**Time**
Random Forest	10	0.8829	∼3.88 hours
Random Forest	5	0.8825	∼1.78 hours
GLM (logit link)	10	0.7874	2.00 mins
GLM (logit link)	5	0.7877	56.16 secs
GLM (probit link)	10	0.7861	5.04 mins
GLM (probit link)	5	0.7864	2.08 mins
LDA	10	0.7840	1.09 mins
LDA	5	0.7843	32.35 secs
GLM (cloglog link)	10	0.7814	6.30 mins
GLM (cloglog link)	5	0.7816	2.94 mins
QDA	10	0.7487	38.58 secs
QDA	5	0.7462	21.02 secs

For each tree within the Random Forest algorithms we allowed subsets containing between 5 to 20 variables to be chosen at every split, and we obtained the best results based on the AUC when this number was 15. This corresponds to the `mtry` argument within the function `randomForest()` from package `randomForest` (Liaw and Wiener, 2002) in R. The values shown in Table 2.2 for the method Random Forest are referent to the value of `mtry` = 15. From the Table we can see that the performance of this metric does not change much for the same method when comparing between 5 or 10-fold cross-validation, as expected, because the K-fold is just a validation strategy. However, there is a large difference between some of the methods, and the Random Forest models were vastly superior to the other methods, but were associated to larger computational burden (although not that large, since four hours is not a long time considering such a big dataset). We therefore chose the Random Forest model to draw our results. The ROC curve of the Random Forest model based on the 10-fold leave-group-out cross-validation can be seen in Figure 2.4.

To summarise all our work, Figure 2.5 shows the step-by-step algorithm of what we would have to do if we obtain a new play to calculate probabilities.

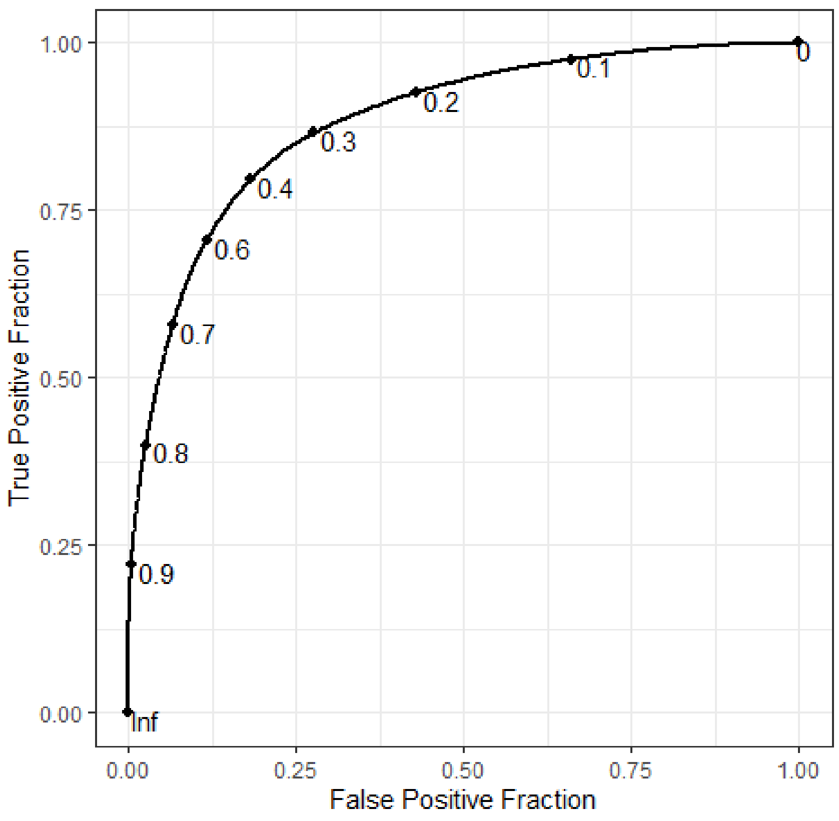


Figure 2.4. ROC curve for the Random Forest model, based on 10-fold leave-group-out cross-validation. The numbers inside the plot represent thresholds (cut-off points) for the probabilities.

All `R` code utilized in this paper is made available at `https://github.com/gustavopompeu/NFLPassCompletion`.

