Be the QB: Optimizing Offensive Play Calls Based on Predicted Defensive Assignments

Adam Koplik and Ian Fratarcangeli - Undergraduate Track (Hamilton College)

I. Introduction

The ability to read a defense is a non-negotiable skill for an NFL quarterback. After breaking the huddle, QBs immediately scan the opposition to identify exploitable weaknesses. Our project aimed to simulate this process by predicting defensive assignments and then “calling” the optimal play.

Initially, we sought to predict general defensive coverages (e.g., Cover-2, Cover-3). However, we quickly realized that analyzing the problem at an individual player level would be more practical. Our project’s ultimate goal was to “be the QB,” enabling a model to call an audible for any given play.

II. Defensive Coverage Assignments: Data Cleaning

The field holding Pro Football Focus’s (PFF) “coverageAssignment” returned an NA value for any player not considered in coverage. We decided to label these players as “blitzers” and narrowed the training data to focus exclusively on pass plays, as rushing plays would typically classify all 11 players as blitzers. Additionally, we filtered out spikes, as well as prevent and goal-line sets.

We further simplified the coverage assignments into four categories: blitzers (from NA), man-to-man, short zone (hook-curl, curl-flat, curl, and hole coverages), and deep zone (deep half, deep third, deep quarter, and deep free coverages). These assignment bins increased the validation accuracy of the random forest model from 26.63% (20 categories) to 58.42% (eight categories) and further to 83.53% (four categories). However, a reverse random forest model, which used the predicted assignments to predict the general coverage, found that the four-bin model achieved a 77.34% accuracy compared to 77.58% for eight bins and 78.41% for the original 20. This trade-off between predictive accuracy and simplicity highlights the value of streamlining defensive assignments for our model.

For general data cleaning, we standardized field direction by shifting the x, y, dir, and orientation of all plays with a **playDirection** of "left." We also calculated the distance of each individual player from the ball at the snap.

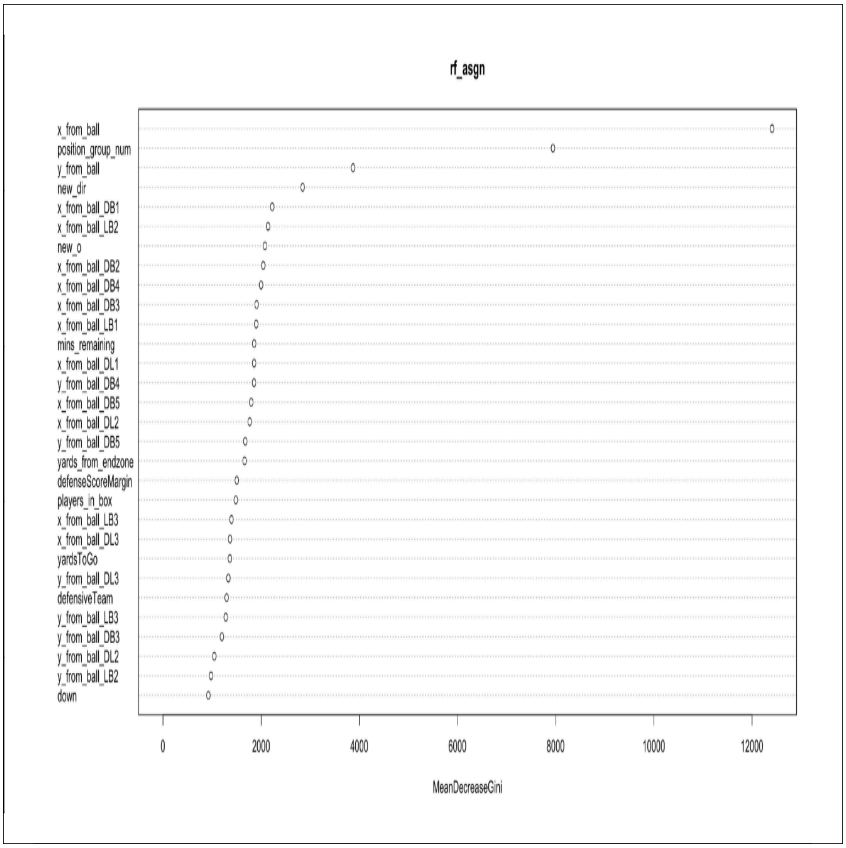
III. Defensive Coverage Assignments: Model

The defensive coverage assignment model used the **randomForest** R library for prediction. The data combined the *plays*, *playerPlays*, *games*, and tracking datasets. Location data was restricted to exactly one second before the snap. Defensive positions were grouped into three categories: defensive backs (DB), defensive linemen (DL), and linebackers (LB). The field **position_player_num** was determined by sorting each position group from “top-to-bottom” of the field. For example, “DB1” represented the defensive back closest to the top of the field, “DB2” the next closest, and so on. The data was split into training and validation sets (80% and 20%, respectively). The following variables were used in the model:

