NFL Combine Score Module

CDT Bryson Daily, CDT Samin Kim

2024-05-06

Learning Goals

- Students will understand Logistic Regression Models
  - Be able to create a Logistic Regression Model
  - Be able to interpret the model properly
- Users will learn Introductory-Level Statistical Concepts
  - Be able to identify statistically significant variables
  - Be able to evaluate and compare models
- They will be introduced to Data Visualization
- Users will have grow in their Critical Thinking and Data interpretation Skills
- They will be able to see the importance of statistics and its Application to Real-World Problems

Introduction

Welcome to the 2024 NFL draft. You are an NFL GM in desperate need of a big-time wide receiver to turn your franchise around. You cannot afford to waste this pick, so make sure you know how to pick 'em. There is a lot of great college players going to be at Indianapolis this year competing at the NFL Combine, an annual event where college football's best players compete in athletic tests in hopes of impressing NFL scouts. There will be lightning fast 40-yard dashes, electric 3-cone drills, and vertical jumps that look like they are being done on trampolines. But which drills at the combine actually translate to future NFL success? You need to make sure your scouts don't get blinded by a certain dazzling combine drill performance if it doesn't lead to production on the field. Looking at the combine results starting in 2000, you need to see which players have had success in the NFL, and see if there is a correlation between them and the drills they performed well or poorly in at the NFL Combine so we can draft the next NFL superstar. While "superstar" can be subjective, our definition is defined as the top 20% of wide receivers in our WR value metric which accounts for yards and touchdowns, which are the most valuable stats for a wide receiver.

Here's a link of a video of NFL Combine of Wide Receivers:

https://www.youtube.com/watch?v=vf9tf3jY6ws&ab_channel=NFL

Methods / Instructional Content

This module will be based on two textbooks; Introduction to NFL Analytics with R and R for Data Science. Both textbook are available online for free, and the links for the online version of textbooks are down below.

- R for Data Science: https://r4ds.had.co.nz/index.html

In reading R for data science we were able to better understand the key concepts related to data cleaning and visualizations in data science. Other things we were able to take from this online textbook include data manipulation and data analysis. For our project, data cleaning and data manipulation are critical to get the project started as we have to make sure the data works with what we are trying to learn. Once those are accomplished, we are able to use visualizations and data analysis to better convey our results.

- Introduction to NFL Analytics with R: https://bradcongelio.com/nfl-analytics-with-r-book/

Similar to R for Data Science, the same key concepts of data science are covered, but, this is a more specific text to the exact league we are using. Using this we are able to get great examples of other data analysis projects in different areas around the league. By using this text we can see how the concepts we learned previously can be applied to real-world problems like professional football analytics.

Exercises / Activities

This module will involve multiple exercises. To successfully complete this module, users should practice the following exercises:

- Data Cleaning and Preparation: Users should be able to prepare data through filtering, joining multiple datasets, and handling outliers.
- Logistic Regression: Users should have understanding about logistic regression model to complete this module.
- Model Comparison: Users should be able to compare multiple models and determine the best fitted module.

Data

The following module utilizes datasets from two sources:

- https://www.kaggle.com/datasets/mitchellweg1/nfl-combine-results-dataset-2000-2022?resource=download
- https://www.kaggle.com/datasets/philiphyde1/nfl-stats-1999-2022/data

The two sources provide datasets containing the NFL combine results and NFL season statistics for every player. From the first source, we will use combine data from 2012 to 2019. From the second source, we will only use the yearly data.

The NFL combine result dataset (2012~2019) has total 2642 samples with 12 variables. The variables of this dataset are player name, position, school, height, weight, 40 year dash, vertical jump, and more.

The NFL yearly statistics dataset has total 1769 samples with 71 variables.The variables of this dataset are name, position, team, season, carries, rushing yards, and more.

Basic statistics for each dataset are shown below.