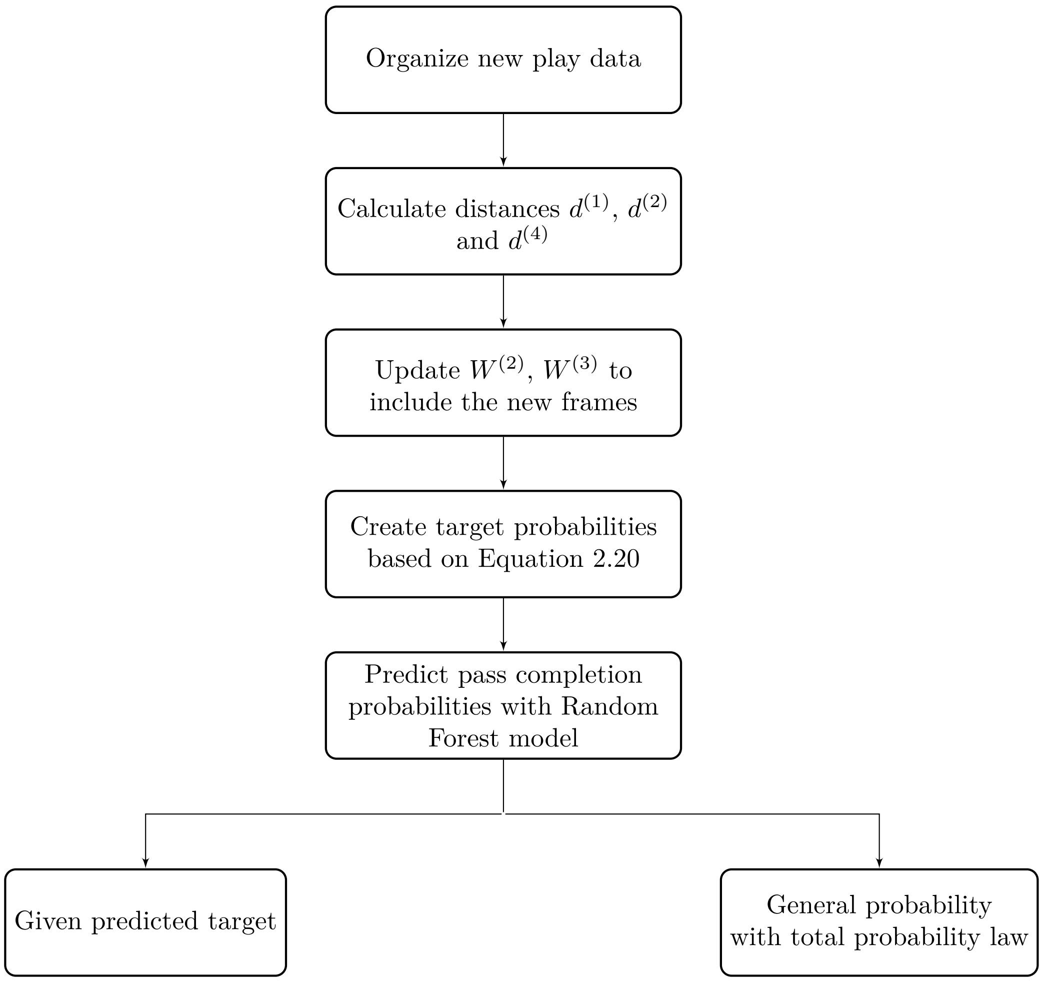

Organize new play data
Calculate distances $d^{(1)}$, $d^{(2)}$ and $d^{(4)}$
Update $W^{(2)}$, $W^{(3)}$ to include the new frames
Create target probabilities based on Equation 2.20
Predict pass completion probabilities with Random Forest model
Given predicted target
General probability with total probability law

Figure 2.5. Step-by-step algorithm of how to calculate pass completion probabilities for new observations

3 RESULTS

The results presented in this section are derived from the test subsets created by the cross-validation. We have 4 or 5 offensive players that can be the target in every play, which means 4 or 5 observations per frame. We were interested in the probabilities $P(C \mid T = i)$ described in Equation 2.21 for us to be able to calculate $P(C)$ as well.

For example, in the first play of the game between the Philadelphia Eagles and the Atlanta Falcons on week 1, which was a completed pass from Matt Ryan to Julio Jones for a 10 yard gain, we present the probabilities for the first frame after the pass started in Table 3.1. With these probabilities we have that for this frame $P(C) = 0.579$.

Table 3.1. Probabilities of players being the target $\boldsymbol{P(T = i)}$ and the completion probability given that the player is the target $\boldsymbol{P(C \mid T = i)}$, for the first frame after the pass started of the first play of the game between the Atlanta Falcons and the Philadelphia Eagles on week 1

Player	$\boldsymbol{P(T = i)}$	$\boldsymbol{P(C \mid T = i)}$
Julio Jones	0.477	0.786
Mohamed Sanu	0.101	0.480
Devonta Freeman	0.314	0.436
Austin Hooper	0.049	0.190
Ricky Ortiz	0.059	0.158

Given that this play was indeed a completed pass, it's expected that for the last few frames the only player with a probability to be the target is the one that really was the target. This results in $P(C) = P(C \mid T = i)$ for i being the player that was the real target, which in this play was Julio Jones. He had $P(C \mid T = i) = 0.524$ in the last frame before the event of pass completed. This is demonstrated in Table 3.2. So our model calculated that the probability of Julio Jones catching the ball on the last frame before the catch as being 52.4%.

Table 3.2. Probabilities of players to be the target and the completion probability given that the player is the target, for the frame when it was considered a completed pass in the first play of the game between the Atlanta Falcons and the Philadelphia Eagles on week 1

Player	$\boldsymbol{P(T = i)}$	$\boldsymbol{P(C \mid T = i)}$
Julio Jones	1.000	0.524
Mohamed Sanu	0.000	0.252
Devonta Freeman	0.000	0.472
Austin Hooper	0.000	0.462
Ricky Ortiz	0.000	0.400

Having the probabilities for all the frames in between, we can make animations to demonstrate the evolution of the probabilities during any play. In Figure 3.1 we see four frames of this Julio Jones (#11) reception, beginning with the frame described in Table 3.1, and ending with the one in Table 3.2. The information we can see on each frame are the completion probability $P(C)$, the shirt number of the player who is the predicted target (player with highest $P(T = i)$ in the frame), the completion probability given predicted target, which is the $P(C \mid T = i)$ for player i with the highest $P(T = i)$ in the frame, and the number of the frame. To see the animated GIF of this whole play, click here (if you are reading this dissertation on PDF).

An example of an animation of a play that was a very probable pass completion can be seen in Figure 3.2. It happened on week 2 in the game between the Cleveland Browns and the New Orleans Saints, and it was a 23 yard pass from Tyrod Taylor (#5) to Rashard Higgins (#81). To see the animated GIF of this whole play, click here (if you are reading this dissertation on PDF).