- **week**
- **quarter**
- **down**
- **yardsToGo**
- **defensiveTeam**
- **x_from_ball** → horizontal distance from the ball for the individual player one second before the snap
- **y_from_ball** → vertical distance from the ball for the individual player one second before the snap
- **position_group_num** → described above
- **mins_remaining** → minutes remaining in the game
- **defenseScoreMargin** → difference between the defensive team’s score and the offense’s
- **new_dir** → *playDirection*-adjusted direction of player
- **new_o** → *playDirection*-adjusted orientation of player
- **[positionGroup]_on_field** → number of each position group on field
- **x_from_ball_[positionGroupNum]** → horizontal distance from the ball for each possible position group number one second before the snap (0 if not on field)
- **y_from_ball_[positionGroupNum]** → vertical distance from the ball for each possible position group number one second before the snap (0 if not on field)
- **players_in_box** → number of defensive plays in "the box" *(within 5 yards of ball both vertically and horizontally)*

Using these variables, a **randomForest** model was created, resulting in the following variable importance graph:

In [1]:
```
library("png")
pp <- readPNG("/kaggle/input/bigdatabowl-viz/rf_importance.png")
par(bg = 'grey', mar=c(0,0,0,0))
plot(x = c(0,1), y = c(0,1) , type="n")
rasterImage(pp,0,0,1,1)
```



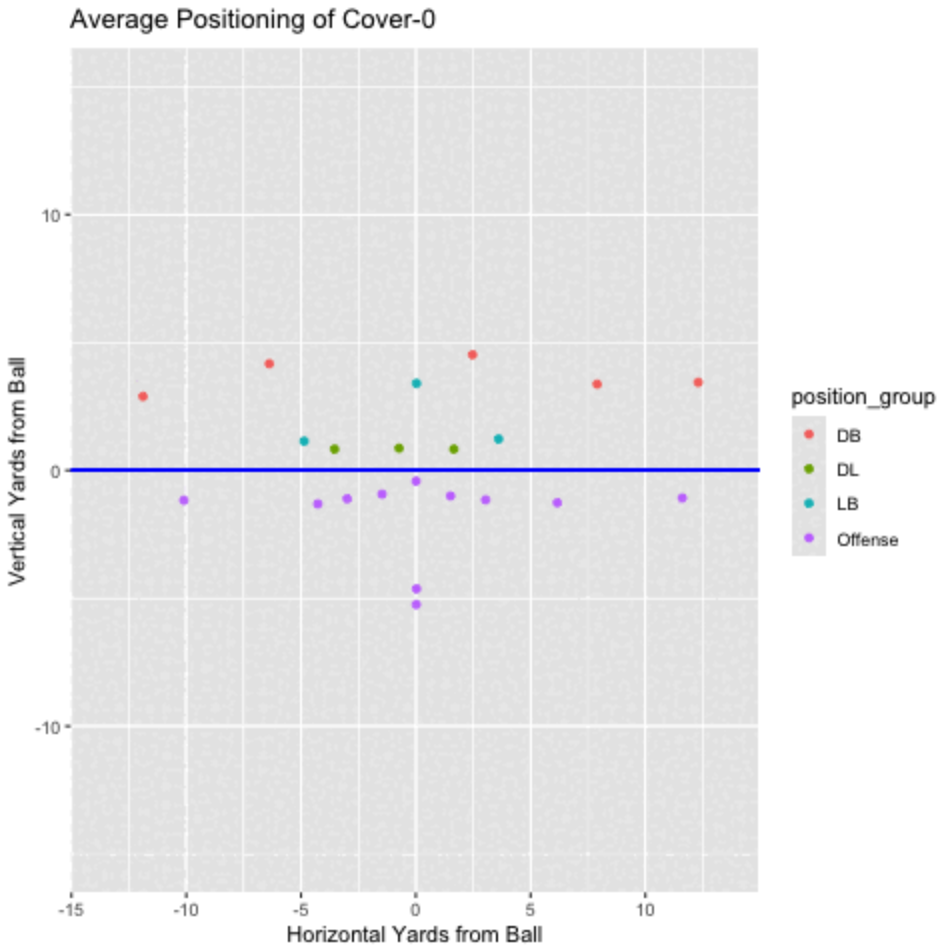
Unsurprisingly, the location and position of an individual player have a significant effect on their predicted assignment. Notably, the distance from the line of scrimmage for the **first** defensive back and the **second** linebacker strongly influences the model. This is likely due to the distinct roles these players assume based on the general defensive coverage. This animation illustrates how these players are typically positioned in different coverages. Pay close attention to the red dot furthest to the left and the middle blue dot *(note: the x and y variables are flipped in all field-level visualizations)*.