Required Packages

This module will use the following packages and are required to install prior to starting this module.

```
# These packages will be used in this module and must be installed
library(tidyverse)
library(dplyr)
library(readr)
library(broom)
library(knitr)
```

NFL Combine Dataset

The combined datasets are available from https://www.kaggle.com/datasets/mitchellweg1/nfl-combine-results-dataset-2000-2022?resource=download, and datasets from 2012 to 2019 must be installed. Make sure the files are saved in the same directory as your executing script file and set the working directory to source file location. If you have issue with setting the working directory, please follow instruction from this link: https://rpubs.com/em_/wdInR

Following code combines all datasets into a single dataframe.

```
# Read NFL combine data from 2012 to 2019.
combine2012 <- read.csv("2012_combine.csv")
combine2013 <- read.csv("2013_combine.csv")
combine2014 <- read.csv("2014_combine.csv")
combine2015 <- read.csv("2015_combine.csv")
combine2016 <- read.csv("2016_combine.csv")
combine2017 <- read.csv("2017_combine.csv")
combine2018 <- read.csv("2018_combine.csv")
combine2019 <- read.csv("2019_combine.csv")

# Create tag (year) to each observation.
combine2012 <- combine2012 %>%
  mutate(year = 2012)
combine2013 <- combine2013 %>%
  mutate(year = 2013)
combine2014 <- combine2014 %>%
  mutate(year = 2014)
combine2015 <- combine2015 %>%
  mutate(year = 2015)
combine2016 <- combine2016 %>%
  mutate(year = 2016)
combine2017 <- combine2017 %>%
  mutate(year = 2017)
combine2018 <- combine2018 %>%
  mutate(year = 2018)
combine2019 <- combine2019 %>%
  mutate(year = 2019)

# Combine all dataframes from 2012 to 2019.
combined_data <- bind_rows(combine2012, combine2013, combine2014, combine2015,
                           combine2016, combine2017, combine2018, combine2019)
```

```r
# Save the dataframe to csv file for later use.
write.csv(combined_data, "combined_data.csv", row.names = FALSE)

# Filter the position.
combined_data <- combined_data %>%
  filter(Pos == "WR")

# Select the necessary variables only for the summary.
combined_data_summary <- combined_data %>%
  select(Ht, Wt, X40yd, Vertical, Bench, Broad.Jump, X3Cone, Shuttle)
```

Basic statistics for Combined Dataset are shown below.

```r
summary(combined_data_summary)
```

```
##       Ht                  Wt            X40yd          Vertical
##  Length:370         Min.   :156.0   Min.   :4.220   Min.   :26.50
##  Class :character   1st Qu.:192.2   1st Qu.:4.440   1st Qu.:33.00
##  Mode  :character   Median :203.0   Median :4.510   Median :35.00
##                     Mean   :202.8   Mean   :4.509   Mean   :35.12
##                     3rd Qu.:214.0   3rd Qu.:4.570   3rd Qu.:37.00
##                     Max.   :243.0   Max.   :4.850   Max.   :45.00
##                                     NA's   :26      NA's   :45
##      Bench         Broad.Jump        X3Cone         Shuttle
##  Min.   : 4.00   Min.   :107.0   Min.   :6.490   Min.   :3.810
##  1st Qu.:11.00   1st Qu.:117.0   1st Qu.:6.820   1st Qu.:4.130
##  Median :14.00   Median :121.0   Median :6.955   Median :4.220
##  Mean   :14.05   Mean   :121.2   Mean   :6.960   Mean   :4.234
##  3rd Qu.:17.00   3rd Qu.:124.0   3rd Qu.:7.090   3rd Qu.:4.330
##  Max.   :27.00   Max.   :141.0   Max.   :7.550   Max.   :4.660
##  NA's   :76      NA's   :50      NA's   :124     NA's   :112
```

NFL Yearly Statistics

The NFL Yearly statistics are available from https://www.kaggle.com/datasets/philiphyde1/nfl-stats-1999-2022/data.

```r
# Read player's season data
season_data <- read.csv("yearly_data_updated_08_23.csv")

# Filter the position
season_data <- season_data %>%
  filter(position == "WR")

season_data_summary <- season_data %>%
  select(receiving_yards, receiving_tds)
```

Baisc statistics for season data are shown below.

```r
summary(season_data_summary)
```

```
##  receiving_yards  receiving_tds
##  Min.   :  -7.0   Min.   : 0.000
##  1st Qu.:  55.0   1st Qu.: 0.000
##  Median : 231.0   Median : 1.000
##  Mean   : 373.7   Mean   : 2.251
##  3rd Qu.: 597.0   3rd Qu.: 4.000
##  Max.   :1964.0   Max.   :18.000
```

Finally, with the data we filtered we merge two dataframes and create new variable whether the player was successful or not. We will also create new variable to determine the performance of the player.