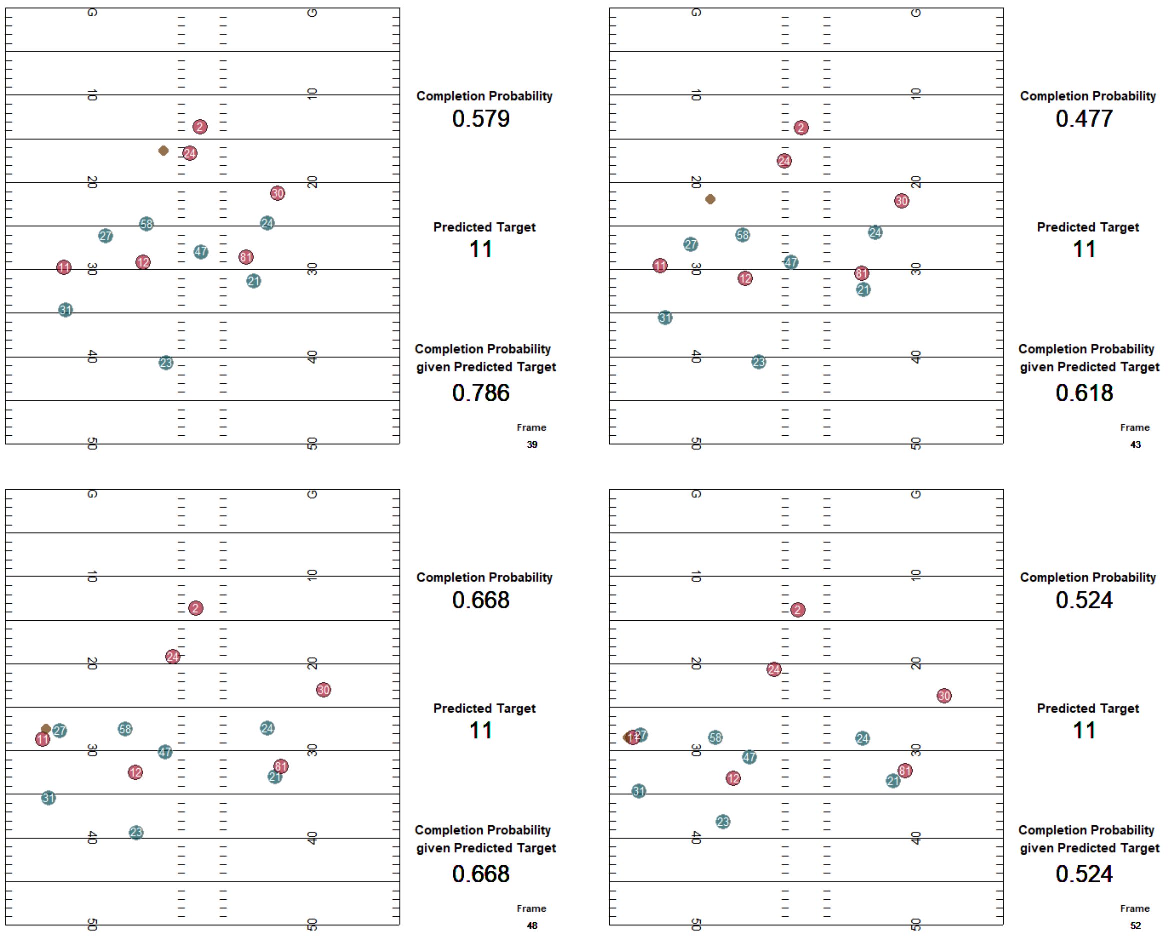


Figure 3.1. Frames 39, 43, 48 and 52 of a Julio Jones (#11) reception in the first play of the game between the Atlanta Falcons and the Philadelphia Eagles on week 1 with information on the probabilities. The ball is represented in brown, the offense in red and the defense in blue.

Another example, this time of an incomplete pass that indeed had a very low completion probability can be seen in Figure 3.3, where in a week 16 game between the Pittsburgh Steelers and the New Orleans Saints, Ben Roethlisberger (#7) tried a deep pass to JuJu Smith-Schuster (#19) but was not successful. To see the animated GIF of this whole play, click here (if you are reading this dissertation on PDF).

In Figure 3.4 we can see effectively how the completion probability we calculated is very relevant by plotting it versus the completion percentage. When we use the completion probability per frame, we have multiple observations for the same play, so the completion percentage represents the proportion of frames corresponding to a play that resulted in a completed pass. When we use the average completion probability, the completion percentage represents the proportion of plays that resulted in a completed pass.

In Table 3.3 we show the Pearson's correlation coefficient and the Lin's concordance correlation coefficient for all the four situation shown in Figure 3.4. Lin's concordance correlation coefficient (CCC) (Lin, 1989) measures the agreement between two variables to evaluate reproducibility. It combines measures of both precision and accuracy to determine how far the observed data deviate from the line of perfect concordance (that is, the line at 45 degrees on a square scatter plot). Lin's coefficient increases in value as a function of the nearness of the data's reduced major axis to the line of perfect concordance (the accuracy of the data) and of the tightness of the data about its reduced major axis (the precision of

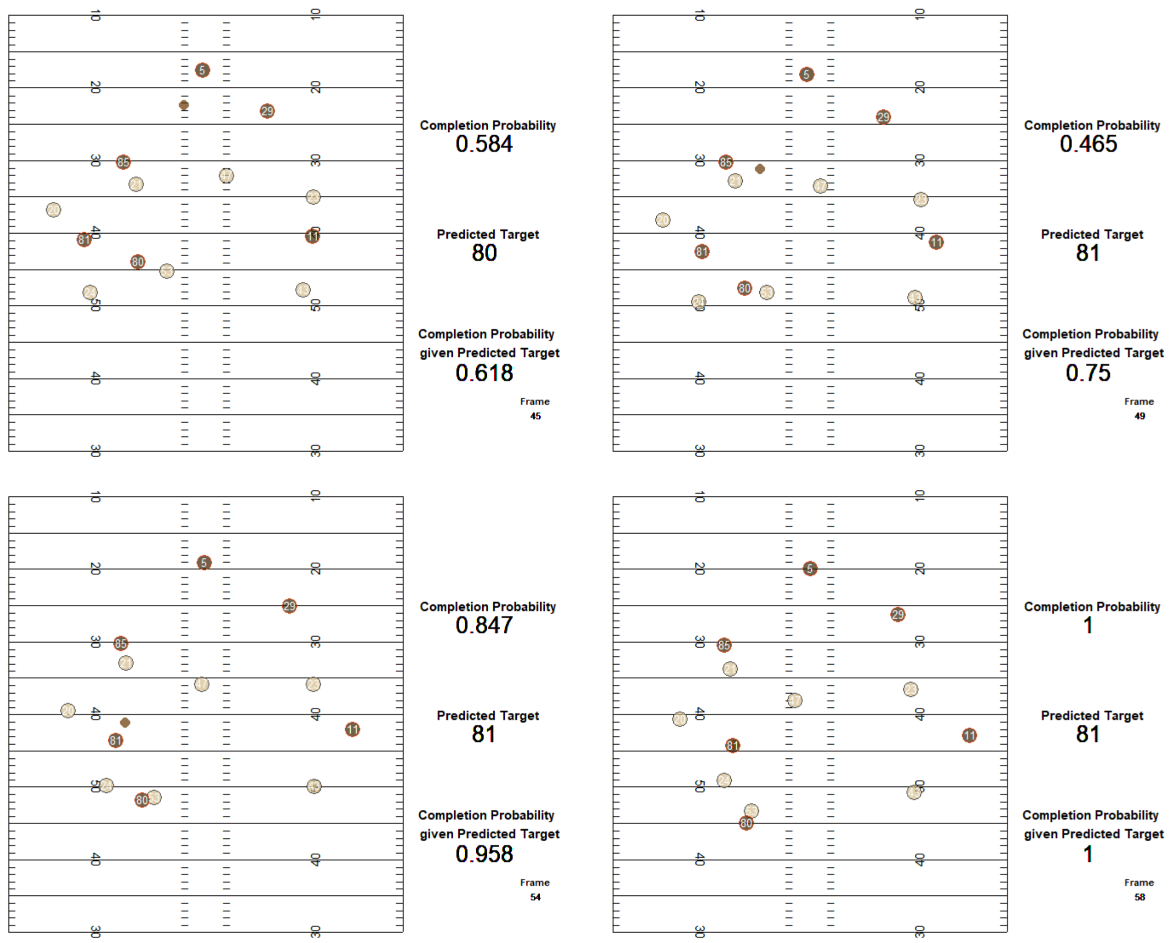


Figure 3.2. Frames 45, 49, 54 and 58 of a Rashard Higgins (#81) reception in the game between the Cleveland Browns and the New Orleans Saints on week 2. The ball is represented in brown, the offense also in brown and the defense in gold.

the data). In `R`, we used function `CCC()` of package `DescTools` to calculate it.

