THE SPORTSDATAVERSE: AN OPEN-SOURCE INITIATIVE

CARNEGIE MELLON SPORTS ANALYTICS CONFERENCE 2021
DATA & SOFTWARE CONTRIBUTION, OPEN-TRACK

Saiem Gilani
Lead Engineer
SportsDataverse.org
`saiem.gilani@gmail.com`

1 Introduction

One of the animating forces at the core of a thriving research community is the availability of open data. This streamlines the process to provide standardized datasets and lays the groundwork for reproducible research and creates more accessible opportunities for research and development among the broader data community. Taking note of the sports analytics papers of the last several years and their impact on the research community, one stands out among all others, *nflWAR: A Reproducible Method for Offensive Player Evaluation in Football* [8]. This paper was accompanied with the release of the `nflscrapR` package, which allowed for the reproducibility of the results of the paper. The package was wildly popular, but there was pressure to keep the package and data maintained long after the paper was demonstrated to be reproducible, and eventually the package was replaced with the `nflfastR` package [1]. The combined efforts of Sebastian Carl and the rest of the nflverse team have set a golden standard for operating a set of open-source sports data packages worthy of praise and emulation.

2 The SportsDataverse Initiative

The SportsDataverse is a broader concept to produce a more cohesive set of open-source sports data packages with emphasis on improving testing standards and creating search-able documentation websites. The aim is to have these resources serve as steadfast public utilities **maintained by the community of developers** for research and development. The intention is for projects to build on top of these packages to prototype ideas and, if desired and deemed appropriate, merge relevant portions of their idea to the data processing pipeline and packages.

The SportsDataverse developer group has created a collection of software packages for sports data with modules written in Python, R, and Node.js. Collectively, the SportsDataverse packages cover 18 sports leagues worth of data, including 11+ men's sports and 7+ women's sports with plans for expansion.

Several of the packages written are directly modeled on the data engineering efforts of the `nflfastR` team, including `cfbfastR`[4], `hoopR`[2], and `wehoop`[5]. Python users can use the `sportsdataverse` python package for the same leagues (and more)[3]. These packages allow for the loading of play-by-play and box score data for 15+ seasons of data for NBA, WNBA, men's and women's college basketball, and college football with progress updates. This saves the user countless hours of setting up a web scrape, setting up a cron job to schedule automated data extracts and allows them to spend more time building their project rather than building the data infrastructure and pipeline from scratch.

3 Data and Programming Language Considerations

The programming languages of Python, R and Node.js were chosen for their widespread usage among the data science, sports analytics, and application development communities. Though there is no current intention to extend to more languages, the data repositories made available through the use of the existing packages can be accessed via your preferred language of choice as painlessly as downloading any file from a URL would be. Most data are available in multiple formats of `CSV`, `Rds`, `JSON` or `Parquet` for ease of inter-language use. Among the goals of the SportsDataverse is to flatten the learning curve the average user has to go through to get access to high quality open-source sports data and analytics. While the utility building and function packaging portion of the project is a "work in progress", we have successfully created an exceptional collection of open-source sports data repositories on GitHub. For example, from the packages I directly contribute to, we have generated **over 250 gigabytes of data** in the aforementioned formats for use in loading and reference across the SportsDataverse.

4 Packages

While there are at present a dozen R packages within the SportsDataverse (2 in Python, 3 in Node.js), for the sake of brevity, I will briefly discuss three in this paper – `cfbfastR` for college football, `hoopR` for NBA and men's college basketball, and `wehoop` for WNBA and women's college basketball coverage. Each package has the ability to load 15+ seasons of play-by-play data for each sport covered, from which users can then create their own processing pipeline for their own analyses. Additionally, each package also allows for access to live game data through access to ESPN's content delivery networks (CDN) and application programming interfaces (API).