In [2]:

```
library("magick")
gif <- image_read("/kaggle/input/bigdatabowl-viz/average_positioning.gif")
gif
```

```
Linking to ImageMagick 6.9.11.60
Enabled features: fontconfig, freetype, fftw, heic, lcms, pango, webp,
x11
Disabled features: cairo, ghostscript, raw, rsvg


Using 4 threads
```

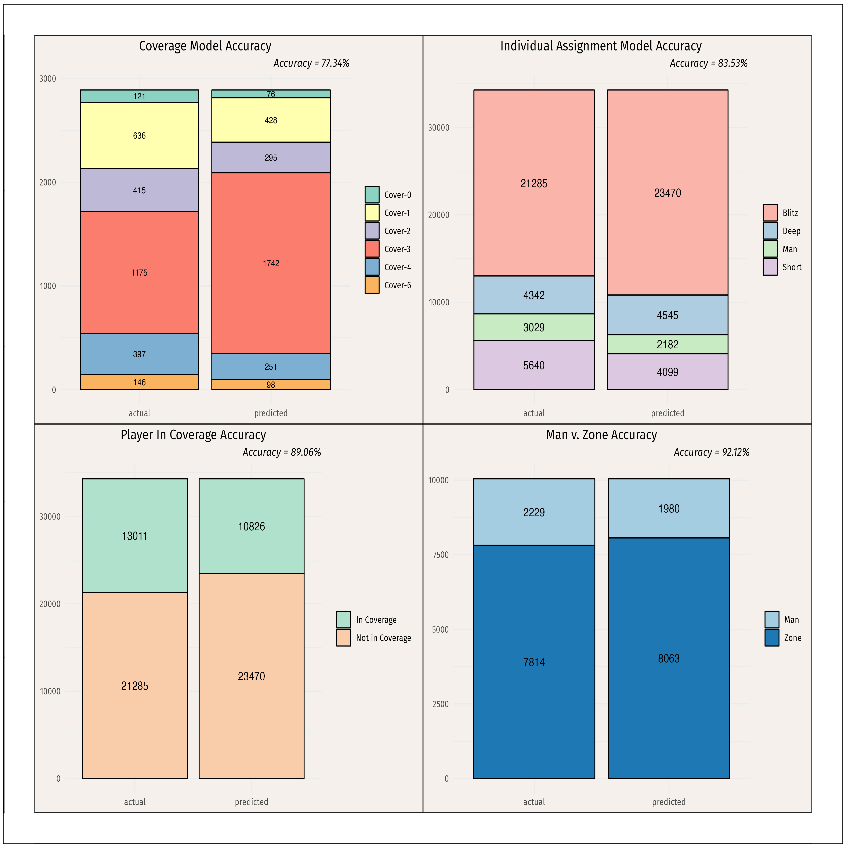


```
# A tibble: 100 × 7
   format width height colorspace matte filesize density
   <chr>  <int>  <int> <chr>      <lgl>    <int> <chr>
 1 GIF      480    480 sRGB       FALSE        0 72x72
 2 GIF      480    480 sRGB       FALSE        0 72x72
 3 GIF      480    480 sRGB       FALSE        0 72x72
 4 GIF      480    480 sRGB       FALSE        0 72x72
 5 GIF      480    480 sRGB       FALSE        0 72x72
 6 GIF      480    480 sRGB       FALSE        0 72x72
 7 GIF      480    480 sRGB       FALSE        0 72x72
 8 GIF      480    480 sRGB       FALSE        0 72x72
 9 GIF      480    480 sRGB       FALSE        0 72x72
10 GIF      480    480 sRGB       FALSE        0 72x72
# ℹ 90 more rows
```

Overall, the model performed exceptionally well. The following graphics illustrate the model's success in predicting overall coverage (based on predicted assignments), individual assignments, whether a player was in coverage, and, for all players in coverage, the accuracy of man vs. zone predictions.

In [3]:

```
library("png")
pp <- readPNG("/kaggle/input/bigdatabowl-viz/Accuracies.png")
par(bg = 'grey', mar=c(0,0,0,0))
plot(x = c(0,1), y = c(0,1) , type="n")
rasterImage(pp,0,0,1,1)
```



IV. Optimizing Offense Play Calls: Data Cleaning

We tried to find the best practical use for our assignment predictions. Initially, we explored whether teams that "disguised" coverages (as determined by the model) achieved better defensive results, but no significant findings emerged. We then shifted our focus to optimizing offensive play calls based on the model's predictions. The first step in this process was creating an expected expected points added (**xEPA**) model based on the presnap locations of all 22 players and their actual assignments. Kneels, spikes, prevent and goal line defensive sets, and scrambles were filtered out. Scrambles were removed due to their unpredictability: quarterbacks tend to take off on "broken" plays, and even those marked as "designed" runs showed enough inconsistency that we decided it was better to omit them altogether. As a result, quarterback runs are considered infeasible plays and are not eligible in the optimization process–a limitation we acknowledge and believe could be addressed with more time.