Table 3.3. Pearson's correlation coefficient and Lin's concordance correlation coefficient for all the different setups in Figure 3.4

Probabilities	**Correlation**	**Concordance**
$P(C)$ per frame	0.978	0.958
$P(C \mid T = i)$ per frame	0.998	0.998
Average $P(C)$ per play	0.958	0.903
Average $P(C \mid T = i)$ per play	0.980	0.942

From this Table, we can see that the results are far better when comparing $P(C \mid T = i)$ to the completion percentage than when comparing $P(C)$, this shows that our probabilities to determine the player most likely to be the target are working very well. Another conclusion that we can draw is that analysing the probabilities frame by frame is better than computing an average probability of all frames on a play. This shows that our model is obtaining very accurate results even in the beginning of plays, most likely because the variables of distance to projection of the line made from the ball or the players are a very good indicator if the pass is going in the right direction and if the offensive player is well guarded by the defense or not.

For the training results, if we consider a 0.5 threshold for predictions if a pass will be completed

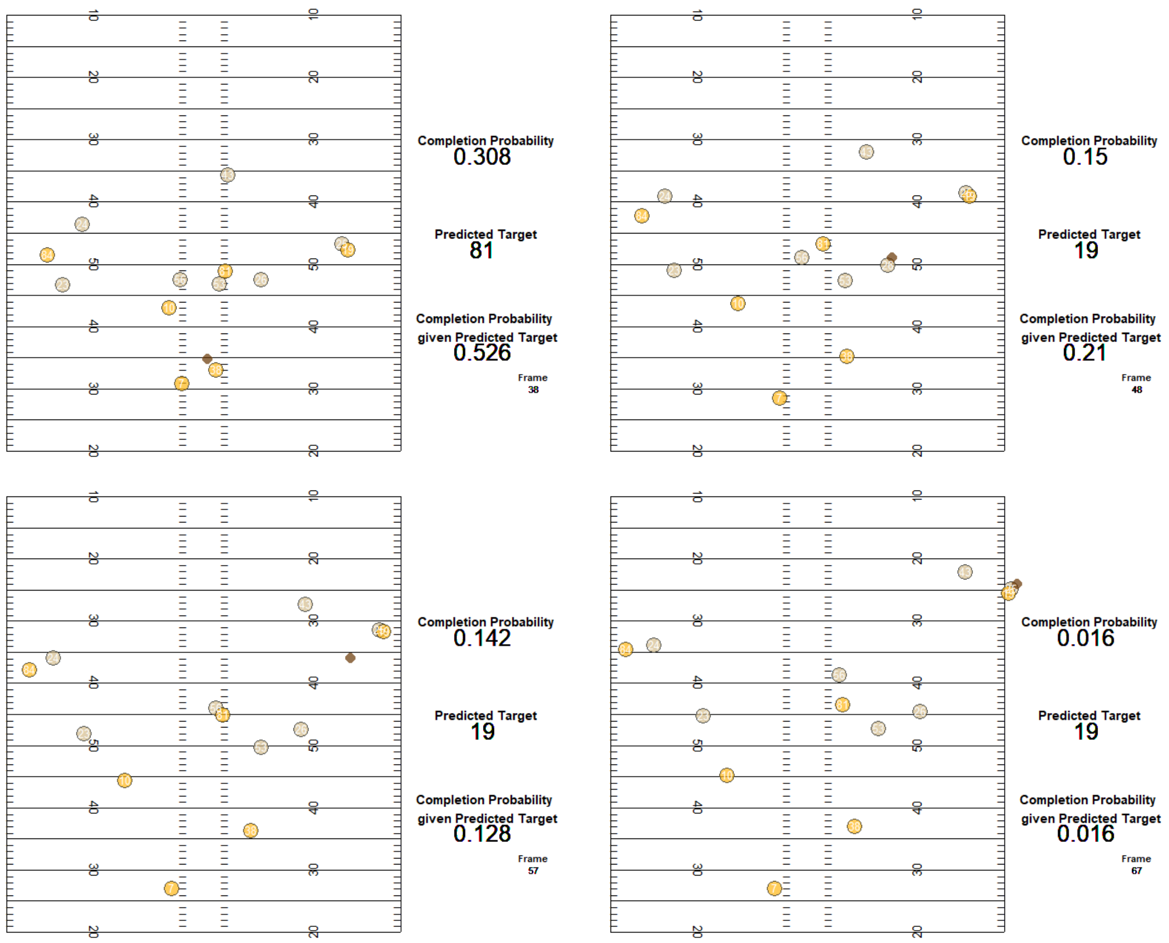


Figure 3.3. Frames 38, 48, 57 and 67 of an incomplete pass from Ben Roethlisberger (#7) to JuJu Smith-Schuster (#19) in the game between the Pittsburgh Steelers and the New Orleans Saints on week 16. The ball is represented in brown, the offense in yellow and the defense in gold.

or not for the predicted target on every frame, we get a 95.8% accuracy in predicting the result of the plays.

3.1 Next Gen Stats

NFL player tracking, also known as Next Gen Stats, is the capture of real time location data, speed and acceleration for every player, every play on every inch of the field. Sensors throughout the stadium track tags placed on players' shoulder pads, charting individual movements within inches (NFL Next Gen Stats, 2021). The player tracking data used in this work was obtained by the NFL Next Gen Stats, and this work in general was inspired by it. During the broadcast of NFL games, many different statistics obtained by the Next Gen Stats team are shown on screen for the audience.

