

# Package: ffopportunity (via r-universe)

September 22, 2026

**Title** Models for Fantasy Football Expected Points

**Version** 0.1.2

**Description** Downloads expected fantasy points data from 'ffverse' repositories if available, and otherwise builds up expected points data by applying models to 'nflverse' play-by-play data.

**License** GPL (>= 3)

**URL** <https://ffopportunity.ffverse.com>,  
<https://github.com/ffverse/ffopportunity>

**BugReports** <https://github.com/ffverse/ffopportunity/issues>

**Depends** R (>= 3.6.0)

**Imports** cli (>= 3.0.0), data.table, dplyr (>= 1.0.0), glue (>= 1.0.0),  
hardhat (>= 0.1.0), janitor (>= 1.0.0), magrittr (>= 1.0.0),  
nflreadr (>= 1.3.0), purrr (>= 0.3.0), rappdirs (>= 0.3.0),  
recipes (>= 0.1.16), rlang (>= 0.1.0), stats, stringr (>= 1.4.0),  
tibble (>= 3.0.0), tidyr (>= 1.0.0), tidyselect (>= 1.0.0),  
utils, xgboost (>= 1.1)

**Suggests** arrow (>= 5.0.0), covr, curl, jsonlite, piggyback (>= 0.1.1),  
progressr (>= 0.8.0), roxygen2 (>= 7.1.0), testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**Config/pak/sysreqs** make libicu-dev libssl-dev

**Repository** <https://ffverse.r-universe.dev>

**Date/Publication** 2025-12-11 15:26:57 UTC

**RemoteUrl** <https://github.com/ffverse/ffopportunity>

**RemoteRef** main

**RemoteSha** 74dcb35a112a71e5349b36abe940316067ec4fec

Contents

|                           |          |
|---------------------------|----------|
| ep_build . . . . .        | 2        |
| ep_cache_models . . . . . | 3        |
| ep_load . . . . .         | 3        |
| ep_predict . . . . .      | 4        |
| ep_preprocess . . . . .   | 5        |
| ep_summarize . . . . .    | 6        |
| <b>Index</b>              | <b>7</b> |

---

|          |                 |
|----------|-----------------|
| ep_build | <i>Build EP</i> |
|----------|-----------------|

---

Description

This function builds Expected Fantasy Points predictions by downloading the xgboost models and play-by-play data, applying the model, and summarizing to player level.

Usage

```
ep_build(season = nflreadr::most_recent_season(), version = "latest")
```

Arguments

- season            a numeric vector of seasons that defaults to most recent season. Must be later than 2006.
- version          an EP model version - one of "latest" (default) or "v1.0.0" (these are currently identical)

Value

a list containing three dataframes: ep\_weekly provides a game-level summary by player, ep\_pbp\_pass provides EP data on pass plays, and ep\_pbp\_rush provides EP data on rush plays.

See Also

Other main: [ep\\_load\(\)](#)

Examples

```
try({ # prevents cran-related errors
  ep_build(season = 2021)
})
```

---

|                 |                         |
|-----------------|-------------------------|
| ep_cache_models | <i>Model versioning</i> |
|-----------------|-------------------------|

---

**Description**

This function checks the cache for a previously downloaded model and then (optionally) tries to download the model from GitHub release.

**Usage**

```
ep_cache_models(  
  version = c("latest", "v1.0.0"),  
  force = FALSE,  
  ask = interactive()  
)
```

**Arguments**

|         |                                                                       |
|---------|-----------------------------------------------------------------------|
| version | one of "latest" or "v1.0.0" - currently these refer to the same thing |
| force   | TRUE or FALSE - forces download regardless of currently existing      |
| ask     | TRUE or FALSE - ask before downloading - force will skip this.        |

**Value**

a status message after attempting to download the model.