Offensive assignments were divided into two fields: **offensiveAssignment** and **offensiveAssignment_specific**. **OffensiveAssignment** was categorized into the following groups, with corresponding numeric factor values for the feasible region:

- **BLOCKER (1):** Any player who isn't passing, running, or running a route *(the QB is considered a BLOCKER on rushing plays)*
- **PASSER (2):** The QB on a pass play
- **RUSHER (3):** The primary ball carrier on a rushing play
- **SHORT ROUTE (4):** Players running the following routes: Screen, Hitch, In, Slant, Angle, Out, or Flat
- **DEEP ROUTE (5):** Players running the following routes: Go, Wheel, Post, Cross, or Corner

The first xEPA model used **offensiveAssignment**, while the second model utilized **offensiveAssignment_specific**, which breaks the five categories into more detailed subcategories:

- **BLOCKER:**
  - **BLOCKER (1):** Any blocking player
- **PASSER:**
  - **PASSER (2):** The quarterback on a pass play
- **RUSHER:**
  - **INSIDE_LEFT (3):** Running inside to the left of the center
  - **INSIDE_RIGHT (4):** Running inside to the left of the right
  - **OUTSIDE_LEFT (5):** Running outside to the left
  - **OUTSIDE_RIGHT (6):** Running outside to the right
- **SHORT ROUTE:**
  - **SHORT ROUTE (7):** Players running the following routes: Screen or Hitch
  - **SHORT-IN ROUTE (8):** Players running the following routes: In, Slant, or Angle

    - **SHORT-OUT ROUTE (9):** Players running the following routes: Out or Flat
- **DEEP ROUTE:**
    - **DEEP-IN ROUTE (10)** Players running the following routes: Post or Cross
    - **DEEP-OUT ROUTE (11)** Players running the following route: Corner
    - **DEEP ROUTE (12)** Players running the following routes: Go or Wheel

V. Optimizing Offense Play Calls: xEPA Models

Both **xEPA** models were **xGBoost** models, tuned using the **Caret** library. The models used similar game situation variables as the prior model (**week**, **quarter**, **down**, **yardsToGo**, **defensiveTeam**, **yards_from_endzone**, **mins_remaining**, **defenseScoreMargin**) as well as the assignments and locations of all 22 players on the field. The parameters were similar for both models, with the same maximum tree depth, eta, and minimum child weight. However, the specific model had a much higher gamma value (3.33 vs. 0.22), which likely made it less prone to overfitting. Both models used a 5-fold cross-validation control method for tuning and employed the **xgbTree** method. **xgbLinear** models were attempted but yielded worse results. The following results were found:

General Model

- RMSE: 1.187
- Average Error: 0.035
- Average Absolute Error: 0.898

Specfic Model

- RMSE: 1.156
- Average Error: 0.018
- Average Absolute Error: 0.875

Clearly, the specific model performed better, but both models were generally good.

VI. Optimizing Offense Play Calls: Best Feasible Solution

After creating these models, the next step was to define a feasible region on which to test and find the best play call. Note that all personnel combinations deemed "infeasible" were excluded prior to training the **xEPA** models. The original feasible region contained the following possibilities for the **general** offensive assignments:

- **OL1-5**: *BLOCKER*
  - This ensures that five offensive linemen are always on the field and blocking. Plays with more than five linemen were filtered out, ensuring only basic formations were used.
- **QB1**: *BLOCKER* or *PASSER*
  - QB is a **BLOCKER** on a rushing play and a **PASSER** on a passing play.
  - Plays where a non-QB threw a pass were filtered out.
- **RB1-2**, **FB1**: *BLOCKER*, *RUSHER*, *SHORT ROUTE*, *DEEP ROUTE*, or *NONE*
- **WR1-5**, **TE1-2**: *BLOCKER*, *SHORT ROUTE*, *DEEP ROUTE*, or *NONE*
  - Non traditional runs were filtered out.
- **RB3**, **TE3**, **QB2**, **OL6-7**: *NONE*
  - No plays with this personnel are allowed.

Following this, constraints on the feasible region were added. First, we defined the following variables:

- X_B = the number of blockers
- X_{WRB} = the number of wide receivers blocking
- X_P = the number of passers
- X_R = the number of rushers
- X_{RR} = the number of route runners

Then, we added the following contraints:

- $X_B + X_R + X_P + X_{RR} = 11$
  - *Exactly 11 offensive players on field*
- $X_B \geq 5$
  - *At least 5 blockers*
- $X_P + X_R = 1$
  - *Exactly one player passing OR rushing*
- $X_P + (X_P * X_{RR}) + X_R + X_B = 11$
  - *Ensures that a rushing play has 10 blockers and zero routes*
- $X_B + X_P \leq 11$
  - *Ensures each passing play has at least one route runner*
- $X_P * X_{WRB} = 0$
  - *Ensures receivers cannot block on passing plays*

After filtering out all possibilities that didn't match, a true feasible region was created. We then ran through a sample of 5000 plays. For each play, the following steps were followed:

1. **Filter by Personnel**

> The feasible region was limited to plays with personnel matching that of the offense on the actual play. The defensive assignments were directly related to the offensive personnel, so it was necessary to ensure the offensive play call matched the real-life positions.

2. **Find Best Solutions With the General xEPA Model**

> We found the **xEPA** for each possible combination of offensive assignments based on the predicted defensive assignments. We then filtered the possibilities to a data frame of **best_choices**, where the **xEPA** was equal to the maximum **xEPA** of all possibilities. Note that due to the utilization of the xgbTree model, there were multiple 'optimal' solutions.