Documented statistics of completion probability calculated by the NFL Next Gen Stats are very hard to find. The only data we found from the 2018 season are presented in an article on the NFL website (The Next Gen Stats Analytics Team, 2018), where they talk about the three most improbable catches of the first week (`https://www.nfl.com/news/next-gen-stats-introduction-to-completion-probability-0ap3000000964655`). These are the only plays we can compare the results from our work with, but unfortunately, one of these three plays is from one of three games that are missing from the database (Denver Broncos vs. Seattle Seahawks), so we actually have results from our model only

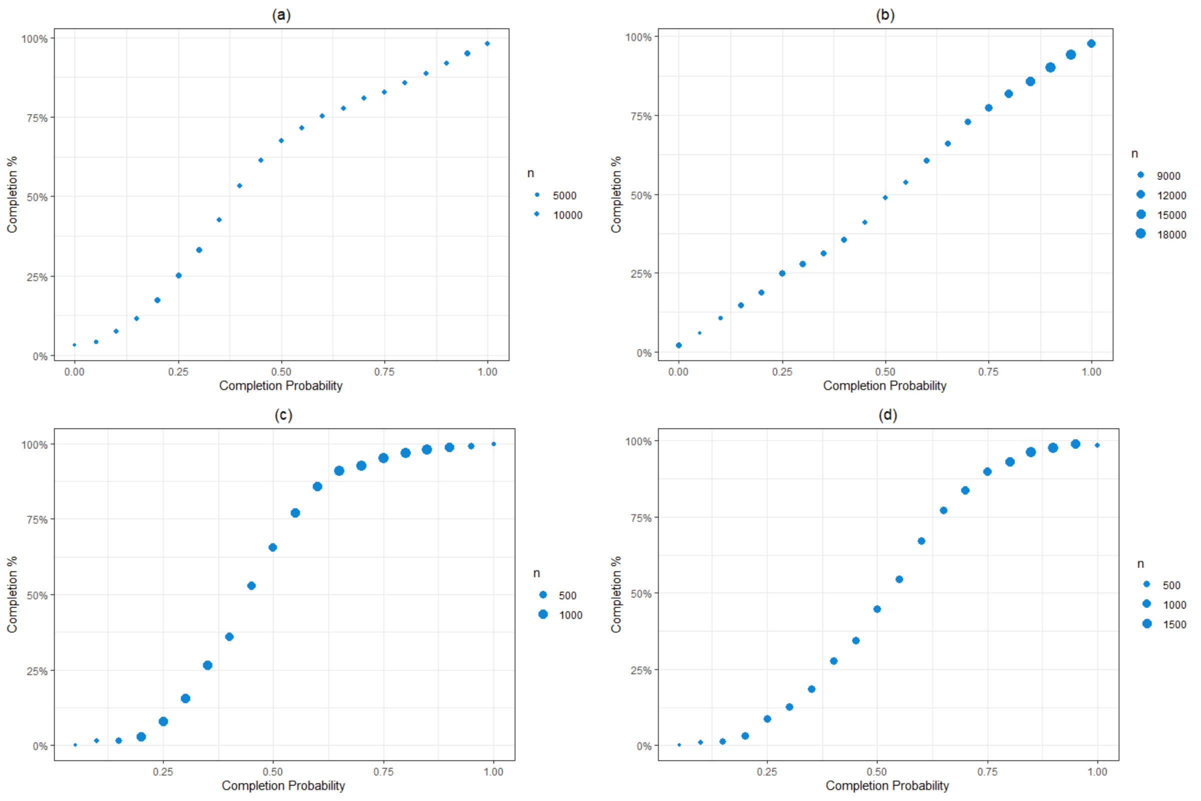


Figure 3.4. (a) General completion probability ($P(C)$) per frame, (b) Completion probability given predicted target ($P(C \mid T = i)$, i representing the player with the highest $P(T = i)$ in the frame) per frame, (c) Average general completion probability per play and (d) Average completion probability given predicted target per play versus completion percentage

for two of these plays.

Figure 3.5 shows four frames of the first play described in the NFL article, a 39-yard touchdown pass from Aaron Rodgers (#12) to Geronimo Allison (#81) in the game between the Green Bay Packers and the Chicago Bears.

In the data we used, the frame in which the play was deemed a completed pass was frame 80, the last one shown in Figure 3.5. From our approach, we obtained a 62.4% completion probability. But if we consider a few frames earlier, such as frame number 75, displayed in Figure 3.6, we see that the ball is already in range of Geronimo Allison and the completion probability is just 24.8%. To see the animated GIF of this entire play, click here (if you are reading this dissertation on PDF).

The other play we can compare with the article is a touchdown from Tom Brady (#12) to Rob Gronkowski (#87) for the New England Patriots against the Houston Texans. We show four frames of this play in Figure 3.7. We can see that the final completion probability obtained from our framework was 28.8%. To see the animated GIF of this whole play, click here (if you are reading this dissertation on PDF).

The Next Gen Stats article does not mention at which point of the play the probabilities shown were calculated; it could be at the moment of the catch or the lowest probability reached during the play, or even an average of the whole play. Also, it could be a general probability (similar to our $P(C)$) or specific to the players cited (similar to our $P(C \mid T = i)$). To cover all these possibilities we made a plot showing the evolution of our calculated completion probabilities for all the frames during the pass (see Figure 3.8), for both the Geronimo Allison and the Rob Gronkowski touchdowns.

We can conclude from Figure 3.8 that our framework produced higher probabilities than the