Performance = receiving touchdowns $*$ 19.3 + receiving yards

We put different weights on the receiving touchdowns and receiving yards to point out the higher value of receiving touchdowns. The values of weights is based on the study conducted by by Stuart in 2008 of Pro Football Reference that states that having 19.3 receiving yards provides the same change in expected points as having a single receiving touchdown. receiving touchdown $*$ 19.3 = receiving yards

We will also use the annual average performance score for 3 seasons to determine whether the player was successful for the beginning of their career.

```
# Select only the variables we need and rename the variable
# so that we can join two dataframes
season_data <- season_data %>%
  select(name, season, receiving_tds, receiving_yards, games) %>%
  rename(., Player = name) %>%
  mutate(avg_receiving_tds = receiving_tds / games, avg_receiving_yards = receiving_yards / games) %>%
  mutate(career_score = avg_receiving_yards + avg_receiving_tds * 19.3)


# Create a function that will return 3 earliest seasons
get_3 <- function(data){
  data %>%
    arrange(season) %>%
    head(3)
}

# Dataframe now only has first 3 seasons for each player.
season_data_top3 <- season_data %>%
  split(pull(., Player) ) %>%
  map_df(get_3, .id = "Player")


# Calculate average career_score per game for 3 seasons.
season_data_top3 <- season_data_top3 %>%
  group_by(Player) %>%
  summarize(avg_career_score = sum(career_score)/3)


# Conduct Left_join so that we have a dataframe
# with combine result and their success.
df <- left_join(season_data_top3, combined_data,by='Player')

# We are dropping rows that do are missing values.
# We are dropping NAs since we will still have 121 subjects as a result
```

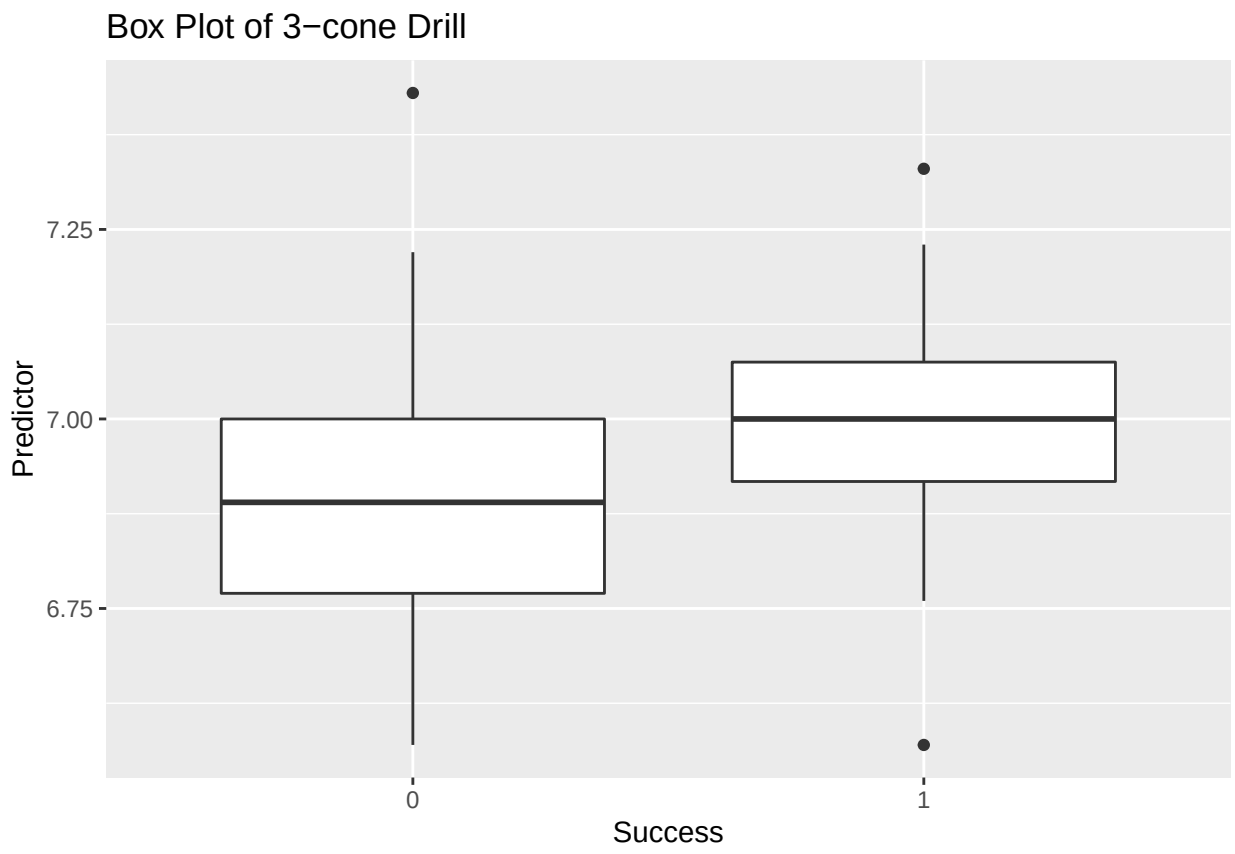
```r
# which is sufficient enough
df <- df %>%
  drop_na()


# If the player is in the top 20%, he will be determined as successful player.
# We will do this to the dataframe by creating new variable called success.

df <- df %>%
  # reorder the dataset by average career score for 3 years.
  arrange(desc(avg_career_score)) %>%
  # if the player was is top 20%, the player will flagged with '1'.
  mutate(success = ifelse(row_number() <= n() * 0.20, 1, 0))
```

