NFL Big Data
Tackle Analysis

By Patrick Su and Tom Zhou

TABLE OF CONTENTS

01

EDA

We use visualizations to understand the structure and distribution of the data

02

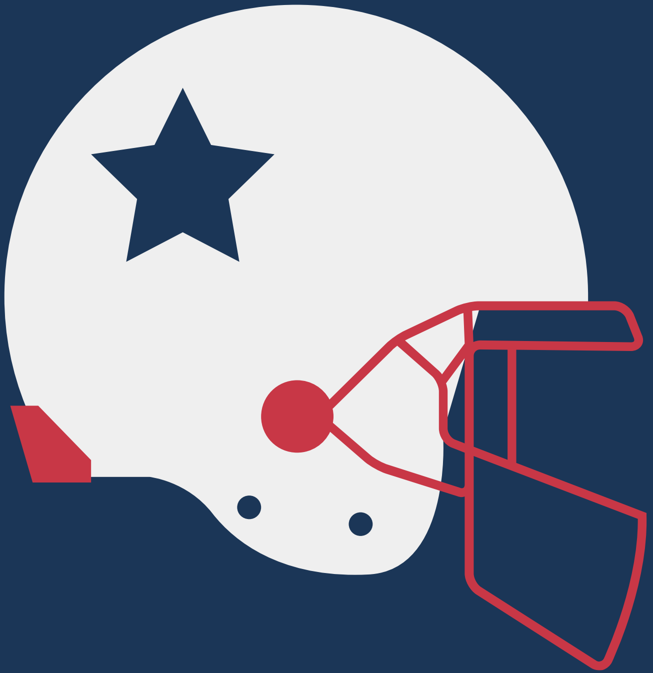
Metric: Yards Saved

We use data manipulation and modeling to design new metric for play evaluation

03

Results

Comparison of traditional tackling efficiency with the new yards saved metric.

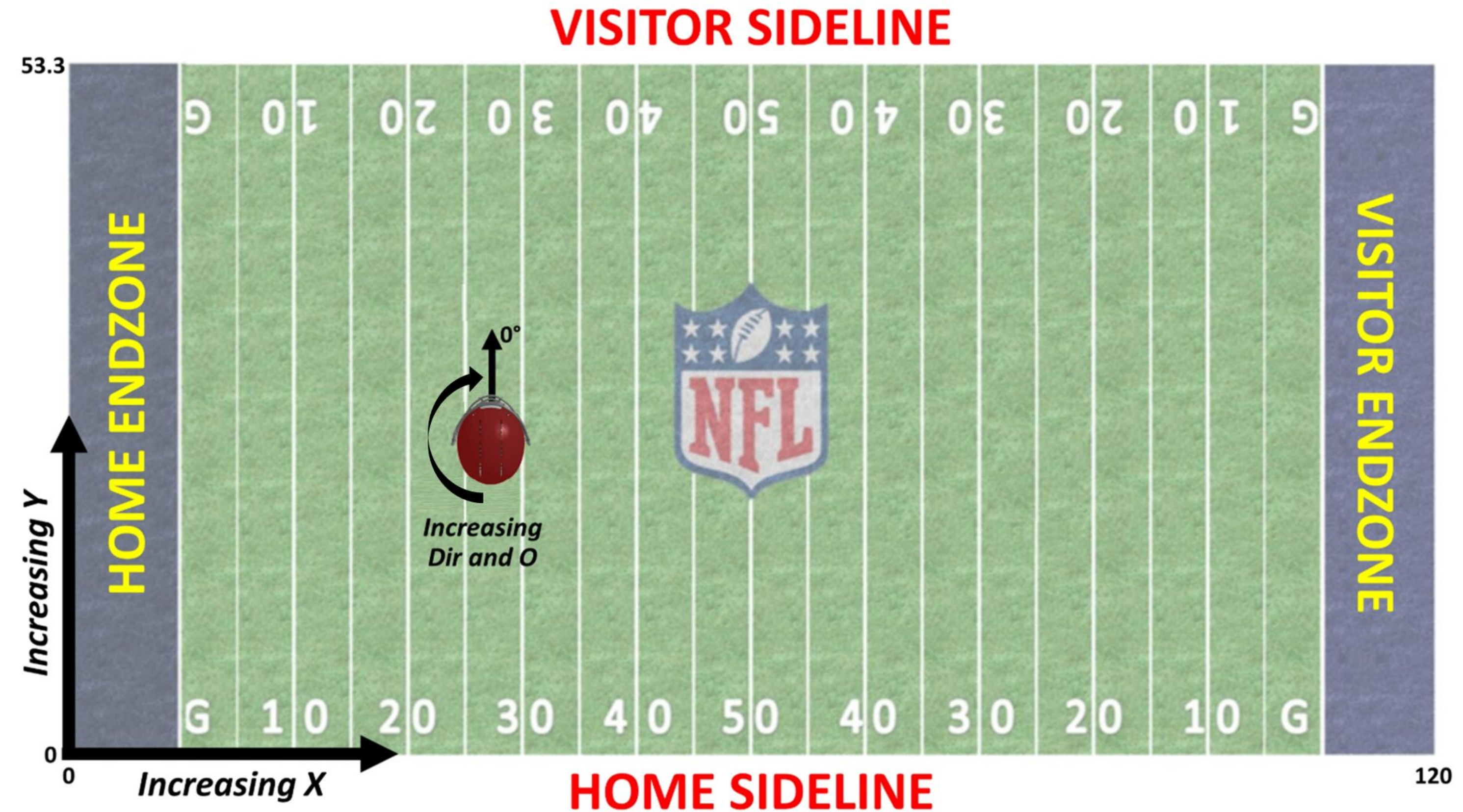


Part 01

Exploratory Data Analysis

Data Source: Provided by the NFL Next Gen Stats team on Kaggle

- Game Data: Basic structure of the season, listed out week of the game and teams involved
- Play Data: Dives into each game's specifics, like play description and quarter of the game
- Player Data: Understand the key players, their background information, and physical attributes
- Tackles Data: Insights into defensive plays and player performance
    - Key Fields: gameId, playId, nflId, tackle, assist, forcedFumble, pff_missedTackle
- Tracking Data: The Game in motion - records Detailed player movements and in-game dynamics
    - Key Fields: gameId, playId, nflId, x, y, s, a, dir, event (specific play types of a frame)


VISITOR SIDELINE
53.3
HOME ENDZONE
VISITOR ENDZONE
G 10 20 30 40 50 40 30 20 10 G
0°
Increasing
Dir and O
Increasing Y
0
0
Increasing X
HOME SIDELINE
120