4.1 Package Installations

The packages listed above have been published on CRAN (the Comprehensive R Archive Network) except for hoopR, which currently must be installed from GitHub until CRAN approves the submission. I prefer the `pacman` installation and library loading method as follows:

```r
if (!requireNamespace('pacman', quietly = TRUE)){install.packages('pacman')}
pacman::p_load_current_gh("saiemgilani/hoopR")
pacman::p_load(cfbfastR, wehoop, dplyr, tictoc, ggplot2, janitor, forcats,
               ggchicklet, paletteer, prismatic, scales)
```

5 `cfbfastR`

5.1 Data Sources

The CollegeFootballData.com API is a resource provided by Bill Radjewski, which allows programmatic access to a database with a number of views for users to query with a free API key[7]. As of `cfbfastR` version 1.5.2, the package exports 67 functions. The bulk (~ 49) of the functions within the package serve as the unofficial R API client for the College Football Data API and those functions' names begin with `cfbd_`. The prefix of exported functions generally indicates the data source of the package, so functions beginning with `espn_` indicate that they are directly getting the data from the ESPN API.

5.2 Play-by-Play with Expected Points and Win Probability (`R/cfbfastR.R`)

```r
tictoc::tic()
pbp <- data.frame()
seasons <- 2014:cfbfastR:::most_recent_season()
progressr::with_progress({
future::plan("multisession")
  cfb_pbp <- cfbfastR::load_cfb_pbp(seasons)
})
tictoc::toc()
glue::glue("College football play-by-play data from {length(unique(cfb_pbp$game_id))} games.")
dplyr::glimpse(cfb_pbp[1:27])
## College football play-by-play data from 6109 games.
# Rows: 1,120,377
# Columns: 27
#$ year                 <dbl> 2014, 2014, 2014, 2014, 2014, 2014, 2014, 201~
#$ week                 <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
#$ id_play              <dbl> 4.005476e+17, 4.005476e+17, 4.005476e+17, 4.0~
#$ game_id              <int> 400547640, 400547640, 400547640, 400547640, 4~
#$ game_play_number     <dbl> 1, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 11, 12,~
#$ half_play_number   <dbl> 1, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 11, 12,~
#$ drive_play_number  <dbl> 1, 1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 3, 4, 1, ~
#$ pos_team           <chr> "Temple", "Temple", "Temple", "Temple", "Temp~
#$ def_pos_team       <chr> "Vanderbilt", "Vanderbilt", "Vanderbilt", "Va~
#$ pos_team_score     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
#$ def_pos_team_score <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
#$ half               <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
#$ period             <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
#$ clock.minutes      <int> 14, 14, 14, 14, 13, 13, 12, 12, 12, 12, 11, 1~
#$ clock.seconds      <int> 55, 55, 45, 20, 50, 25, 58, 50, 7, 0, 20, 40,~
#$ play_type          <chr> "Kickoff Return (Offense)", "Penalty", "Pass ~
#$ play_text          <chr> "Hayden Lekacz kickoff for 64 yds , Khalif He~
#$ down               <dbl> 1, 1, 1, 2, 3, 1, 2, 3, 4, 1, 2, 3, 4, 4, 1, ~
#$ distance           <dbl> 10, 10, 15, 12, 7, 10, 8, 8, 3, 10, 7, 3, 7, ~
#$ yards_to_goal      <dbl> 65, 81, 86, 83, 78, 70, 68, 68, 63, 72, 69, 6~
#$ yards_gained       <dbl> 18, -5, 3, 5, 8, 2, 0, 5, 8, 3, 4, -4, 0, 0, ~
#$ EPA                <dbl> -0.565383839, -0.469578030, -0.484324992, -0.~
#$ ep_before          <dbl> 0.8358481, 0.2704643, -0.4884171, -0.9727421,~
#$ ep_after           <dbl> 0.2704643, -0.1991138, -0.9727421, -1.1872586~
#$ wpa                <dbl> -0.0224244, -0.0271206, -0.0118346, -0.000090~
#$ wp_before          <dbl> 0.4919244, 0.4695000, 0.4423794, 0.4305448, 0~
#$ wp_after           <dbl> 0.4695000, 0.4423794, 0.4305448, 0.4304547, 0~
```

6 hoopR

6.1 Data Sources

As of hoopR version 1.3, the package exports 191 functions. The bulk (∼ 132) of the functions within the package serve as a NBA Stats API wrapper and those functions' names begin with nba_ and another 33 functions cover scraping KenPom's college basketball statistics website. The prefix of exported functions generally indicates the data source of the package, so functions beginning with espn_ indicate that they are directly getting the data from the ESPN API, including those capable of use during live game-play for both men's college basketball and the NBA.