With the complete dataframe, let's have some visualizations.

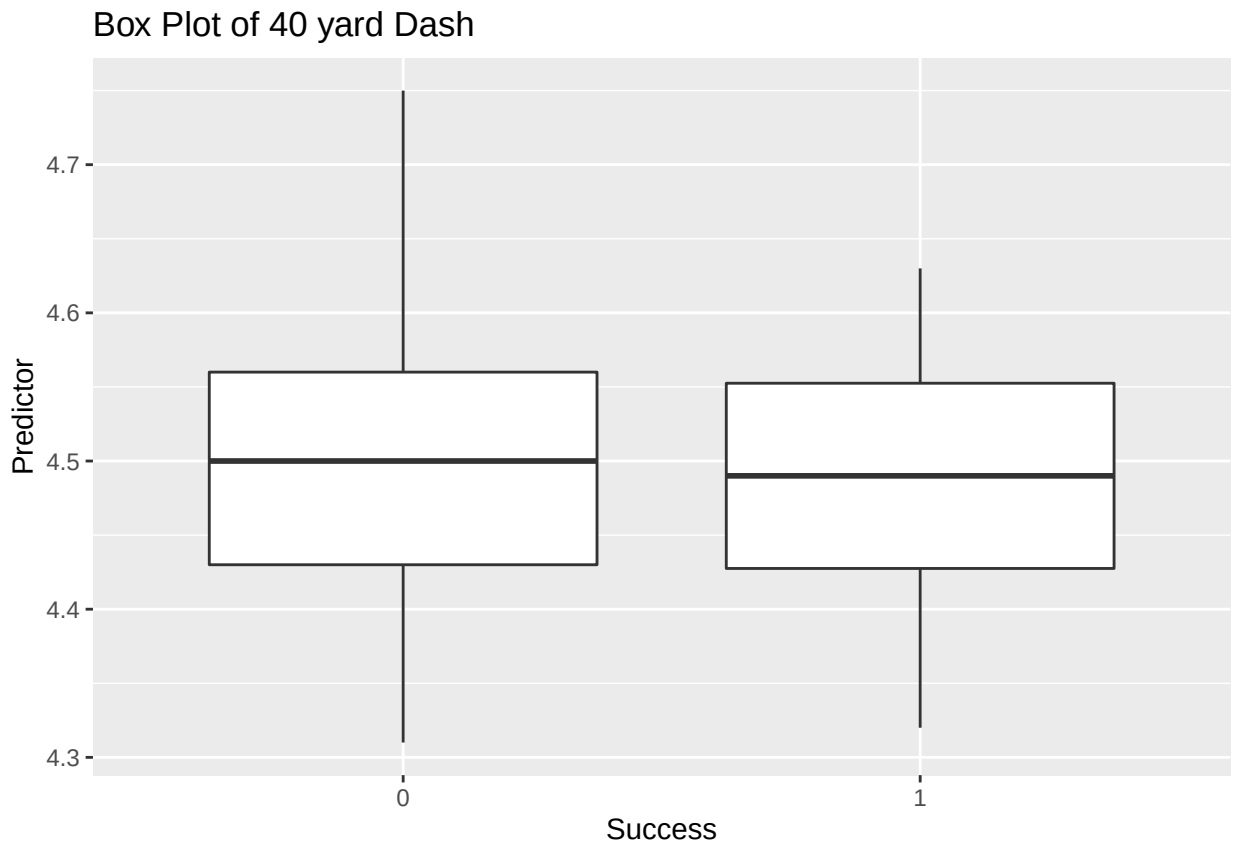
```r
# Box Plot for 3 Cone Drill
ggplot(df, aes(x = as.factor(success), y = X3Cone)) +
  geom_boxplot() +
  labs(
    title = "Box Plot of 3-cone Drill",
    x = "Success",
    y = "Predictor")
```



