

# Package ‘iimi’

July 26, 2024

**Title** Identifying Infection with Machine Intelligence

**Version** 1.1.1

**Description** A novel machine learning method for plant viruses diagnostic using genome sequencing data. This package includes three different machine learning models, random forest, XGBoost, and elastic net, to train and predict mapped genome samples. Mappability profile and unreliable regions are introduced to the algorithm, and users can build a mappability profile from scratch with functions included in the package. Plotting mapped sample coverage information is provided.

**Encoding** UTF-8

**RoxygenNote** 7.3.1

**LazyData** true

**VignetteBuilder** knitr

**Imports** GenomicAlignments, IRanges, Rsamtools, data.table, mltools, randomForest, xgboost, Biostrings, stats, MTPS, R.utils, caret, stringr, dplyr, Rdpack

**RdMacros** Rdpack

**Depends** R (>= 3.5.0)

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown, httr

**Config/testthat/edition** 3

**License** MIT + file LICENSE

**LazyDataCompression** xz

**NeedsCompilation** no

**Author** Haochen Ning [aut],  
Ian Boyes [aut],  
Ibrahim Numanagić [aut] (<<https://orcid.org/0000-0002-2970-7937>>),  
Michael Rott [aut],  
Li Xing [aut] (<<https://orcid.org/0000-0002-4186-7909>>),  
Xuekui Zhang [aut, cre] (<<https://orcid.org/0000-0003-4728-2343>>)

**Maintainer** Xuekui Zhang <xuekui@uvic.ca>

**Repository** CRAN

**Date/Publication** 2024-07-26 08:20:02 UTC

## Contents

convert_bam_to_rle . . . . .	2
convert_rle_to_df . . . . .	3
create_high_nucleotide_content . . . . .	4
create_mappability_profile . . . . .	4
example_cov . . . . .	5
example_diag . . . . .	6
nucleotide_info . . . . .	6
plot_cov . . . . .	7
predict_iimi . . . . .	8
trained_en . . . . .	9
trained_rf . . . . .	9
trained_xgb . . . . .	9
train_iimi . . . . .	10
unreliable_regions . . . . .	11
virus_segments . . . . .	11

<b>Index</b>	<b>13</b>
--------------	-----------

---

convert_bam_to_rle	<i>convert_bam_to_rle</i>
--------------------	---------------------------

---

### Description

Converts one or more indexed and sorted BAM files into a run-length encodings (RLEs) list.

### Usage

```
convert_bam_to_rle(bam_file, paired = FALSE)
```

### Arguments

bam_file	path to BAM file(s).
paired	Indicate if the sequencing paired is single-end or paired-end reads. TRUE if paired-end. FALSE if single-end.

### Value

A list of coverage profile(s) in RLE format with one or more samples.

### Examples

```
## Not run:
## Please change the path to your folder where you
## store sorted and indexed BAM files of mapped samples

rles <- convert_bam_to_rle("path/to/bam/file")
```

```
## End(Not run)
```

---

```
convert_rle_to_df      convert_rle_to_df
```

---

## Description

Converts a list of run-length encodings (RLEs) into a data frame with 16 features after mappability profiling and nucleotide filtering.

## Usage

```
convert_rle_to_df(  
  covs,  
  unreliable_region_df = unreliable_regions,  
  unreliable_region_enabled = TRUE,  
  additional_nucleotide_info = data.frame()  
)
```

## Arguments

**covs** A list of Coverage profile(s) in RLE format. Can be one or more samples.

**unreliable\_region\_df** The unreliable regions of the virus segments. Default is `unreliable_regions`. It includes the mappability profile from a host genome (we only have *Arabidopsis thaliana* right now) and virus references, and the regions that have CG% and A% over 60% and 45% respectively.

**unreliable\_region\_enabled** Default is TRUE. If TRUE, the input will be checked against `unreliable_region_df`. If FALSE, this step will be skipped.

**additional\_nucleotide\_info** Additional nucleotide information for virus segments that are not included in `nucleotide_info`. The information provided must be a data frame that follows the format of `nucleotide_info`. Default is an empty data frame.

## Value

A data frame object that contains the mapping result for each virus segment that the plant sample reads are aligned to and a RLE list of coverage information.

## Examples

```
## Not run:  
df <- convert_rle_to_df(example_cov)  
  
## End(Not run)
```

---

```
create_high_nucleotide_content  
    create_high_nucleotide_content
```

---

**Description**

Creates a data frame of the start and end positions of the regions\_a that are considered high in A% and GC%.