6.2 NBA Shot Clock Chart (R/hoopR.R)

If you never perused the amazing Owen Phillips's blog, The F5[6], the following example will give you a taste of the NBA content you missed out on by adapting one of his tutorials to use the hoopR package.

6.2.1 Using hoopR's NBA stats API functions

```
source("R/hoopR_utils.R")
pacman::p_load(dplyr, ggplot2, janitor, forcats,ggchicklet, paletteer, prismatic, scales)
# Get and filter data to top 30 players in FG3A ----
fg3a_leaders <- hoopR::nba_leaguedashplayerstats(season="2020-21")$LeagueDashPlayerStats
fg3a_leaders <- fg3a_leaders %>%
  janitor::clean_names() %>%
  dplyr::mutate(fg3a=as.numeric(.data$fg3a)) %>%
  dplyr::arrange(desc(.data$fg3a)) %>%
  slice(1:30) %>%
  dplyr::select(.data$player_name, .data$player_id, .data$fg3a)
# Get player shot-clock range stats-----
shot_clock_opts <- c("24-22", "22-18+Very+Early", "18-15+Early",
                     "15-7+Average", "7-4+Late", "4-0+Very+Late")
```

```
shot_clock_range_df <- purrr::map_df(shot_clock_opts, function(.x){
  return(data.frame(shot_clock_range=.x,
    hoopR::nba_leaguedashplayerptshot(shot_clock_range=.x)$LeagueDashPTShots))
})
shot_clock_range_df <- shot_clock_range_df %>%
  janitor::clean_names() %>%
  dplyr::select(player_id, player_name, fg3m, fg3a, shot_clock_range)
```