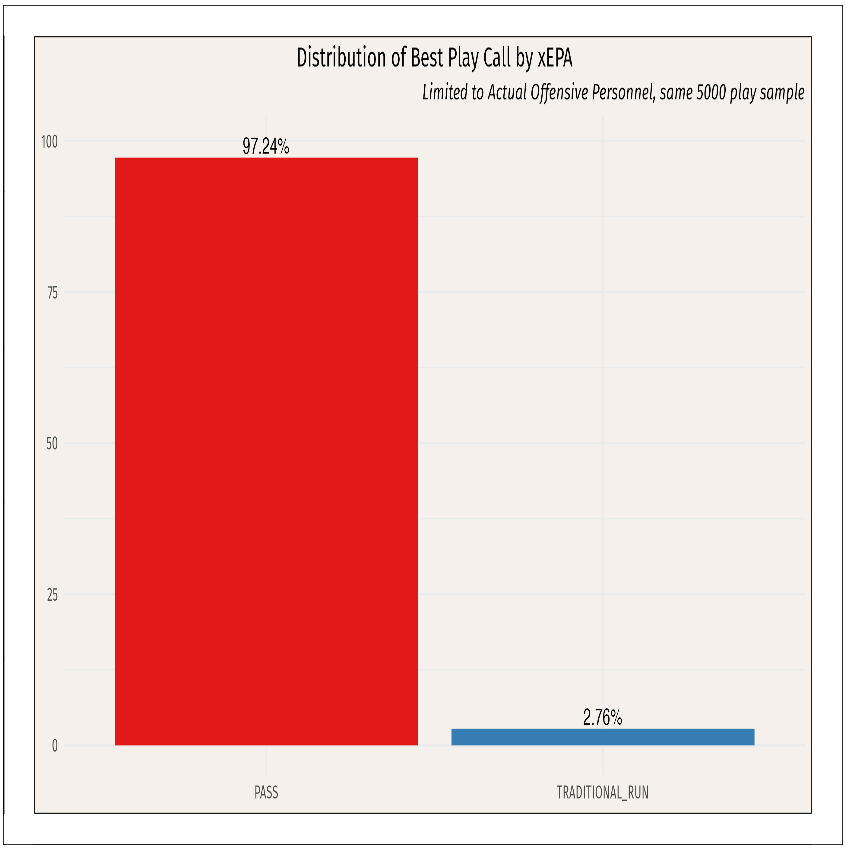
3. **Find Best Solutions With the Specific xEPA Model**

> Using the **best_choices** data frame from the prior step, we then expanded each assignment into its various possibilities (as discussed in section IV). For example, a play with 10 blocking assignments and RB1 "running" the ball was then broken up into four possibilities, based on the different possible directions and areas of the run. The following code found the xEPA for each possible combination of specific offensive assignments based on the predicted defensive assignments. The median xEPA of all offensive possibilities was used as the optimal solution.

VII. Optimizing Offense Play Calls: Results

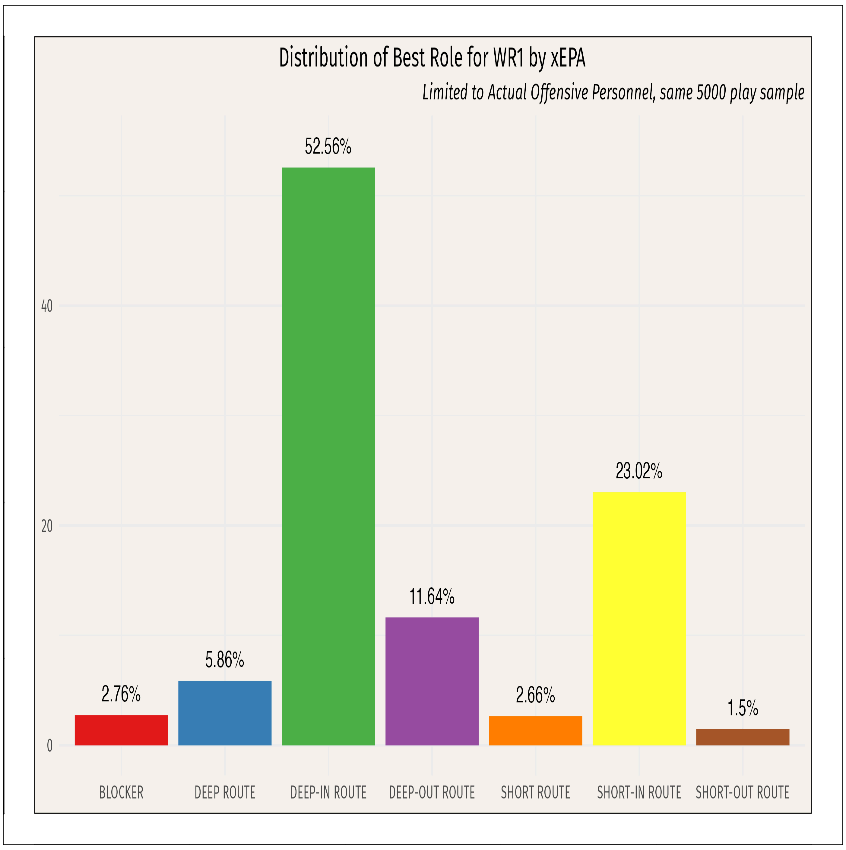
The model found the "optimal" play to be a passing play 97.24% of the time. While this clearly wasn't a realistic playbook for a team to have, it does align with EPA as a statistic, which often has higher values for passing plays. A reason for this discrepancy between real and hypothetical play calls is likely that the model couldn't account for the flow of the game. Factors like setting up plays, clock management, and player talent weren't incorporated into the model, which could result in discrepancies.