**Usage**

```
create_high_nucleotide_content(gc = 0.6, a = 0.45, window = 75)
```

**Arguments**

gc	The threshold for GC content. It is the proportion of G and C nucleotides in a sliding window. Default is 0.6.
a	The threshold for A nucleotide. It is the proportion of A nucleotide in a sliding window. Default is 0.45.
window	The sliding window size of your choice. Default is 75.

**Value**

A data frame of the start and end positions of the regions\_a that are considered high in A% and GC%.

**Examples**

```
## Not run: high_nucleotides_regions <- create_high_nucleotide_content()
```

---

```
create_mappability_profile  
    create_mappability_profile
```

---

**Description**

Creates a data frame of start and end positions of the regions that are considered unmappable. Unmappable areas indicate that they can be mapped to another virus segment or a host genome. Note that we only have Arabidopsis Thaliana as a host.

**Usage**

```
create_mappability_profile(path_to_bam_files, category, window = 75)
```

**Arguments**

path_to_bam_files	Path to the folder that stores the indexed and sorted BAM file(s).
category	Type of unreliable region you are creating. You can use categories in the provided unreliable_regions data frame or customize in your own way.
window	The sliding window size of your choice. Default is 75.

**Value**

A data frame of start and end positions of the regions that are considered unmappable.

**Examples**

```
## Not run:
## Please change the path to your folder where you store the mapped viruses
mappability_profile <- create_mappability_profile("path/to/folder",
  category = "Unmappable regions")

## End(Not run)
```

---

example\_cov

*Coverage profiles of three plant samples.*

---

**Description**

A list of coverage profiles for three plant samples. This is only a toy sample. You can use it for running the examples in the vignette. We recommend using more data to train the model, the more the better.

**Usage**

```
example_cov
```

**Format**

A list of 3 run length encoding (RLE) lists for 3 plant samples. Each RLE list has the RLE vector of a virus segment

---

example\_diag                      *Known diagnostics result of virus segments*

---

### Description

A matrix containing the known truth about the diagnostics result for each plant sample for the example data. It records whether the sample is infected with a virus segment. Each column is a sample, and each row is a virus segment's diagnostics status for three samples.

### Usage

example\_diag

### Format

A matrix with 3 columns:

**S1** Sample one

**S2** Sample two

**S3** Sample three

---

nucleotide\_info                      *Nucleotide information of virus segments*

---

### Description

A data set containing the GC content and other information about the virus segments from the official Virtool virus data base (version 1.4.0). The variables are as follows:

### Usage

nucleotide\_info

### Format

A data frame with 7 variables:

**virus\_name** The virus name

**iso\_id** The virus isolate ID

**seg\_id** The virus segment ID

**A\_percent** The percentage of A nucleotides in the virus segment

**C\_percent** The percentage of C nucleotides in the virus segment

**T\_percent** The percentage of T nucleotides in the virus segment

**GC\_percent** The percentage of G and C nucleotides in the virus segment (GC content)

**seg\_len** The length of the virus segment

---

plot_cov	<i>plot_cov()</i>
----------	-------------------

---

**Description**

Plots the coverage profile of the mapped plant sample.

**Usage**

```
plot_cov(  
  covs,  
  legend_status = TRUE,  
  nucleotide_status = TRUE,  
  window = 75,  
  ...  
)
```

**Arguments**

covs	An RLE list of coverage information of one or more plant samples.
legend_status	Whether display legend. Default is TRUE.
nucleotide_status	Whether display a sliding window of A percentage and CG content. Default is TRUE.
window	The sliding window size. Default is 75.
...	Other arguments that can be passed to plot, lines, or legend.

**Value**

The coverage profile of the mapped plant sample.

**Examples**

```
plot_cov(example_cov$S1)
```

---

predict_iimi	<i>predict_iimi()</i>
--------------	-----------------------

---

### Description

Uses a machine learning model to predict the infection status for the plant sample(s). User can use their own model if needed.

### Usage

```
predict_iimi(newdata, method = "xgb", trained_model, report_virus_level = TRUE)
```

### Arguments

newdata	A matrix or data frame that contains the features extracted from the coverage profile using <code>convert_bam_to_cov()</code> .
method	The machine learning method of choice, <code>rf</code> , <code>xgb</code> , or <code>en</code> . <code>rf</code> stands for random forest model; <code>xgb</code> stands for XGBoost model; and <code>en</code> stands for elastic net model.
trained_model	The trained model. If not provided, default model is used.
report_virus_level	If <code>TRUE</code> , the function returns the aggregated results based on the virus. If <code>FALSE</code> , the function returns the unaggregated results based on segment level with each decision's probability decided by the model. We do not recommended to set this to <code>FALSE</code> .