It seems like the higher score you have for the 3 cone drill, the more likely you are going to be successful in the future.

Let's also create a box plot for the 40 yard dash dril.

```
# Box Plot for 40 Yard Dash
ggplot(df, aes(x = as.factor(success), y = X40yd)) +
  geom_boxplot() +
  labs(
    title = "Box Plot of 40 yard Dash",
    x = "Success",
    y = "Predictor")
```



There's not much distinction between successful players and not-successful players. It seems like the there's not much association between 40 yard dash and success. We will find out more about this with the statistical models.

Logistic Regression Model.

Logistic regression model is an effective model to predict the outcome for binary variable. In this case, the response variable is success, which is binary. Here is the equation for logistic regression model.

$p = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$

$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 \cdot x_1$

Where p represents the probability of the outcome, x represents the predictor variable. Each beta represents the coefficients for each variable, and β_0 represents the intercept. $\log\left(\frac{p}{1-p}\right)$ represents log odds, which is the ratio of success to failure.

Using Logistic regression model, we can find out the relationship between the success of wide receiver and the NFL combine drills.

Logistic Regression Model with Single variable

Now that we have a complete dataframe ready to create logistics regression model, we will create logistics regression model for each combine drill.

```
log_yd <- glm(success ~ X40yd, data = df, family = binomial())

log_vertical <- glm(success ~ Vertical, data = df, family = binomial())

log_bench <- glm(success ~ Bench, data = df, family = binomial())

log_jump <- glm(success ~ Broad.Jump, data = df, family = binomial())

  log_cone <- glm(success ~ X3Cone, data = df, family = binomial())

log_shuttle <- glm(success ~ Shuttle, data = df, family = binomial())
```

With 6 models, we will compare the predictor variables and determine which drill is most influential. To determine the influence of the predictor variable, we must take a look at the coefficients and p-values.

The p-value is often used as a method to determine the statistical significance of the variable. If the p-value is less than 0.05 but higher than 0.01, it is often determined that it has statistical significance. If the p-value is less than 0.01, we can determine that it has very high statistical significance.

The coefficient of the predictor variable represents the change in log odds of response variable for every unit change in the predictor variable. The larger coefficient indicates larger change in log odds for every unit change in predictor variable.

Let's compare the coefficients and p-values of each variable from all 5 models.

```
# Create table for comparison
coef_summary <- data.frame(
  Model = c("40yd", "Vertical", "Bench", "Broad Jump", "3Cone", "Shuttle"),
  Coefficient = c(
    coef(log_yd)["X40yd"],
    coef(log_vertical)["Vertical"],
    coef(log_bench)["Bench"],
    coef(log_jump)["Broad.Jump"],
    coef(log_cone)["X3Cone"],
    coef(log_shuttle)["Shuttle"]
  ),
  pValue = c(
    summary(log_yd)$coefficients["X40yd", "Pr(>|z|)"],
    summary(log_vertical)$coefficients["Vertical", "Pr(>|z|)"],
    summary(log_bench)$coefficients["Bench", "Pr(>|z|)"],
    summary(log_jump)$coefficients["Broad.Jump", "Pr(>|z|)"],
    summary(log_cone)$coefficients["X3Cone", "Pr(>|z|)"],
    summary(log_shuttle)$coefficients["Shuttle", "Pr(>|z|)"]
  )
)
```

```r
# Calculate Odds Ratios as well
coef_summary$OddsRatio <- exp(coef_summary$Coefficient)


# Print the Summary
print(coef_summary)
```

```
##                 Model Coefficient     pValue  OddsRatio
## X40yd            40yd -1.69269712 0.50965975  0.1840225
## Vertical     Vertical  0.04089794 0.59376749  1.0417458
## Bench           Bench  0.08064762 0.16252763  1.0839889
## Broad.Jump Broad Jump -0.01768674 0.66596808  0.9824688
## X3Cone          3Cone  3.16026518 0.02329972 23.5768472
## Shuttle       Shuttle  1.88804275 0.25457611  6.6064256
```