---

|         |                                  |
|---------|----------------------------------|
| ep_load | <i>Load Expected Points data</i> |
|---------|----------------------------------|

---

**Description**

This function downloads precomputed expected points data from the ffoportunity automated releases.

**Usage**

```
ep_load(  
  season = nflreadr::most_recent_season(),  
  type = c("weekly", "pbp_pass", "pbp_rush"),  
  version = c("latest", "v1.0.0")  
)
```

**Arguments**

|         |                                                                                                     |
|---------|-----------------------------------------------------------------------------------------------------|
| season  | A numeric vector of four digit years associated with given NFL seasons - defaults to latest season. |
| type    | Data type - one of "weekly", "pbp_pass", or "pbp_rush"                                              |
| version | EP model version: one of "latest" (default) or "v1.0.0" - these are currently identical.            |

**Value**

a dataframe identical to what would be returned by `ffoportunity::ep_build()` for a given season.

**See Also**

Other main: [ep\\_build\(\)](#)

**Examples**

```
try({
  ep_load() %>% head(10)
  ep_load(2020:2021) %>% head(10)
  ep_load(2021, type = "pbp_pass") %>% head(10)
  ep_load(2006, type = "pbp_rush", version = "v1.0.0") %>% head(10)
})
```

---

ep\_predict

---

*Predict EP*


---

**Description**

This function runs the prediction functions over preprocessed data.

**Usage**

```
ep_predict(preprocessed_pbp, version = c("latest", "v1.0.0"))
```

**Arguments**

|                  |                                                                                           |
|------------------|-------------------------------------------------------------------------------------------|
| preprocessed_pbp | list with dataframes created by <code>ep_preprocess</code>                                |
| version          | ep model version - available is "latest" or "v1.0.0" (these are currently the same thing) |

**Value**

a dataframe with the expected fields added

## Examples

```
try({  
  preprocessed <- readRDS(system.file("ep_preprocessed.rds", package = "ffopportunity"))  
  # this file is equivalent to nflreadr::load_pbp(2021) %>% head(1000) %>% ep_preprocess()  
  ep_predict(preprocessed)  
})
```

---

ep\_preprocess

*Preprocess Data*

---

## Description

This function performs pre-processing steps to make expected points predictions on nflreadr data

## Usage

```
ep_preprocess(pbp)
```

## Arguments

pbp                      pbp dataframe from nflreadr::load\_pbp()

## Value

a list of two dataframes (one for passes and one for rushes) of nflreadr data with the expectedpoints columns transformed for prediction

## See Also

vignette("basic") for example usage

## Examples

```
try({ # catch failures for CRAN purposes  
  pbp_download <- readRDS(system.file("pbp_download.rds", package = "ffopportunity"))  
  # this file is equivalent to nflreadr::load_pbp(2021) %>% head(1000)  
  ep_preprocess(pbp_download)  
})
```

---

ep\_summarize

*Summarize EP*


---

### Description

This function summarizes the EP data up to the game level

### Usage

```
ep_summarize(
  predicted_pbp,
  stat_type = c("all", "expected_points", "team_stats")
)
```

### Arguments

**predicted\_pbp** list with dataframes created by `ep_predict`

**stat\_type** options to limit the columns returned by `ep_summarize` - available options are "all", "expected\_points", and "team stats"

### Value

a dataframe with the expected points fields added

### See Also

`vignette("basic")` for example usage

### Examples

```
try({
  predicted <- readRDS(system.file("ep_predicted.rds", package = "ffoportunity"))
  # equivalent to nflreadr::load_pbp(2021) %>% head(100) %>% ep_preprocess() %>% ep_predict()
  ep_summarize(predicted)
})
```

# Index

## \* **main**

ep\_build, [2](#)

ep\_load, [3](#)

ep\_build, [2](#), [4](#)

ep\_cache\_models, [3](#)

ep\_load, [2](#), [3](#)

ep\_predict, [4](#)

ep\_preprocess, [5](#)

ep\_summarize, [6](#)