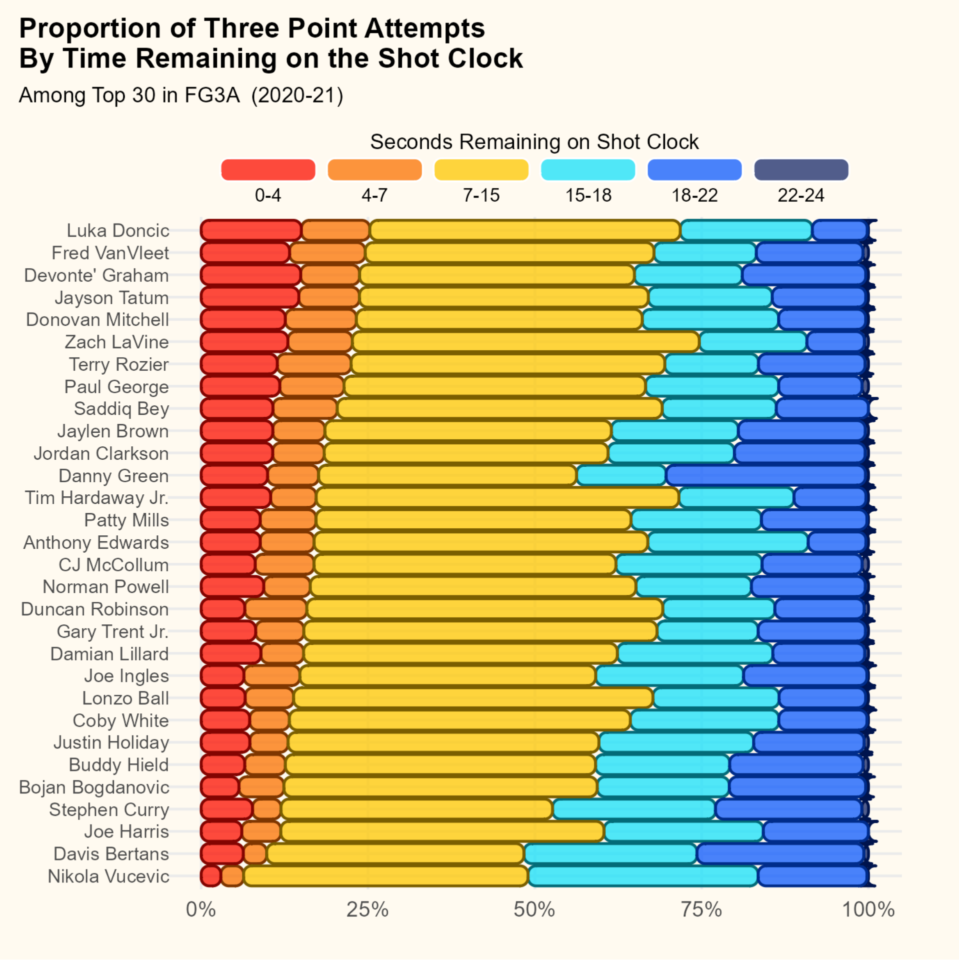
6.2.2 Data Prep for Plotting

```
## Clean the shot_clock_range variable ----
shot_clock_range_df <- clean_shot_clock_range(shot_clock_range_df)
## Create shot clock range FG3A frequency ----
shot_clock_range_df <- shot_clock_range_df %>%
  dplyr::group_by(.data$player_name, .data$player_id) %>%
  dplyr::mutate(shotclock_freq = .data$fg3a / sum(.data$fg3a)) %>% dplyr::ungroup()
## Create combined text labels for the ranges -----
shot_clock_range_df <- shot_clock_range_df %>%
  dplyr::mutate(
    shotclock_cat = dplyr::case_when(
      .data$shot_clock_range %in% c("0-4", "4-7") ~ "Late",
      .data$shot_clock_range %in% c("15-18", "7-15") ~ "Average",
      .data$shot_clock_range %in% c("22-24", "18-22") ~ "Early",
      TRUE ~ NA_character_)) %>%
  dplyr::group_by(.data$player_name, .data$shotclock_cat) %>%
  dplyr::mutate(sum_shotclock_cat = sum(.data$shotclock_freq)) %>%
  dplyr::ungroup()
#### Convert overall shot clock category to a factor ----
shot_clock_range_df$shotclock_cat <- as.factor(shot_clock_range_df$shotclock_cat)
shot_clock_range_df$shotclock_cat <- factor(shot_clock_range_df$shotclock_cat,
                                            levels = c("Late", "Average", "Early"))
### Order players by the proporition of Late 3s----
shot_clock_range_df <- shot_clock_range_df %>%
  dplyr::arrange(.data$shotclock_cat, .data$sum_shotclock_cat) %>%
  dplyr::mutate(player_name = factor(.data$player_name, unique(.data$player_name)))
```

6.2.3 Plot the Data!

```
shot_clock_range_df %>%
  ggplot(aes(player_name, shotclock_freq)) +
  geom_chicklet(aes(fill = fct_rev(shot_clock_range),
    color = after_scale(clr_darken(fill, 0.5))), alpha = .75) +
  scale_y_continuous(position = "left",
    labels = c("0%", "25%", "50%", "75%", "100%"), limits = c(0, 1)) +
  coord_flip() +
  scale_fill_manual(values = c(paletteer_d("fishualize::Hypsypops_rubicundus", direction = -1),
  "#172869FF")) +
  guides(fill=guide_legend(keywidth= .5, keyheight= .2, default.unit="inch",
    title.position = 'top', label.position = 'bottom',  nrow = 1) ) +
  theme_minimal(base_size = 10) +
  theme(legend.position = 'top',
        axis.title.y = element_blank(), axis.title.x = element_blank(),
        legend.text = element_text(size = 7, vjust = 4),
        legend.title = element_text(size = 8, hjust = .5, vjust = -2),
        legend.margin=margin(0,0,0,0), legend.box.margin=margin(0,0,-10,0),
        panel.grid.minor = element_blank(),
        plot.background = element_rect(fill = 'floralwhite', color = "floralwhite"),
        plot.title = element_text(face = 'bold', size = 10.5),
        plot.subtitle = element_text(size = 8),
        plot.title.position = "plot",   plot.margin = unit(c(.5, 1.5, 1, .5), "lines"),
        axis.text.y = element_text(margin=margin(0,-3,0,0), size = 6)) +
  labs(title = "Proportion Of Three Point Attempts By Time Remaining On The Shot Clock",
       subtitle ="Among Top 30 In FG3A  (2020-21)",
       fill = "Seconds Remaining On Shot Clock")
```