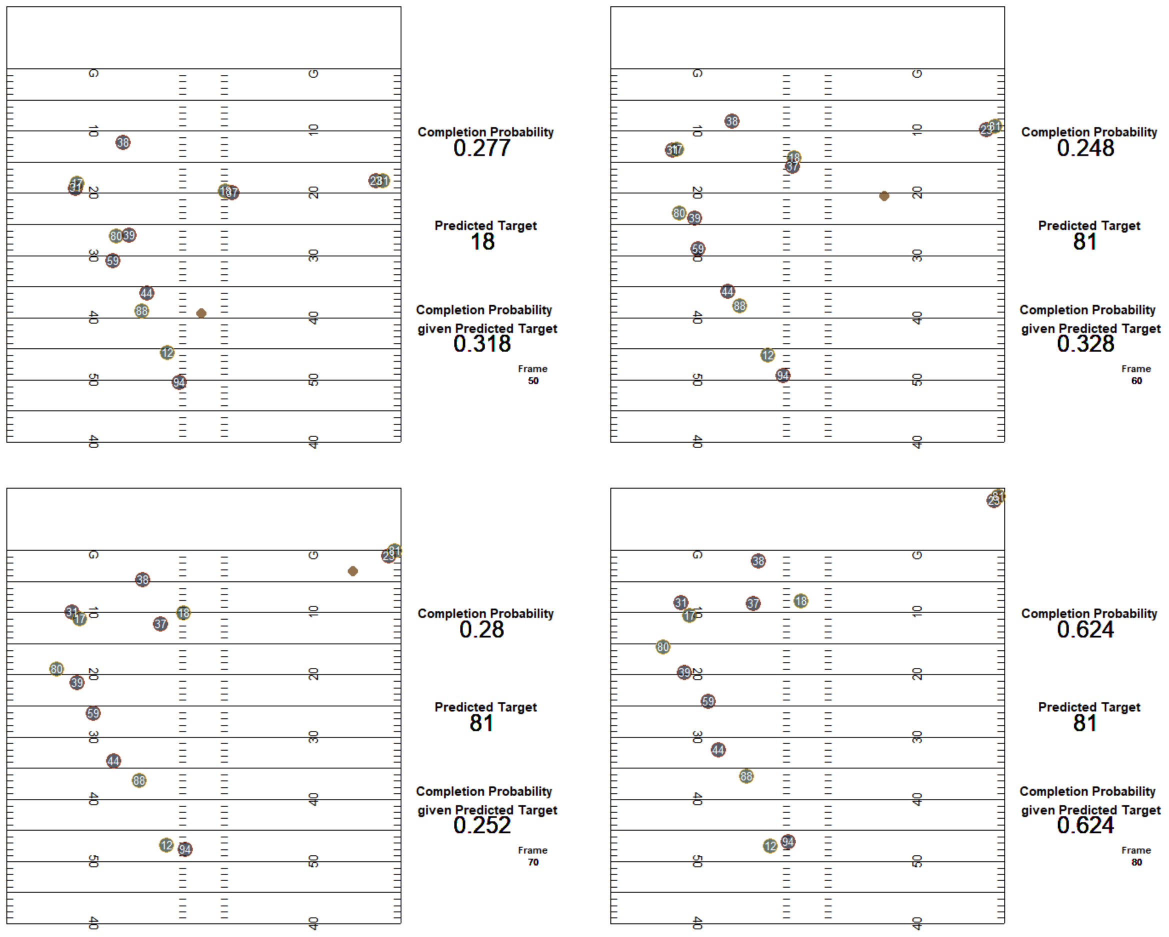


Figure 3.5. Frames 50, 60, 70 and 80 of the Geronimo Allison (#81) touchdown for the Green Bay Packers versus the Chicago Bears on week 1. The ball is represented in brown, the offense in green and the defense in navy.

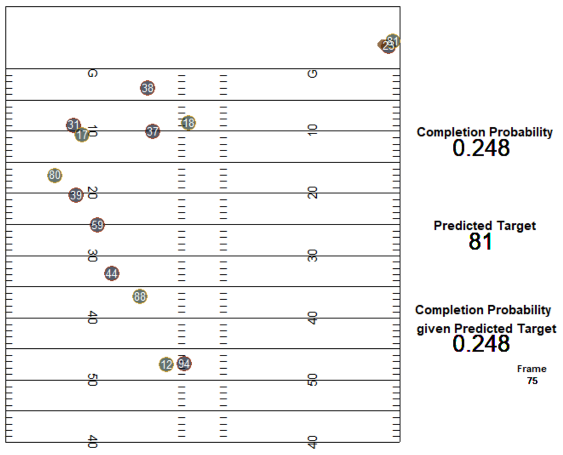


Figure 3.6. Frame 75 of the Geronimo Allison (#81) touchdown for the Green Bay Packers versus the Chicago Bears on week 1. The ball is represented in brown, the offense in green and the defense in navy.