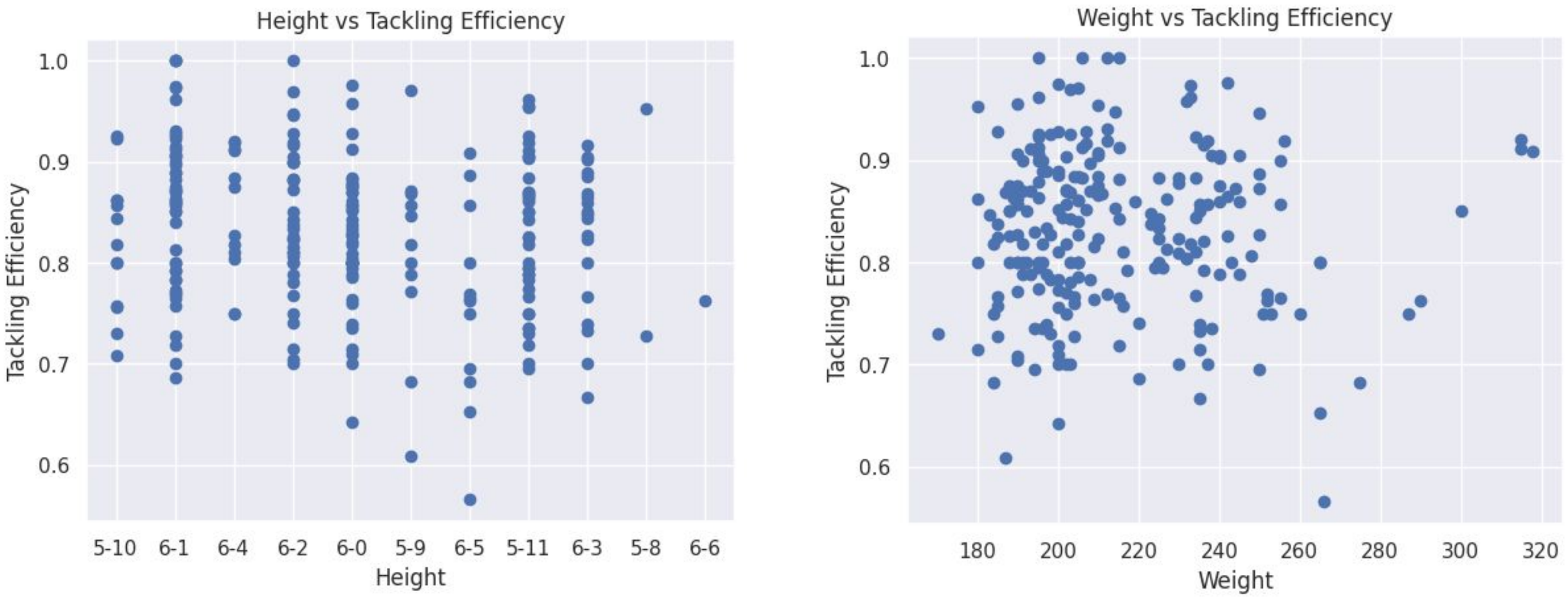
Objective

- Identify what makes a tackler not just good, but great?
- Understand each player’s strength, ultimately enhancing the team's defensive tactics on the field

The Traditional Measure: Tackle Efficiency

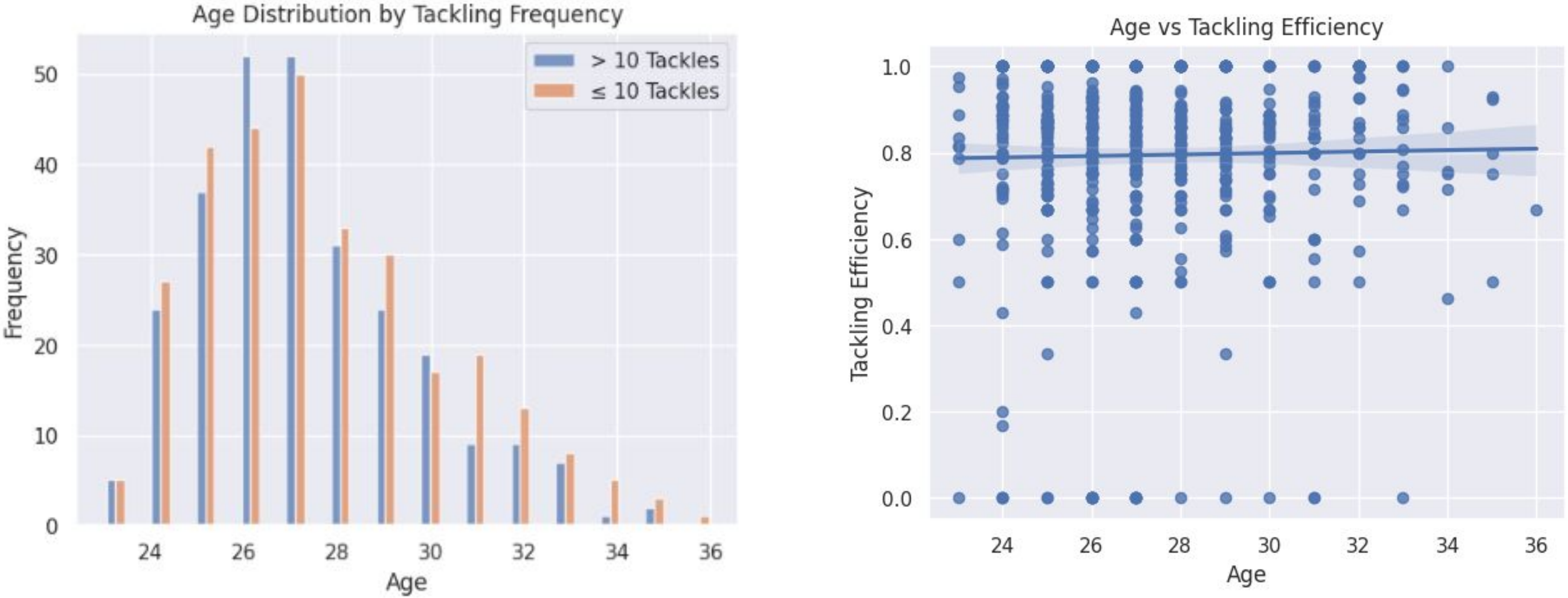
- Tackle Efficiency := Total Tackles / (Total Tackles + Total Missed Tackles)
    - The ratio of successful tackles to total tackle attempts, reflecting a player’s reliability in halting the opposition

Does Height and Weight affect Tackling Efficiency? (After filtering out total tackles < 20)

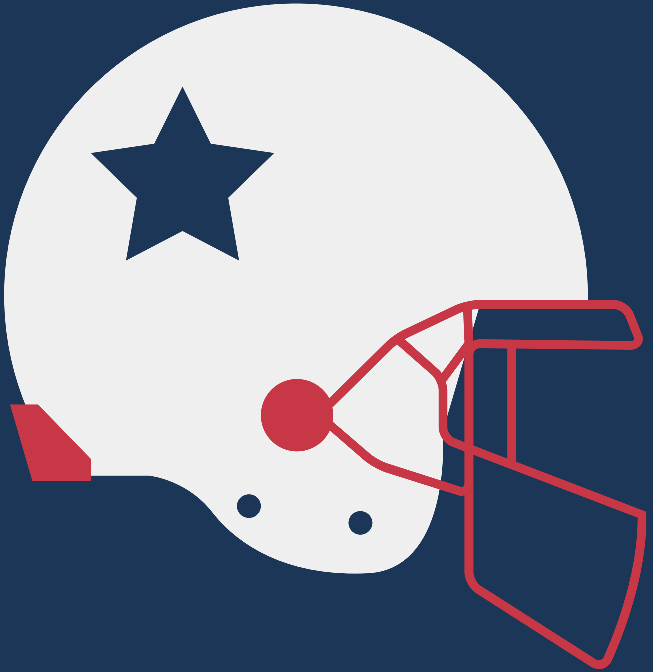


- Not really, since players are usually matched to defend players with similar physical attributes