In [4]:
```
pp <- readPNG("/kaggle/input/bigdatabowl-viz/play_call2.png")
par(bg = 'grey', mar=c(0,0,0,0))
plot(x = c(0,1), y = c(0,1) , type="n")
rasterImage(pp,0,0,1,1)
```



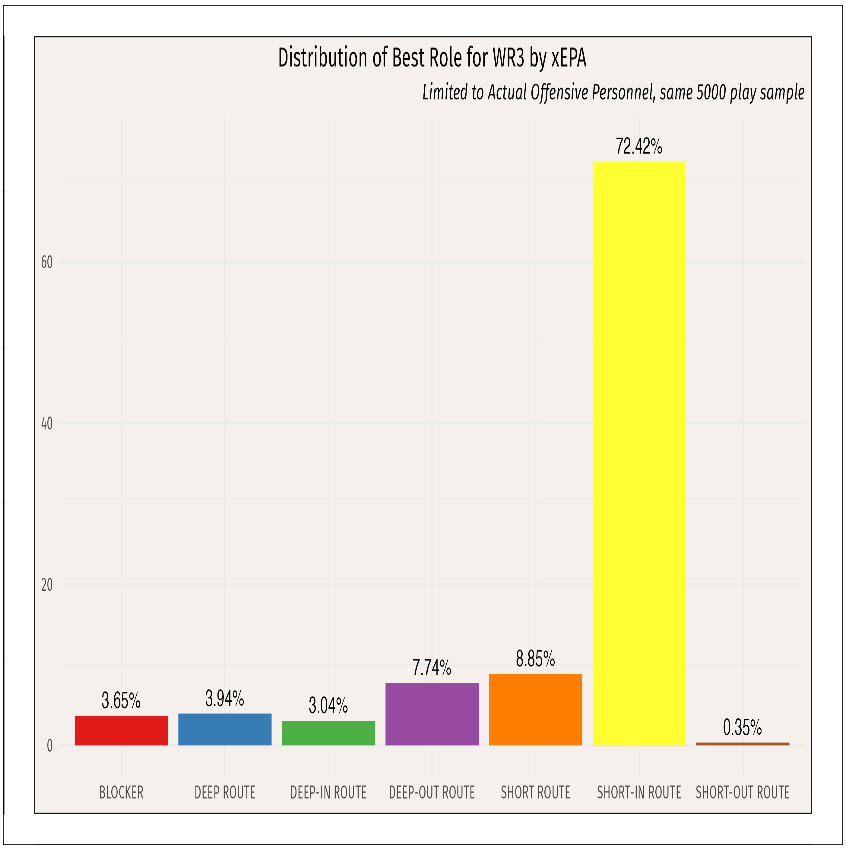
Looking at the 'best role' for a **WR1**, we see that the top receiver is often asked to run a 'deep-in' route.

In [5]:
```
pp <- readPNG("/kaggle/input/bigdatabowl-viz/WR1.png")
par(bg = 'grey', mar=c(0,0,0,0))
plot(x = c(0,1), y = c(0,1) , type="n")
rasterImage(pp,0,0,1,1)
```



When comparing this to the role of a **WR3**, who is often positioned on the bottom half of the field but can also be in the slot, we see that the model doesn't strictly call for deep routes and shows a diversity of calls.

In [6]:
```
pp <- readPNG("/kaggle/input/bigdatabowl-viz/WR3.png")
par(bg = 'grey', mar=c(0,0,0,0))
plot(x = c(0,1), y = c(0,1) , type="n")
rasterImage(pp,0,0,1,1)
```