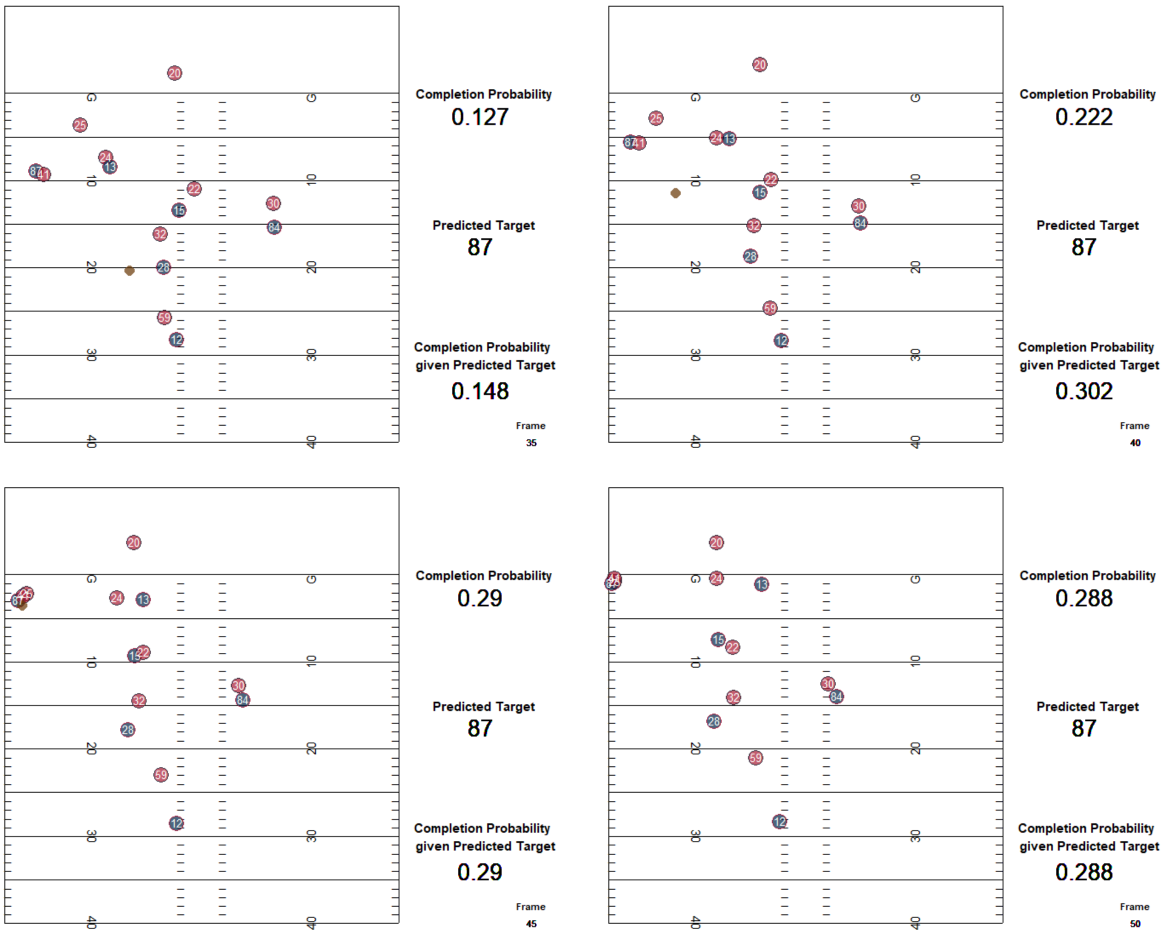


Figure 3.7. Frames 35, 40, 45 and 50 of the Rob Gronkowski (#87) touchdown for the New England Patriots versus the Houston Texans on week 1. The ball is represented in brown, the offense in blue and the defense in red.

NFL Next Gen Stats in almost every possible scenario, and since both plays were completed passes, our framework provided a very competitive performance when compared to Next Gen Stats, with a more detailed output.

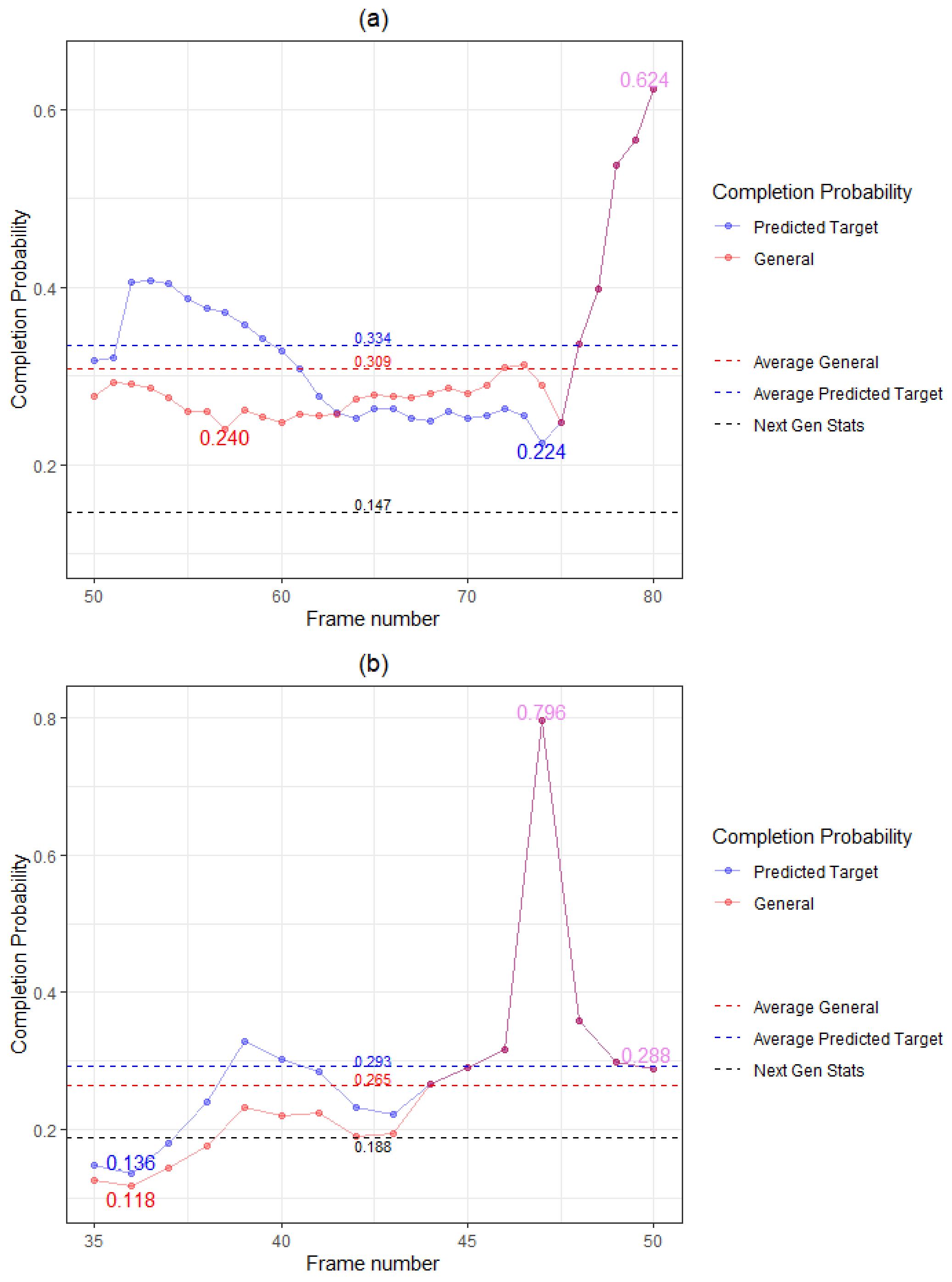


Figure 3.8. Evolution of completion probabilities $P(C)$ and $P(C \mid T = i)$ for all the frames during the pass in the (a) Geronimo Allison and (b) Rob Gronkowski touchdowns from week 1 games

4 DISCUSSION

We calculated completion probabilities of most passing plays in the NFL 2018 season. A very relevant point is that to obtain a completion probability, we only need information on the current frame and previous ones, meaning that we do not need to know how or when the play will end. This allows us to compute the probabilities for a play in real time, given that we have enough computational power and can obtain the necessary information (such as the coordinates of the players in the field).

If the vertical coordinates of the ball were also available, this could certainly be used to improve our framework even further. This is because sometimes the ball could be very close to a player when looking at the available x and y coordinates, but in reality the ball is very high up in the air and going in the direction of a player further up in the field.

Our empirical probabilities of players being the target of the play proved to be very effective, even in the initial frames of the passes. The distance from players to the line projection of the ball was essential to obtain these good results.