Does Age matter for Tackling Efficiency



- No linear correlation, we will now use 10 tackles as a threshold for ranking consideration



Part 02

Metric:

Yards Saved

Introduction

- What is `nflBigData` class? What is it for?
  - Purpose: NFL Big Data Bowl datasets manipulations
  - A database with methods and pipelines
  - Inspired by https://www.kaggle.com/code/carrot1500/evaluating-tackle-difficulty

Class Attributes

- PLAYIDS: Key indices for dataset merging
- Datasets Used:
  - Tackles: Data on player tackles
  - Plays: Information on game plays
  - Players: Player profiles and statistics
  - Weekly Tracking: Player tracking data for weeks 1-9

Initialization

- Instance variables are initialized with class-level data
- Creation of empty lists for carrier and defender views

```python
class nflBigData:
    """
    A class for analyzing NFL Big Data Bowl datasets.

    This class includes methods for calculating various metrics related to player movements and interactions
    during NFL plays, using data from multiple sources, including tracking data, play information, and player
    attributes.

    Attributes:
        PLAYIDS (list): A list of strings used as primary keys to index the datasets.
        tackles (DataFrame): DataFrame containing tackle data.
        plays (DataFrame): DataFrame containing play data.
        players (DataFrame): DataFrame containing player data.
        weekly_tracking (list): A list of DataFrames containing weekly tracking data.
        carrier_list (list): A list to store DataFrames merged with ball carrier information.
        defender_list (list): A list to store DataFrames merged with defender information.
    """

    PLAYIDS = ["gameId", "playId"]  # Combination of indices used for merging datasets.

    # Reading CSV files containing tackles, plays, and player data.
    tackles = pd.read_csv("/content/drive/MyDrive/DSGA1007/data/tackles.csv")
    plays = pd.read_csv("/content/drive/MyDrive/DSGA1007/data/plays.csv")
    players = pd.read_csv("/content/drive/MyDrive/DSGA1007/data/players.csv")

    # Reading weekly tracking data for weeks 1 to 9.
    weekly_tracking = [pd.read_csv(f"/content/drive/MyDrive/DSGA1007/data/tracking_week_{i}.csv") for i in range(1, 10)]

    def __init__(self):
        # Initialize instance variables with class variables.
        self.PLAYIDS = nflBigData.PLAYIDS
        self.plays = nflBigData.plays
        self.players = nflBigData.players
        self.tackles = nflBigData.tackles
        self.trackings = nflBigData.weekly_tracking
        self.carrier_list = []  # Initialize empty list for merged DataFrames (ball carriers).
        self.defender_list = []  # Initialize empty list for merged DataFrames (defenders).
```

Data manipulation Methods

- Ball Carrier View Construction:
  - Merges tracking data with play and ball carrier information
  - Appends processed data to carrier list
- Defender View Construction:
  - Combines latest ball carrier view with tracking data
  - Identifies nearby defenders and blockers
- Physical Characteristics Construction:
  - Adds height and weight information to player data
  - Merges physical characteristics with defender data
- Labeling
  - Merges tackling data to label defenders’ actions

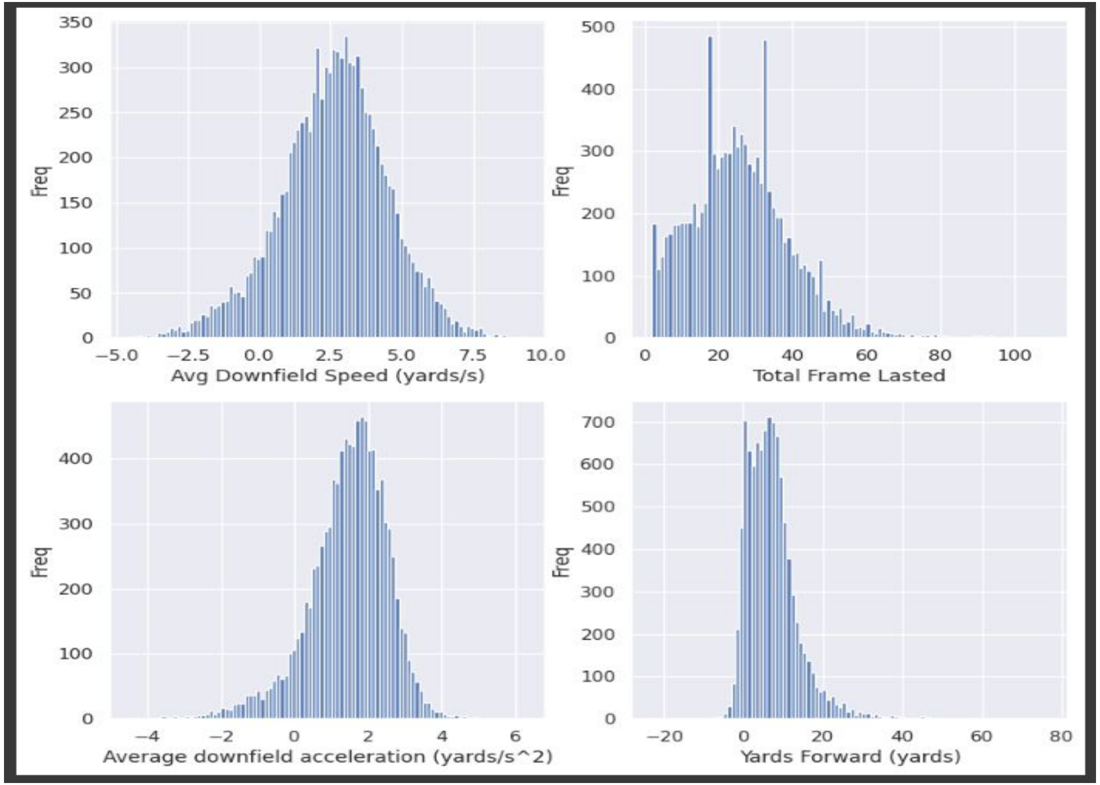
Calculation methods

- **Downfield Speed** calculation method
- **Downfield Acceleration** calculation method
- **Distance to Carrier** calculation method
- **Yards Forward** calculation method

Additional (outside-of-class) calculation and Data manipulation methods

- **Average Downfield Speed** calculation method
- **Average Downfield Acceleration** calculation method
- **Total frames lasted** calculation method
- **Total yards gained** calculation method

The distribution of the key quantities needed for metric calculation:



Now that we have...