Figure 1: Attempts By Time Remaining On The Shot Clock: Top 30 Three Point FGA



7 wehoop

7.1 Data Sources

As of `wehoop` version 1.1, the package exports 27 functions. The prefix of exported functions generally indicates the data source of the package, so functions beginning with `espn_` indicate that they are directly getting the data from the ESPN API, including those capable of use during live game-play for both women's college basketball and the WNBA.

7.2 WNBA full play-by-play seasons (2002-2021) 1-2 minutes (R/wehoop.R)

```
tictoc::tic()
progressr::with_progress({
  wnba_pbp <- wehoop::load_wnba_pbp(2002:2021)
})
tictoc::toc()
## 12.01 sec elapsed
glue::glue("WNBA play-by-play data from {length(unique(wnba_pbp$game_id))} games.")
## WNBA play-by-play data from 4674 games.
# Rows: 1,782,985
# Columns: 42
#$ shooting_play             <lgl> FALSE, FALSE, TRUE, FALSE, TRUE, ~
#$ sequence_number           <chr> "1", "2", "3", "4", "5", "6", "7"~
#$ period_display_value      <chr> "1st Quarter", "1st Quarter", "1s~
#$ period_number             <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
#$ home_score                <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
#$ coordinate_x              <int> 0, 0, 35, 0, 40, 0, 0, 25, 0, 0, ~
#$ coordinate_y              <int> 0, 0, 11, 0, 5, 0, 0, 0, 0, 0, 0,~
#$ scoring_play              <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
#$ clock_display_value       <chr> "20:00", "20:00", "19:34", "19:31~
#$ team_id                   <chr> NA, "6", "6", "9", "9", "6", "6",~
#$ type_id                   <chr> "411", "615", "20558", "587", "20~
#$ type_text                 <chr> "Start Period", "Jumpball", "Jump~
#$ away_score                <int> 0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 3, ~
#$ id                        <dbl> 22052500600000, 22052500600001, 2~
#$ text                      <chr> "Start of the 1st Half.", "Jumpba~
#$ score_value               <int> 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 1, ~
#$ participants_0_athlete_id <chr> NA, "18", "7", "21", "18", "6", "~
#$ participants_1_athlete_id <chr> NA, "6", NA, NA, NA, NA, "6", "20~
#$ participants_2_athlete_id <chr> NA, "9", NA, NA, NA, NA, NA, NA, ~
#$ type_abbreviation         <lgl> NA, NA, NA, NA, NA, NA, NA, NA, N~
#$ season                    <int> 2002, 2002, 2002, 2002, 2002, 200~
#$ season_type               <int> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, ~
#$ away_team_id              <int> 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, ~
#$ away_team_name            <chr> "New York", "New York", "New York~
#$ away_team_mascot          <chr> "Liberty", "Liberty", "Liberty", ~
#$ away_team_abbrev          <chr> "NYL", "NYL", "NYL", "NYL", "NYL"~
#$ away_team_name_alt        <chr> "New York", "New York", "New York~
#$ home_team_id              <int> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, ~
#$ home_team_name            <chr> "Los Angeles", "Los Angeles", "Lo~
#$ home_team_mascot          <chr> "Sparks", "Sparks", "Sparks", "Sp~
#$ home_team_abbrev          <chr> "LOS", "LOS", "LOS", "LOS", "LOS"~
#$ home_team_name_alt        <chr> "Los Angeles", "Los Angeles", "Lo~
#$ home_team_spread          <dbl> 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5~
#$ game_spread               <dbl> 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5~
#$ home_favorite             <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRU~
#$ clock_minutes             <chr> "20", "20", "19", "19", "19", "19~
#$ clock_seconds             <chr> "00", "00", "34", "31", "21", "19~
#$ half                      <chr> "1", "1", "1", "1", "1", "1", "1"~
#$ lag_half                  <chr> NA, "1", "1", "1", "1", "1", "1",~
#$ lead_half                 <chr> "1", "1", "1", "1", "1", "1", "1"~
#$ game_play_number          <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11~
#$ game_id                   <int> 220525006, 220525006, 220525006, ~
```