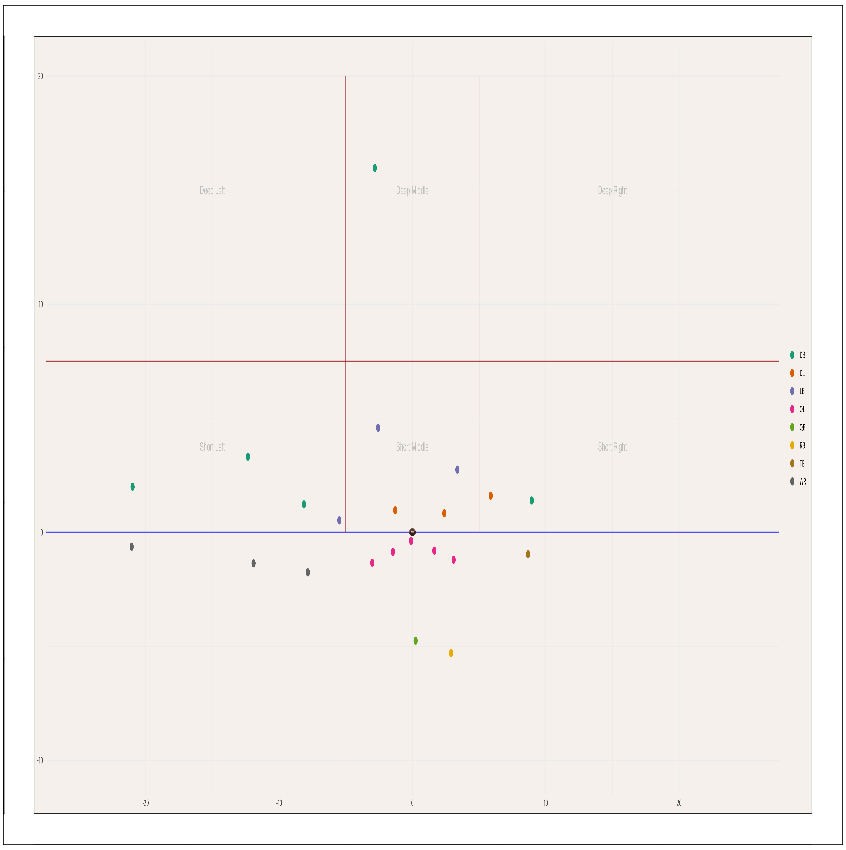


VIII. In Practice

Let's take a look at an example of how the model would treat a given play.

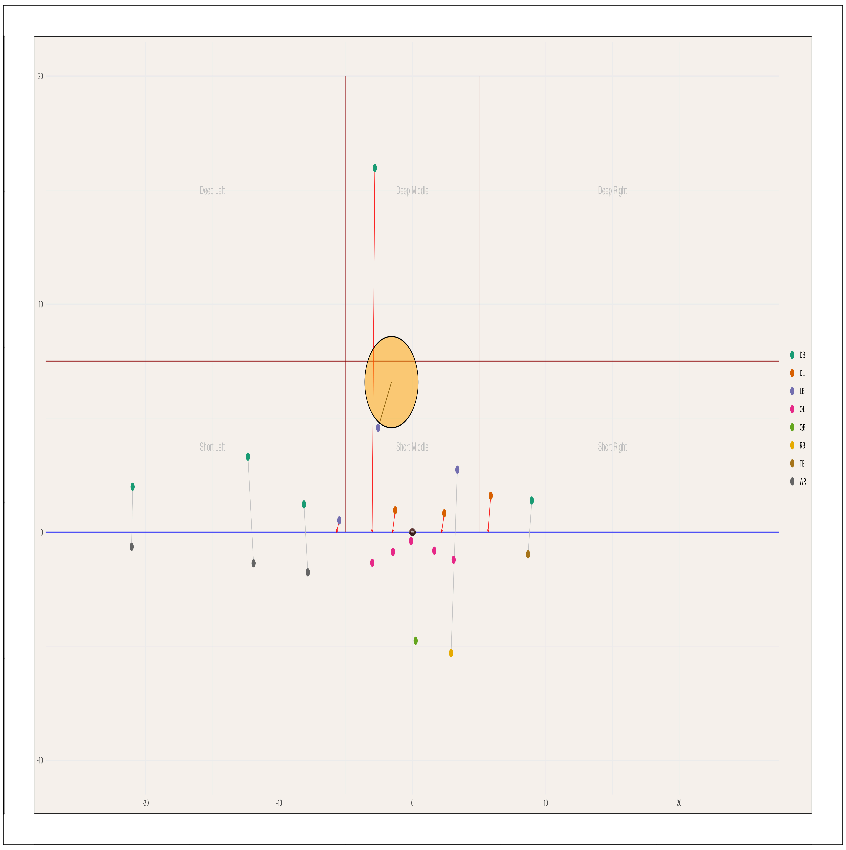
In [7]:

```
pp <- readPNG("/kaggle/input/bigdatabowl-viz/setup_plot.png")
par(bg = 'grey', mar=c(0,0,0,0))
plot(x = c(0,1), y = c(0,1) , type="n")
rasterImage(pp,0,0,1,1)
```