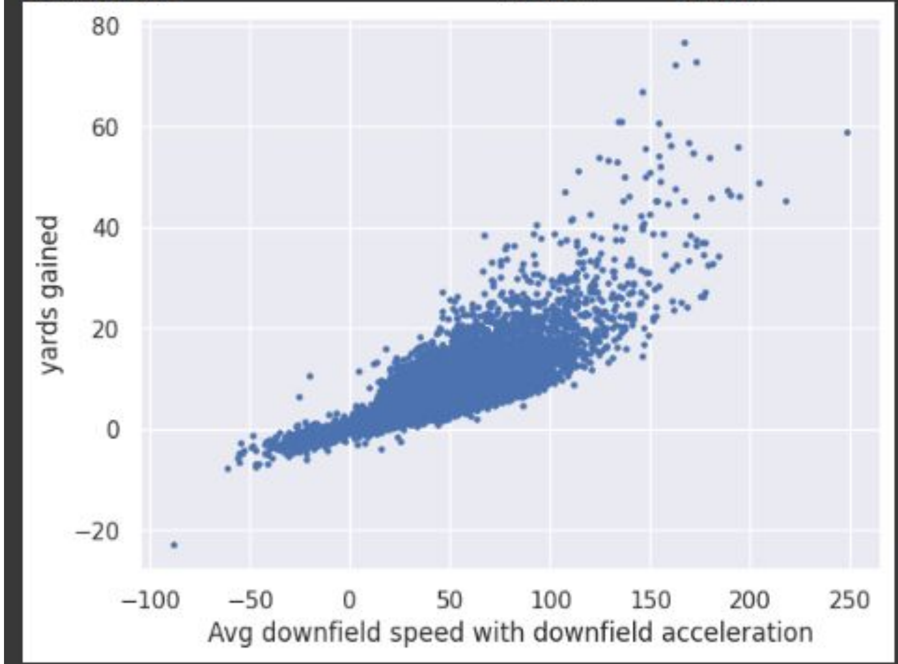
- the average ball carrier downfield speed in a play
- the average ball carrier downfield acceleration in a play
- the total number of frames lasted in a play

We can fit a linear regression model to...

- predict potential yards that a ball carrier can gain at any given frame in a play!
- Our assumption: yards ~ speed + time * acceleration

The correlation of the yards gained and downfield speed with acceleration:

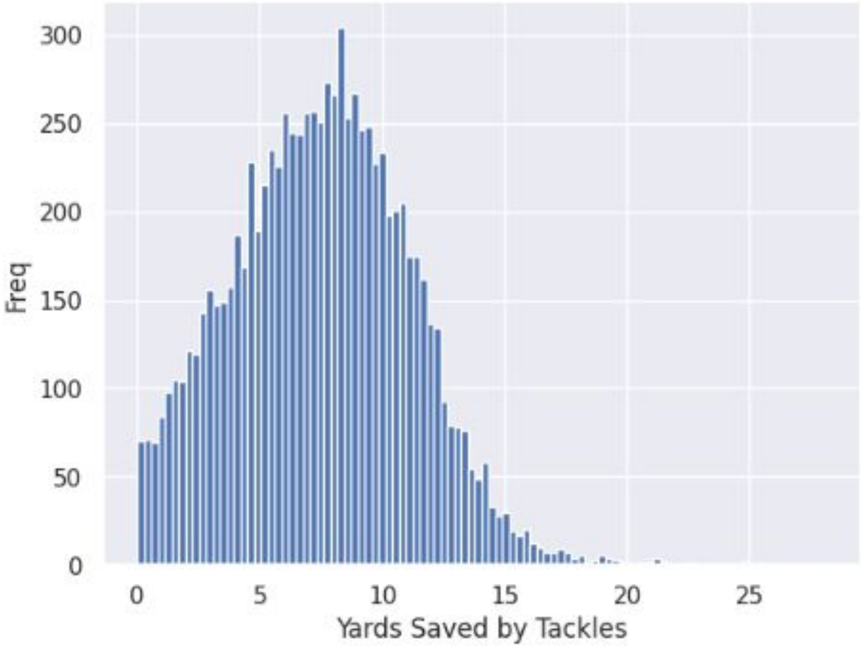
	avg_s_downfield_plus_acc	yards_forward
avg_s_downfield_plus_acc	1.000000	0.829891
yards_forward	0.829891	1.000000

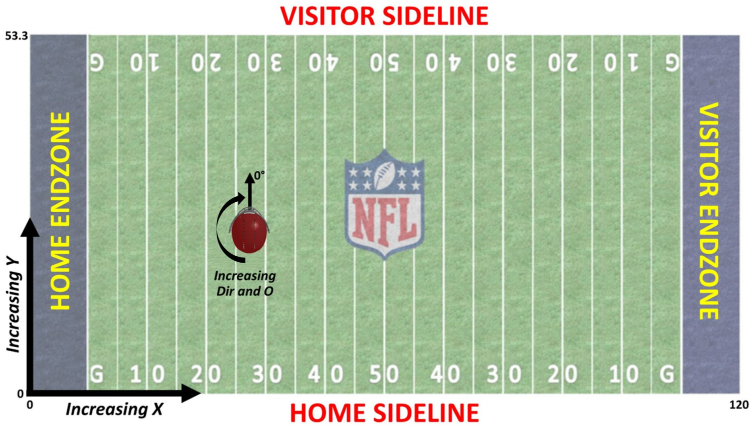


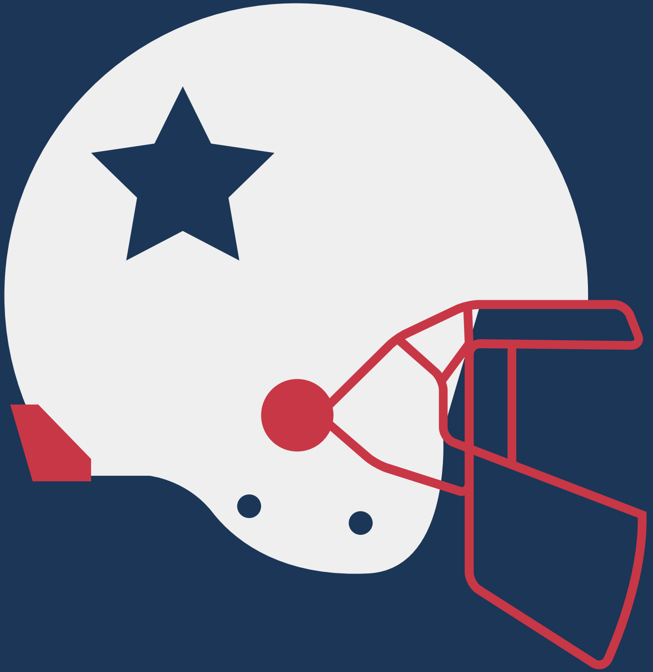
Our Metric to evaluate a tackle: Yards Saved