7.3 WNBA full team box score seasons (2003-2021) 5-30 seconds

```
tictoc::tic()
progressr::with_progress({
  wnba_team_box <- wehoop::load_wnba_team_box(2003:2021)
})
tictoc::toc()
## 8.06 sec elapsed
glue::glue("WNBA team boxscore data from {length(unique(wnba_team_box\$game_id))} games.")
## WNBA team boxscore data from 4312 games.
```

7.4 WNBA full player box score seasons (2002-2021) 5-30 seconds

```
tictoc::tic()
progressr::with_progress({
  wnba_player_box <- wehoop::load_wnba_player_box(2002:2021)
})
tictoc::toc()
## 6.68 sec elapsed
glue::glue("WNBA player boxscore data from {length(unique(wnba_player_box\$game_id))} games.")
## WNBA player boxscore data from 4677 games.
```

7.5 Women's college basketball full play-by-play seasons (2004-2021) 45-90 seconds

```
tictoc::tic()
progressr::with_progress({
  wbb_pbp <- wehoop::load_wbb_pbp(2004:2021)
})
tictoc::toc()
## 44.51 sec elapsed
glue::glue("WBB play-by-play data from {length(unique(wbb_pbp$game_id))} games.")
## WBB play-by-play data from 26023 games.
dplyr::glimpse(wbb_pbp[1:53])
# Rows: 8,650,487
# Columns: 53
# $ shooting_play                   <lgl> TRUE, FALSE, FALSE, TRUE, FALSE, ~
# $ sequence_number                 <chr> "1", "2", "3", "4", "5", "6", "7"~
# $ period_display_value            <chr> "1st Half", "1st Half", "1st Half~
# $ period_number                   <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
# $ home_score                      <int> 0, 0, 0, 0, 0, 2, 2, 3, 5, 5, 5, ~
# $ scoring_play                    <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
# $ clock_display_value             <chr> "19:51", "19:51", "19:33", "19:18"~
# $ team_id                         <chr> "228", "150", "150", "228", "150"~
# $ type_id                         <chr> "20558", "587", "601", "20558", "~
# $ type_text                       <chr> "JumpShot", "Defensive Rebound", ~
# $ away_score                      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
# $ id                              <dbl> 2.405802e+12, 2.405802e+12, 2.405~
# $ text                            <chr> "Lakeia Stokes missed Two Point J~
# $ score_value                     <int> 0, 0, 0, 0, 0, 2, 0, 1, 2, 0, 0, ~
# $ participants_0_athlete_id       <chr> "285", "436", "2475", "282", "247~
# $ participants_1_athlete_id       <chr> NA, NA, NA, NA, NA, NA, NA, NA, N~
# $ season                          <int> 2004, 2004, 2004, 2004, 2004, 200~
# $ season_type                     <int> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, ~
# $ away_team_id                    <int> 228, 228, 228, 228, 228, 228, 228~
# $ away_team_name                  <chr> "Clemson", "Clemson", "Clemson", ~
# $ away_team_mascot                <chr> "Tigers", "Tigers", "Tigers", "Ti~
# $ away_team_abbrev                <chr> "CLEM", "CLEM", "CLEM", "CLEM", "~
# $ away_team_name_alt              <chr> "Clemson", "Clemson", "Clemson", ~
# $ home_team_id                    <int> 150, 150, 150, 150, 150, 150, 150~
# $ home_team_name                  <chr> "Duke", "Duke", "Duke", "Duke", "~
# $ home_team_mascot                <chr> "Blue Devils", "Blue Devils", "Bl~
# $ home_team_abbrev                <chr> "DUKE", "DUKE", "DUKE", "DUKE", "~
# $ home_team_name_alt              <chr> "Duke", "Duke", "Duke", "Duke", "~
# $ home_team_spread                <dbl> 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5~
# $ game_spread                     <dbl> 