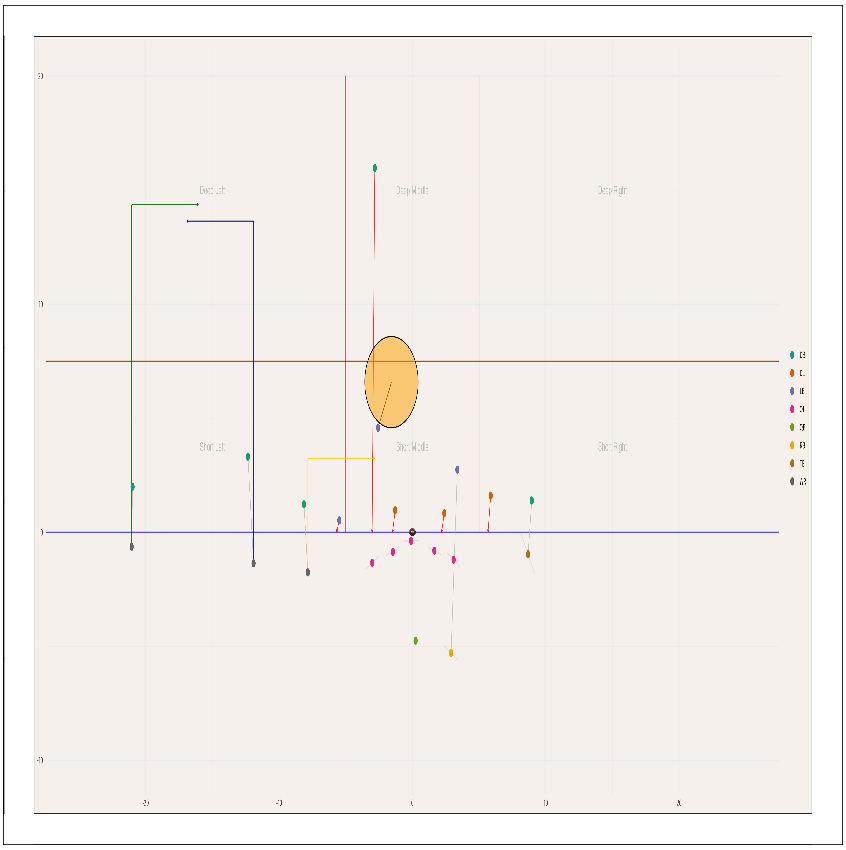
This is a first-quarter, 1st and 10 play from the Week 2, 2022 New England Patriots-Pittsburgh Steelers game. The Patriots have the ball at the Pittsburgh 47-yard-line. Looking at the personnel, the offense is in a pretty basic set (QB-RB-3WR-TE-5OL), and the defense is also in a basic set (3DL-3LB-5DB). Note that the Patriots' tight end, Jonnu Smith, is the only receiver on the right side of the offensive line, with all three wideouts positioned on the left side. The first step for the model is to "read the defense."

In [8]:
```
pp <- readPNG("/kaggle/input/bigdatabowl-viz/defense_added.png")
par(bg = 'grey', mar=c(0,0,0,0))
plot(x = c(0,1), y = c(0,1) , type="n")
rasterImage(pp,0,0,1,1)
```



The model predicts man-to-man coverage for all the defensive backs on the short level of the field, as well as for the right outside linebacker. It predicts a short zone for the middle linebacker and expects the remaining players, including the safety, to rush the QB. The "predicted" zone area and man-to-man matchup are based on very elementary linear models that should be improved upon. After reading the defense, the model then determines the optimal play call.

In [9]:
```
pp <- readPNG("/kaggle/input/bigdatabowl-viz/offense_added.png")
par(bg = 'grey', mar=c(0,0,0,0))
plot(x = c(0,1), y = c(0,1) , type="n")
rasterImage(pp,0,0,1,1)
```



Notice that the model calls for all wide receivers to run routes, with **WR1** running a deep-in, **WR2** a deep-out, and **WR3** a short-in. Believing a blitz is coming, it also calls for **RB1** to stay back to help block, as well as **TE1**. The play call clearly trusts its wide receivers in man coverage and wants to take advantage of that.

IX. Limitations

"Knowing that the defensive assignments are predicted with high accuracy, the 'optimal' play call is based on fair assumptions. That being said, there are certainly some limitations. Firstly, we allow for only a limited diversity of plays and formations. The model doesn't account for QB runs, WR runs, or other play types. Secondly, as mentioned earlier, the model rarely suggests a traditional run. With more time, we would have liked to investigate whether there is a way to bring in more context for a play, such as clock management or the 'flow of the game.' Finally, there is no real way to check the 'accuracy' of the optimal play, since coaches rarely call the optimal play within a game."

X. Conclusion

The analysis demonstrates the power of machine learning and optimization techniques within football strategy. Our Random Forest model successfully predicts defensive assignments with high accuracy, and by combining these predictions with an xGBoost-based xEPA model, we were able to evaluate play outcomes and identify the best player assignments for maximizing offensive efficiency.

Our findings highlight the potential to enhance decision-making in football, from predicting opponent strategies to optimizing play-calling. The model could be used in real-time to run through thousands of possible play calls in seconds, acting as a decision-making tool for the quarterback.

Acknowledgements

A big thanks to Prof. Chinthaka Kuruwita for his guidance on this project and for helping shape us as statistics students.