### Value

A data frame of diagnostics result for each sample

### Examples

```
## Not run: df <- convert_rle_to_df(example_cov)
predictions <- predict_iimi(df)

## End(Not run)
```



---

trained_en	<i>A trained model using the default Elastic Net settings</i>
------------	---

---

**Description**

A trained model using the default Elastic Net settings

**Usage**

```
trained_en
```

**Format**

An object of class `cv.glmnet` of length 13.

---

trained_rf	<i>A trained model using the default Random Forest settings</i>
------------	---

---

**Description**

A trained model using the default Random Forest settings

**Usage**

```
trained_rf
```

**Format**

An object of class `randomForest.formula` (inherits from `randomForest`) of length 15.

---

trained_xgb	<i>A trained model using the default XGBoost settings</i>
-------------	---

---

**Description**

A trained model using the default XGBoost settings

**Usage**

```
trained_xgb
```

**Format**

An object of class `list` of length 2.

---

<code>train_iimi</code>	<code>train_iimi()</code>
-------------------------	---------------------------

---

### Description

Trains a XGBoost (default), Random Forest, or Elastic Net model using user-provided data.

### Usage

```
train_iimi(
  train_x,
  train_y,
  method = "xgb",
  nrounds = 100,
  max_depth = 10,
  gamma = 6,
  ntree = 100,
  k = 5,
  ...
)
```

### Arguments

<code>train_x</code>	A data frame or a matrix of predictors.
<code>train_y</code>	A response vector of labels (needs to be a factor).
<code>method</code>	The machine learning method of choice, Random Forest or XGBoost, or Elastic Net model. Default is XGBoost model.
<code>nrounds</code>	Max number of boosting iterations for XGBoost model. Default is 100.
<code>max_depth</code>	Maximum depth of a tree in XGBoost model. Default is 10.
<code>gamma</code>	Minimum loss reduction required in XGBoost model. Default is 6.
<code>ntree</code>	Number of trees in Random Forest model. Default is 100.
<code>k</code>	Number of folds. Default is 5.
<code>...</code>	Other arguments that can be passed to <code>randomForest</code> , <code>xgboost</code> , or <code>glmnet</code> .

### Value

A Random Forest, XGBoost, Elastic Net model

### Examples

```
## Not run:
df <- convert_rle_to_df(example_cov)
train_x <- df[, -c(1:4)]
train_y = c()
for (ii in 1:nrow(df)) {
```

```

    seg_id = df$seg_id[ii]
    sample_id = df$sample_id[ii]
    train_y = c(train_y, example_diag[seg_id, sample_id])
  }
  trained_model <- train_iimi(train_x = train_x, train_y = train_y)

## End(Not run)

```

---

unreliable\_regions      *The unreliable regions of the virus segments*

---

### Description

A data frame of unmappable regions and regions of CG% and A% over 60% and 45% respectively for the virus segments. It is worth to note that if a virus segment does not have any unreliable regions, that virus segment is not shown in this data frame.

### Usage

```
unreliable_regions
```

### Format

A data frame of unreliable regions in the run-length encoding format for virus segments.

**Start** The start position of the region that is considered unreliable

**End** The end position of the region that is considered unreliable

**Virus segment** The virus segment ID

**Categories** The category that this unreliable region belong to, which are Unmappable regions (host), Unmappable regions (virus), CG% > 60%, A% > 45%.

---

virus\_segments      *A DNASTringSet of virus segments from the Virtool virus data base (version 1.4.0)*

---

### Description

A DNASTringSet of virus segments from the Virtool virus data base (version 1.4.0)

### Usage

```
virus_segments
```

**Format**

An object of class DNASTringSet of length 3138.

# Index

## \* datasets

- example\_cov, 5
- example\_diag, 6
- nucleotide\_info, 6
- trained\_en, 9
- trained\_rf, 9
- trained\_xgb, 9
- unreliable\_regions, 11
- virus\_segments, 11

  

- convert\_bam\_to\_rle, 2
- convert\_rle\_to\_df, 3
- create\_high\_nucleotide\_content, 4
- create\_mappability\_profile, 4

  

- example\_cov, 5
- example\_diag, 6

  

- nucleotide\_info, 6

  

- plot\_cov, 7
- predict\_iimi, 8

  

- train\_iimi, 10
- trained\_en, 9
- trained\_rf, 9
- trained\_xgb, 9

  

- unreliable\_regions, 11

  

- virus\_segments, 11