- Our LR can predict the potential yards will be gained at any given frame in a play
- We can then plug in the downfield speed of a ball carrier at the frame when he is successfully tackled by a defender
  - The output of the LR model can then be interpreted as:
    - **the yards the ball carrier could have gained if the defender had missed the tackle.**
    - Or **# of yards saved by a successful tackle**

The distribution of the Metric:

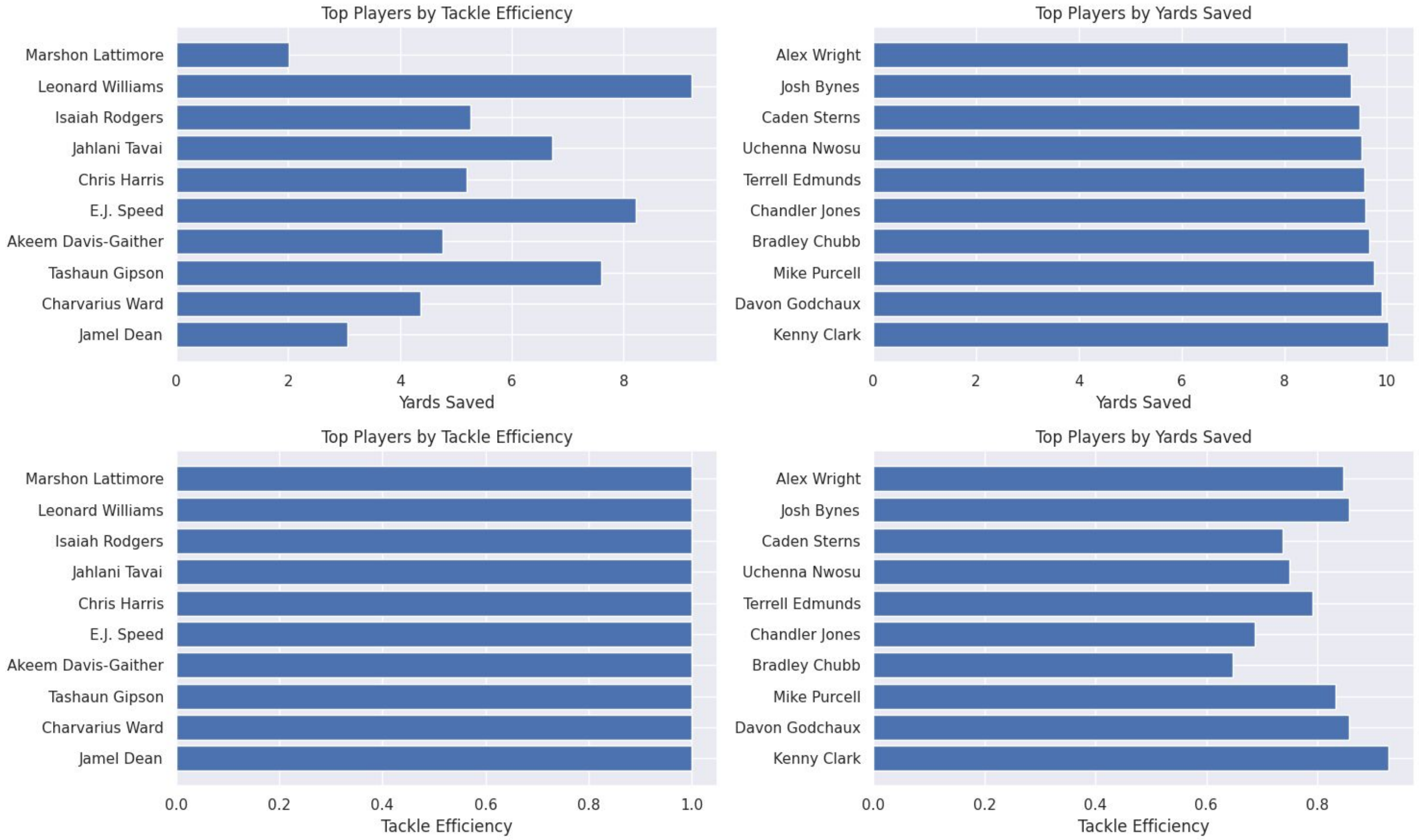




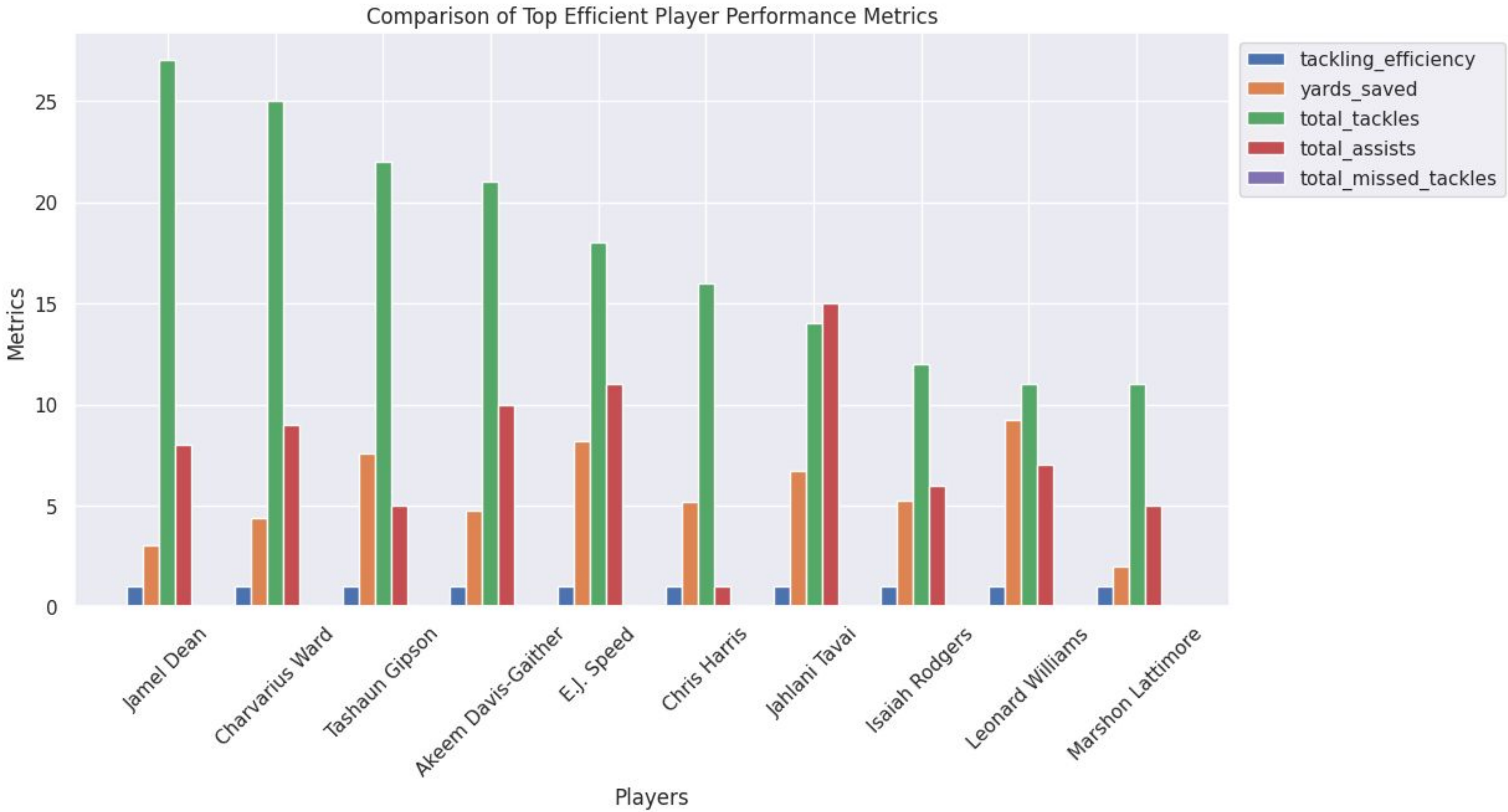


Part 03

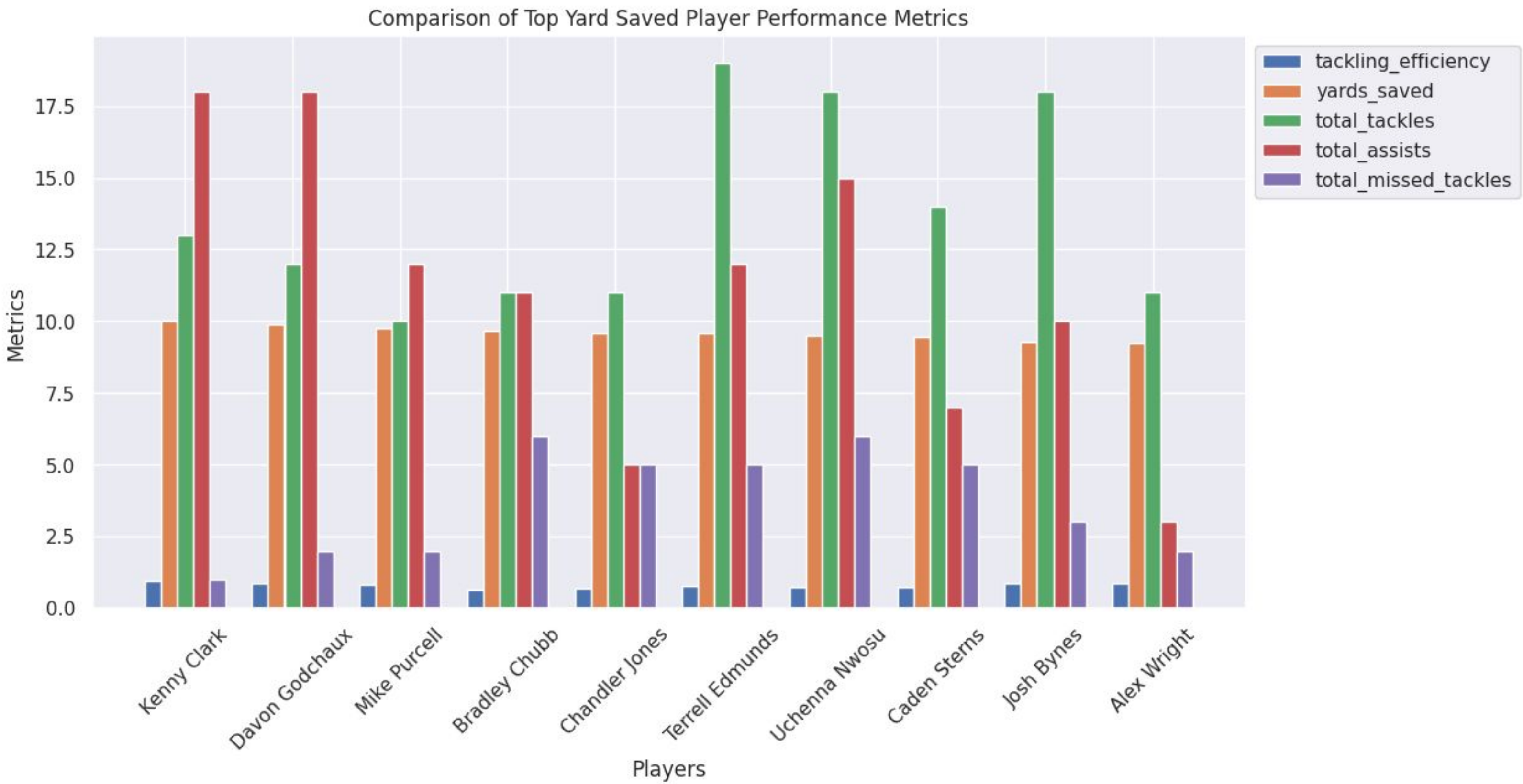
Results



Top Players by Tackle Efficiency
Marshon Lattimore
Leonard Williams
Isaiah Rodgers
Jahlani Tavai
Chris Harris
E.J. Speed
Akeem Davis-Gaither
Tashaun Gipson
Charvarius Ward
Jamel Dean
0
2
4
6
8
Yards Saved
Top Players by Yards Saved
Alex Wright
Josh Bynes
Caden Sterns
Uchenna Nwosu
Terrell Edmunds
Chandler Jones
Bradley Chubb
Mike Purcell
Davon Godchaux
Kenny Clark
0
2
4
6
8
10
Yards Saved
Top Players by Tackle Efficiency
Marshon Lattimore
Leonard Williams
Isaiah Rodgers
Jahlani Tavai
Chris Harris
E.J. Speed
Akeem Davis-Gaither
Tashaun Gipson
Charvarius Ward
Jamel Dean
0.0
0.2
0.4
0.6
0.8
1.0
Tackle Efficiency
Top Players by Yards Saved
Alex Wright
Josh Bynes
Caden Sterns
Uchenna Nwosu
Terrell Edmunds
Chandler Jones
Bradley Chubb
Mike Purcell
Davon Godchaux
Kenny Clark
0.0
0.2
0.4
0.6
0.8
Tackle Efficiency