Based on the result, the only variable that had a statistical significance was 3-Cone drill and had the highest odds ratio. The 3-Cone drill had a coefficient of 3.16 with the p-value of 0.002, and an odds ratio of 23.57. This means that for every unit increase in 3-cone drill (X3Cone), there will be 3.86 increase in log odds of being successful. The odds ratio of 3-cone drill shows that the odds of success are multiplied by 23.57 for every unit increase in 3-cone drill.

Logistic Regression Model with Multiple Predictor Variables

We can also create logistic regression model with multiple predictor variables. The equation is the following:

$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + ... + \beta_n \cdot x_n$

With 6 drills for NFL combine, we will have all six drills as predictor variable to our model.

```r
log_multiple <- glm(success ~ X40yd + Vertical + Bench + Broad.Jump + X3Cone +
                      Shuttle, data = df, family = binomial())

multiple_result <- summary(log_multiple)

print(multiple_result)
```

```
##
## Call:
## glm(formula = success ~ X40yd + Vertical + Bench + Broad.Jump +
##     X3Cone + Shuttle, family = binomial(), data = df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2625  -0.7197  -0.5327  -0.3538   2.3719
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -12.74667   17.51552  -0.728   0.4668
## X40yd        -2.89837    3.00131  -0.966   0.3342
## Vertical      0.08973    0.10742   0.835   0.4035
## Bench         0.05820    0.06622   0.879   0.3795
## Broad.Jump   -0.05059    0.05496  -0.921   0.3573
```

```
## X3Cone        3.24378    1.73979   1.864   0.0623 .
## Shuttle       0.94481    2.04397   0.462   0.6439
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 120.54  on 120  degrees of freedom
## Residual deviance: 111.35  on 114  degrees of freedom
## AIC: 125.35
##
## Number of Fisher Scoring iterations: 4
```

Similar to the models with single variable, we can see that 3 cone drill has lowest the p-values and also has the highest coefficient.

Performance Comparison (Goodness-of-Fit)

One method to compare a model using goodness-of-fit test is use Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) values. It's an effective mothod to compare the goodness-of-fit of statistical models with same data set. The lower AIC and BIC value indicates better fit. The followings are the equations for AIC and BIC:

$AIC = -2 * \ln L + 2 * k$ $BIC = -2 * \ln L + 2 * \ln N * k$ where $L =$ likelihodd, $N =$ number of observations, $k =$ number of parameters

```
aic_single <- AIC(log_cone)
aic_multiple <- AIC(log_multiple)

bic_single <- BIC(log_cone)
bic_multiple <- BIC(log_multiple)

model_comparison <- data.frame(
  Model = c("Single Predictor Variable Model",
            "Multiple Predictor Variable Model"),
  AIC = c(aic_single, aic_multiple),
  BIC = c(bic_single, bic_multiple)
)

print(model_comparison)
```

```
##                               Model      AIC      BIC
## 1   Single Predictor Variable Model 119.0857 124.6773
## 2 Multiple Predictor Variable Model 125.3536 144.9241
```

Based on the comparison, we can conclude that the logistic regression model with single predictor variable performs better than the model with multiple predictors.

Conclusions

For this module, we used two data, which are NFL combine data and NFL season data. As a result of data preparation, we were able to get a final dataset with 121 observations. We created a logistic regression

models for each NFL combine drill, and compared the p-value and coefficients. As a result, only 3-cone drill was statistically significance, and had a highest coefficients.

Based on our analysis using logistic regression model, we can conclude that 3-cone drill is the most important and significant variable to predict NFL wide receiver's future performance.

Future Works

To predict the whether the player will be successful or not, we created a binary variable to set logistic regression model. In the future, we can directly the predict the expected performance, or specifically receiving yards. To predict non-binary outcome, we can use linear regression model. If the variables do not show linear relationship, we can set generalized additive model to predict the player's future performance.