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5~
# $ home_favorite                   <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRU~
# $ game_spread_available           <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
# $ game_id                         <int> 240580150, 240580150, 240580150, ~
# $ qtr                             <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
# $ time                            <chr> "19:51", "19:51", "19:33", "19:18"~
# $ clock_minutes                   <chr> "19", "19", "19", "19", "19", "19"~
# $ clock_seconds                   <chr> "51", "51", "33", "18", "18", "13"~
# $ half                            <chr> "1", "1", "1", "1", "1", "1", "1"~
# $ game_half                       <chr> "1", "1", "1", "1", "1", "1", "1"~
# $ lag_qtr                         <dbl> NA, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
# $ lead_qtr                        <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
# $ lag_game_half                   <chr> NA, "1", "1", "1", "1", "1", "1",~
# $ lead_game_half                  <chr> "1", "1", "1", "1", "1", "1", "1"~
# $ start_quarter_seconds_remaining <int> 1191, 1191, 1173, 1158, 1158, 115~
# $ start_half_seconds_remaining    <int> 1791, 1791, 1773, 1758, 1758, 175~
# $ start_game_seconds_remaining    <int> 2991, 2991, 2973, 2958, 2958, 295~
# $ game_play_number                <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11~
# $ end_quarter_seconds_remaining   <dbl> 600, 1191, 1191, 1173, 1158, 1158~
# $ end_half_seconds_remaining      <dbl> 1200, 1791, 1791, 1773, 1758, 175~
# $ end_game_seconds_remaining      <dbl> 2400, 2991, 2991, 2973, 2958, 295~
# $ period                          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
# $ coordinate_x                    <dbl> NA, NA, NA, NA, NA, NA, NA, NA, N~
# $ coordinate_y                    <dbl> NA, NA, NA, NA, NA, NA, NA, NA, N~
```

7.6 Women’s college basketball full team box score seasons (2006-2021) 5-30 seconds

```
tictoc::tic()
progressr::with_progress({
  wbb_team_box <- wehoop::load_wbb_team_box(2006:2021)
})
tictoc::toc()
## 7.12 sec elapsed
glue::glue("WBB team boxscore data from {length(unique(wbb_team_box$game_id))} games.")
## WBB team boxscore data from 20762 games.
```

7.7 Women’s college basketball full player box score seasons (2006-2021) 5-30 seconds

```
tictoc::tic()
progressr::with_progress({
  wbb_player_box <- wehoop::load_wbb_player_box(2006:2021)
})
tictoc::toc()
## 4.36 sec elapsed
glue::glue("WBB player boxscore data from {length(unique(wbb_player_box$game_id))} games.")
## WBB player boxscore data from 20906 games.
```

8 What next?

The first phase of the initiative is to broaden the number of sports the packages covered by the packages through existing data sources, to improve the quality of the existing packages, and to recruit existing open-source sports data packages and their developers to the organization.

The end goal of the initiative would allow the interested to get started in sports analytics with a command as simple as:

```
# Python (already possible, go try it)
pip install sportsdataverse
# R (Work in progress)
install.packages(sportsdataverse)
# Node.js (already possible, go try it)
yarn add sportsdataverse
```

For a thorough covering of the documentation for each of the packages and their functionalities, please refer to Table 1 for the Python packages, Table 2 for the R packages, and Table 3 for the Node.js modules.