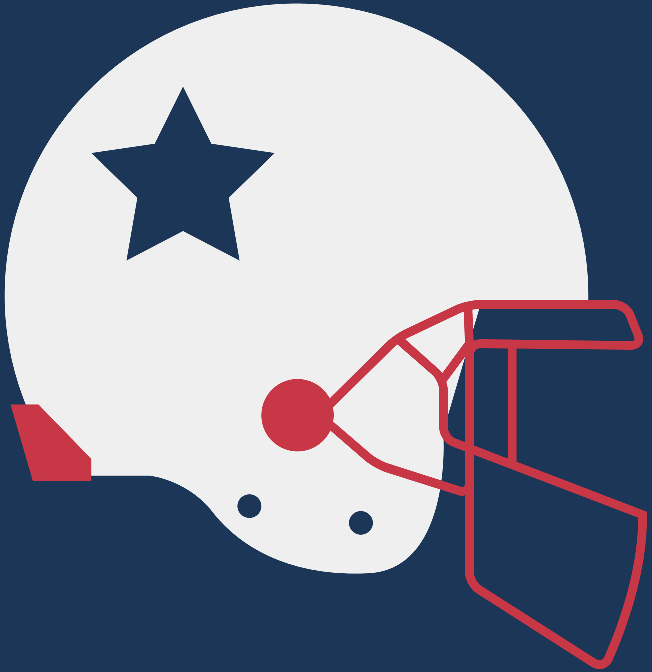
Comparison of Top Efficient Player Performance Metrics
Metrics
Players
tackling_efficiency
yards_saved
total_tackles
total_assists
total_missed_tackles
0
5
10
15
20
25
Jamel Dean
Charvarius Ward
Tashaun Gipson
Akeem Davis-Gaither
E.J. Speed
Chris Harris
Jahlani Tavai
Isaiah Rodgers
Leonard Williams
Marshon Lattimore


Comparison of Top Yard Saved Player Performance Metrics
tackling_efficiency
yards_saved
total_tackles
total_assists
total_missed_tackles
Metrics
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Kenny Clark
Davon Godchaux
Mike Purcell
Bradley Chubb
Chandler Jones
Terrell Edmunds
Uchenna Nwosu
Caden Sterns
Josh Bynes
Alex Wright
Players

Future work

- **More Advanced Metrics:** Evaluating the difficulty of one's tackle (based on their starting distance from the ball carrier, speed, angle, weight difference etc.)
- **Prediction Models:** Based on a given formation/play setup, how likely a certain defensive player can stop the offense from making plays
- **Visualization**: Visualize the detailed play based on the tracking data using heatmap etc.
- **Miscellaneous**: Further explore the ideas we encountered during the data exploration (see appendix)

Thank You!



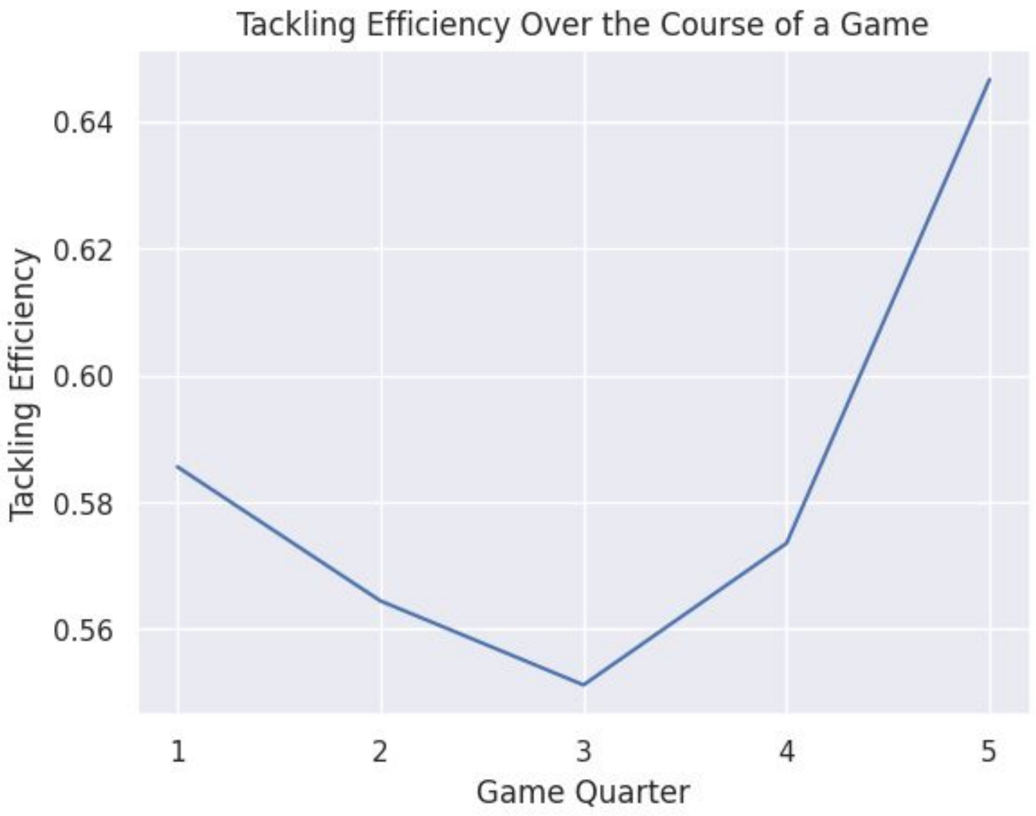
Part 04

Appendix

Appendix A

Speed Distribution During Tackles
Frequency
8000
7000
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
Speed (yards/second)

Appendix B.1

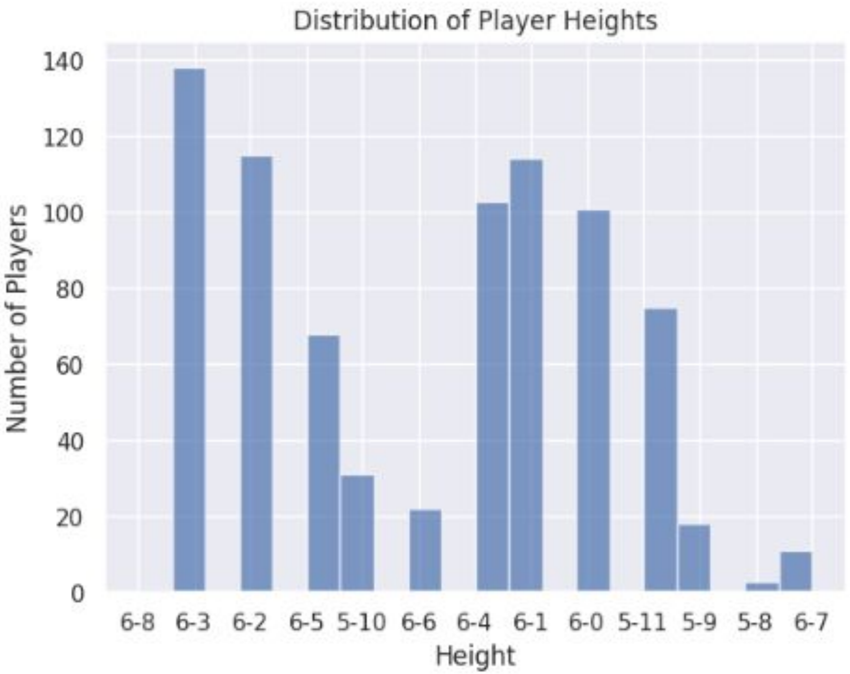

Tackling Efficiency Over the Course of a Game
Tackling Efficiency
0.64
0.62
0.60
0.58
0.56
1
2
3
4
5
Game Quarter