The Random Forest model proved to be vastly superior to the other ones tested to predict the completion probabilities, even though it took a lot more time to compute. The results obtained were extremely good when comparing to the real completion percentage of the passes, as shown in Figure 3.4. We could not find the same data for other seasons, which unfortunately made it impossible to expand our work through more games and seasons.

Further work would include an improvement on how to determine during plays if a player still has a chance to be the target or not, and possibly utilize information not available on the data to create variables to differentiate specific players based on their historical performance on the NFL and college football.

REFERENCES

Andri et mult. al., S., 2021 *DescTools: Tools for Descriptive Statistics*. R package version 0.99.42.

Angerer, P., T. Kluyver, and J. Schulz, 2021 *repr: Serializable Representations*. R package version 1.1.3.

Bliss, T. and M. Lopez, 2020 2021 Big Data Bowl Demo / Tutorial. `https://www.kaggle.com/tombliss/tutorial`, [Online; accessed 25-May-2021].

Breiman, L., 2001 Random forests. Machine learning **45**: 5–32.

Breiman, L., J. Friedman, R. Olshen, and C. Stone, 1984 Classification and regression trees. wadsworth int. Group **37**: 237–251.

Cawley, G. C. and N. L. Talbot, 2010 On over-fitting in model selection and subsequent selection bias in performance evaluation. The Journal of Machine Learning Research **11**: 2079–2107.

Dobson, A. J. and A. G. Barnett, 2018 *An introduction to generalized linear models*. CRC press.

Gough, C., 2020 Total revenue of the National Football League 2001-2019. `https://www.statista.com/statistics/193457/total-league-revenue-of-the-nfl-since-2005/`, [Online; accessed 21-May-2021].

Grolemund, G. and H. Wickham, 2011 Dates and times made easy with lubridate. Journal of Statistical Software **40**: 1–25.

Hanley, J. A. and B. J. McNeil, 1982 The meaning and use of the area under a receiver operating characteristic (roc) curve. Radiology **143**: 29–36, PMID: 7063747.

Hijmans, R. J., 2021 *raster: Geographic Data Analysis and Modeling*. R package version 3.4-10.

Hoo, Z. H., J. Candlish, and D. Teare, 2017 What is an roc curve? Emergency Medicine Journal **34**: 357–359.

Johnson, R. A. and D. W. Wichern, 2007 *Applied Multivariate Statistical Analysis*. Applied Multivariate Statistical Analysis, Pearson Prentice Hall.

Kaggle, 2020 NFL Big Data Bowl. `https://www.kaggle.com/c/nfl-big-data-bowl-2021/overview`, [Online; accessed 24-May-2021].

Kuhn, M., 2020 *caret: Classification and Regression Training*. R package version 6.0-86.

Liaw, A. and M. Wiener, 2002 Classification and regression by randomforest. R News **2**: 18–22.

Lin, L. I.-K., 1989 A concordance correlation coefficient to evaluate reproducibility. Biometrics pp. 255–268.

NFL Next Gen Stats, 2021 Ngs: Nfl next gen stats.

Pedersen, T. L. and D. Robinson, 2020 *gganimate: A Grammar of Animated Graphics*. R package version 1.0.7.

Pinheiro, J., D. Bates, S. DebRoy, D. Sarkar, and R Core Team, 2021 *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-152.

R Core Team, 2021 *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria.

Richter, F., 2021 Super Bowl Pales in Comparison to the Biggest Game in Soccer. `https://www.statista.com/chart/16875/super-bowl-viewership-vs-world-cup-final/`, [Online; accessed 21-May-2021].

Robin, X., N. Turck, A. Hainard, N. Tiberti, F. Lisacek, J.-C. Sanchez, and M. Müller, 2011 proc: an open-source package for r and s+ to analyze and compare roc curves. BMC Bioinformatics **12**: 77.

Sachs, M. C., 2017 plotROC: A tool for plotting roc curves. Journal of Statistical Software, Code Snippets **79**: 1–19.

Slowikowski, K., 2021 *ggrepel: Automatically Position Non-Overlapping Text Labels with 'ggplot2'.* R package version 0.9.1.

Stone, M., 1974 Cross-validatory choice and assessment of statistical predictions. Journal of the Royal Statistical Society: Series B (Methodological) **36**: 111–133.

The Next Gen Stats Analytics Team, 2018 Next gen stats: Introduction to completion probability. `https://www.nfl.com/news/next-gen-stats-introduction-to-completion-probability-0ap3000000964655`.

Venables, W. N. and B. D. Ripley, 2002 *Modern Applied Statistics with S.* Springer, New York, fourth edition, ISBN 0-387-95457-0.

Weisstein, E., 2021 Point-Line Distance–2-Dimensional. From MathWorld–A Wolfram Web Resource. `https://mathworld.wolfram.com/Point-LineDistance2-Dimensional.html`, [Online; accessed 25-May-2021].

Wickham, H., 2007 Reshaping data with the reshape package. Journal of Statistical Software **21**: 1–20.

Wickham, H., 2011 The split-apply-combine strategy for data analysis. Journal of Statistical Software **40**: 1–29.

Wickham, H., M. Averick, J. Bryan, W. Chang, L. D. McGowan, R. François, G. Grolemund, A. Hayes, L. Henry, J. Hester, M. Kuhn, T. L. Pedersen, E. Miller, S. M. Bache, K. Müller, J. Ooms, D. Robinson, D. P. Seidel, V. Spinu, K. Takahashi, D. Vaughan, C. Wilke, K. Woo, and H. Yutani, 2019 Welcome to the tidyverse. Journal of Open Source Software **4**: 1686.

Wickham, H. and D. Seidel, 2020 *scales: Scale Functions for Visualization.* R package version 1.1.1.

Wilke, C. O., 2020 *cowplot: Streamlined Plot Theme and Plot Annotations for 'ggplot2'.* R package version 1.1.1.

Yurko, R., F. Matano, L. F. Richardson, N. Granered, T. Pospisil, K. Pelechrinis, and S. L. Ventura, 2020 Going deep: models for continuous-time within-play valuation of game outcomes in american football with tracking data. Journal of Quantitative Analysis in Sports **16**: 163–182.

Zou, K. H., A. J. O'Malley, and L. Mauri, 2007 Receiver-operating characteristic analysis for evaluating diagnostic tests and predictive models. Circulation **115**: 654–657.

Zwillinger, D. and S. Kokoska, 1999 *CRC Standard Probability and Statistics Tables and Formulae.* CRC Press.