9 Package Documentation Reference

These are the packages that have agreed to join us the SportsDataverse:

Python Packages in the SportsDataverse

Package	Sports Leagues	Repository	Author(s)
sportsdataverse	College Football, NFL, NBA, WNBA, NHL, Men's and Women's College Basketball, most collegiate sports	GitHub - Docs	Saiem Gilani

Table 1: Python packages contributed to the SportsDataverse

R Packages in the SportsDataverse

Package	Sports Leagues	Repository	Author(s)
cfbfastR	College Football	GitHub - Docs	Saiem Gilani, Akshay Easwaran, Jared Lee, Eric Hess
hoopR	NBA and Men's College Basketball	GitHub - Docs	Saiem Gilani
wehoop	WNBA and Women's College Basketball	GitHub - Docs	Saiem Gilani and Geoff Hutchinson
gamezoneR	Men's and Women's College Basketball	GitHub - Docs	Jack Lichtenstein
worldfootballR	EPL, La Liga, Bundesliga, Serie A, Ligue 1, RFPL	GitHub - Docs	Jason Zivkovic
baseballr	MLB, MiLB, NCAA Baseball	GitHub - Docs	Bill Petti
hockeyR	NHL	GitHub - Docs	Dan Morse
fastRhockey	Premier Hockey Federation	GitHub - Docs	Ben Howell
powerplay	NHL	GitHub - Docs	Saiem Gilani
cfbplotR	College Sports visualizations	GitHub - Docs	Jared Lee
cfb4th	College Football Modeling	GitHub - Docs	Jared Lee
puntr	NFL and College Football	GitHub - Docs	Dennis Brookner and Raphael LadenGuindon
recruitR	Men's College Sports Recruiting	GitHub - Docs	Saiem Gilani

Table 2: R Packages contributed to the SportsDataverse

Node.js Packages in the SportsDataverse

Package	Sports Leagues	Repository	Author(s)
sportsdataverse	College Football, NFL, NBA, WNBA, NHL, Men's and Women's College Basketball, most collegiate sports	GitHub - Docs	Saiem Gilani
nfl-nerd	NFL	GitHub	Annie Tran

Table 3: Node.js modules contributed to the SportsDataverse

10 References

[1] Sebastian Carl and Ben Baldwin. *nflfastR: Functions to Efficiently Access NFL Play by Play Data*. R package version 4.2.0. 2021. URL: https://CRAN.R-project.org/package=nflfastR (cit. on p. 1).

[2] Saiem Gilani. *hoopR: The SportsDataverse's R Package for Men's Basketball Data.* 2021. URL: https://hoopR.sportsdataverse.org/ (cit. on p. 1).

[3] Saiem Gilani. *sportsdataverse-py: The SportsDataverse's Python Package for Sports Data.* 2021. URL: https://py.sportsdataverse.org (cit. on p. 1).

[4] Saiem Gilani, Akshay Easwaran, Jared Lee, and Eric Hess. *cfbfastR: The SportsDataverse's R Package for College Football Data.* 2021. URL: https://saiemgilani.github.io/cfbfastR/ (cit. on p. 1).

[5] Saiem Gilani and Geoff Hutchinson. *wehoop: The SportsDataverse's R Package for Women's Basketball Data.* 2021. URL: https://wehoop.sportsdataverse.org (cit. on p. 1).

[6] Owen Phillips. *How to: Get data from Stats.nba.com*. Feb. 2021. URL: https://thef5.substack.com/p/nba-shotclock-chart (cit. on p. 3).

[7] Bill Radjewski. *College Football Data*. 2016. URL: https://www.collegefootballdata.com/ (cit. on p. 2).

[8] Ronald Yurko, Samuel Ventura, and Maksim Horowitz. "nflWAR: a reproducible method for offensive player evaluation in football". In: *Journal of Quantitative Analysis in Sports* 15.3 (2019), pp. 163–183. DOI: doi:10.1515/jqas-2018-0010. URL: https://doi.org/10.1515/jqas-2018-0010 (cit. on p. 1).