Appendix B.2

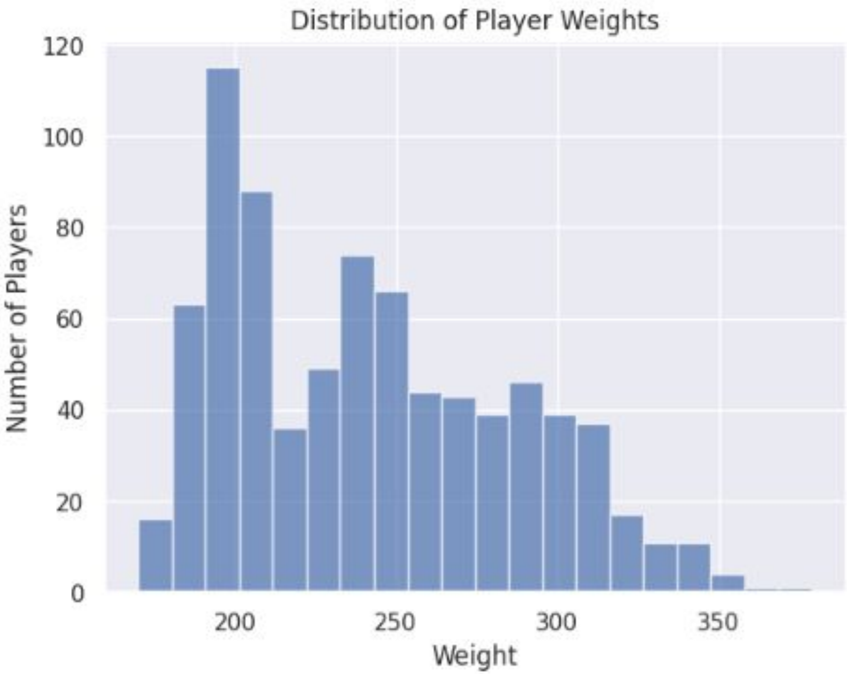
Group by 'playType' and calculate tackling efficiency

	playType	total_attempts	successful_tackles	tackling_efficiency
0	After Pass	7035	4798	0.682018
1	Other	10391	5121	0.492830

Appendix C



Distribution of Player Heights
Number of Players
0
20
40
60
80
100
120
140
6-8
6-3
6-2
6-5
5-10
6-6
6-4
6-1
6-0
5-11
5-9
5-8
6-7
Height



Distribution of Player Weights
Number of Players
0
20
40
60
80
100
120
200
250
300
350
Weight

Appendix D.1

```
games.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 136 entries, 0 to 135
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   gameId             136 non-null    int64
 1   season             136 non-null    int64
 2   week               136 non-null    int64
 3   gameDate           136 non-null    object
 4   gameTimeEastern    136 non-null    object
 5   homeTeamAbbr       136 non-null    object
 6   visitorTeamAbbr    136 non-null    object
 7   homeFinalScore     136 non-null    int64
 8   visitorFinalScore  136 non-null    int64
dtypes: int64(5), object(4)
memory usage: 9.7+ KB
```

```
plays.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12486 entries, 0 to 12485
Data columns (total 35 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   gameId                            12486 non-null  int64
 1   playId                            12486 non-null  int64
 2   ballCarrierId                     12486 non-null  int64
 3   ballCarrierDisplayName            12486 non-null  object
 4   playDescription                   12486 non-null  object
 5   quarter                           12486 non-null  int64
 6   down                              12486 non-null  int64
 7   yardsToGo                         12486 non-null  int64
 8   possessionTeam                    12486 non-null  object
 9   defensiveTeam                     12486 non-null  object
 10  yardlineSide                      12319 non-null  object
 11  yardlineNumber                    12486 non-null  int64
 12  gameClock                         12486 non-null  object
 13  preSnapHomeScore                  12486 non-null  int64
 14  preSnapVisitorScore               12486 non-null  int64
 15  passResult                        6105 non-null   object
 16  passLength                        5634 non-null   float64
 17  penaltyYards                      615 non-null    float64
 18  prePenaltyPlayResult              12486 non-null  int64
 19  playResult                        12486 non-null  int64
 20  playNullifiedByPenalty            12486 non-null  object
 21  absoluteYardlineNumber            12486 non-null  int64
 22  offenseFormation                  12482 non-null  object
 23  defendersInTheBox                 12481 non-null  float64
 24  passProbability                   12149 non-null  float64
 25  preSnapHomeTeamWinProbability     12486 non-null  float64
 26  preSnapVisitorTeamWinProbability  12486 non-null  float64
 27  homeTeamWinProbabilityAdded       12486 non-null  float64
 28  visitorTeamWinProbilityAdded      12486 non-null  float64
 29  expectedPoints                    12486 non-null  float64
 30  expectedPointsAdded               12485 non-null  float64
 31  foulName1                         592 non-null    object
 32  foulName2                         25 non-null     object
 33  foulNFLId1                        592 non-null    float64
 34  foulNFLId2                        25 non-null     float64
dtypes: float64(12), int64(12), object(11)
```

Appendix D.2

```
players.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1683 entries, 0 to 1682
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   nflId        1683 non-null   int64
 1   height       1683 non-null   object
 2   weight       1683 non-null   int64
 3   birthDate    1204 non-null   object
 4   collegeName  1683 non-null   object
 5   position     1683 non-null   object
 6   displayName  1683 non-null   object
dtypes: int64(2), object(5)
memory usage: 92.2+ KB
```

```
tackles.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17426 entries, 0 to 17425
Data columns (total 7 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   gameId            17426 non-null  int64
 1   playId            17426 non-null  int64
 2   nflId             17426 non-null  int64
 3   tackle            17426 non-null  int64
 4   assist            17426 non-null  int64
 5   forcedFumble      17426 non-null  int64
 6   pff_missedTackle  17426 non-null  int64
dtypes: int64(7)
memory usage: 953.1 KB
```

```
weekly_tracking[0].info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1407439 entries, 0 to 1407438
Data columns (total 17 columns):
 #   Column         Non-Null Count    Dtype
---  ------         --------------    -----
 0   gameId         1407439 non-null  int64
 1   playId         1407439 non-null  int64
 2   nflId          1346246 non-null  float64
 3   displayName    1407439 non-null  object
 4   frameId        1407439 non-null  int64
 5   time           1407439 non-null  object
 6   jerseyNumber   1346246 non-null  float64
 7   club           1407439 non-null  object
 8   playDirection  1407439 non-null  object
 9   x              1407439 non-null  float64
 10  y              1407439 non-null  float64
 11  s              1407439 non-null  float64
 12  a              1407439 non-null  float64
 13  dis            1407439 non-null  float64
 14  o              1346397 non-null  float64
 15  dir            1346397 non-null  float64
 16  event          130268 non-null   object
dtypes: float64(9), int64(3), object(5)
memory usage: 182.5+ MB
```