Package ‘REPTILE’

October 12, 2022

Type Package
Title Regulatory DNA Element Prediction
Version 1.0
Date 2016-6-16
Author Yupeng He
Description Predicting regulatory DNA elements based on epigenomic signatures. This package is more of a set of building blocks than a direct solution. REPTILE regulatory prediction pipeline is built on this R package. See <https://github.com/yupenghe/REPTILE> for more information.
Maintainer Yupeng He <yupeng.he.bioinfo@gmail.com>
URL https://github.com/yupenghe/REPTILE
License BSD_2_clause + file LICENSE
Depends R (>= 3.2.2), foreach (>= 1.4.3), doParallel (>= 1.0.10)
Imports optparse (>= 1.3.2), randomForest (>= 4.6-12), flux(>= 0.3-0)
NeedsCompilation no
Repository CRAN
Date/Publication 2016-06-21 09:23:58

R topics documented:

  REPTILE-package .............................................. 2
  calculate_epimark_deviation .................................... 4
  get_option_parser_compute_score ................................ 5
  get_option_parser_evaluation .................................... 5
  get_option_parser_training ..................................... 6
  read_epigenomic_data ........................................... 6
  read_label ...................................................... 7
  reptile_eval_prediction ....................................... 8
  reptile_predict ............................................... 9
  reptile_predict_genome_wide .................................. 11
  reptile_predict_one_mode .................................... 13
Description

Predicting DNA regulatory elements based on epigenomic signatures. This package is more of a set of building blocks than a direct solution. REPTILE regulatory prediction pipeline is built on this R package. Please check the url below for details:

https://github.com/yupenghe/REPTILE

Details

Accurate enhancer identification is critical for understanding the spatiotemporal transcriptional regulation during development as well as the functional impact of disease-related non-coding genetic variants. REPTILE is an algorithm to identify the precise location of enhancers by integrating histone modification data and base-resolution DNA methylation profiles.

REPTILE was designed based on three observations: 1) regions that are differentially methylated (or differentially methylated regions, DMRs) across diverse cell and tissue types strongly overlap with enhancers. 2) With base-resolution DNA methylation data, the boundaries of DMRs can be accurately defined, circumventing the difficulty of determining enhancer boundaries. 3) DMR size is often smaller (~500bp) than known enhancers, known negative regions (regions with no observable enhancer activity) and genomic windows used in enhancer prediction (~2kb), all of which we termed as "query regions". Together with the association between transcription factor binding and DNA methylation level, DMRs may serve as high-resolution enhancer candidates and capture the local epigenomic patterns that would otherwise be averaged/washed out in analysis focusing on the query regions.

Running REPTILE involves four major steps. First, to identify DMRs, we compared the methylomes of target sample (where putative enhancers will be generated) and several other samples with different cell/tissue types (as reference). In the next step, input files for REPTILE are prepared, which store the information of query regions, DMRs and the epigenomic data. Taking these inputs, REPTILE represents each DMR or query region as a feature vector, where each element corresponds to either intensity or intensity deviation of one epigenetic mark. Intensity deviation is defined as the intensity in target sample subtracted by the mean intensity in reference samples (i.e. reference epigenome) and it captures the tissue-specificity of each epigenetic mark. In the third step, based on the feature vectors of known enhancers and negative regions as well as the feature vectors of the DMRs within them, we trained an enhancer model, containing two random forest classifiers, which respectively predict enhancer activities of query regions and DMRs. In the last step, REPTILE uses the enhancer model to calculate enhancer confidence scores for DMRs and query regions, based on which the final predictions are made.

The two key concepts on REPTILE are:
REPTILE-package

- Query regions - known enhancers, known negative regions and genomic windows used for enhancer prediction
- DMRs - differentially methylated regions

In REPTILE, DMRs are used as high-resolution candidates to capture the fine epigenomic signatures in query regions.

Author(s)

Yupeng He
Maintainer: Yupeng He <yupeng.he.bioinfo@gmail.com>

References

He, Yupeng et al., REPTILE: Regulatory Element Prediction based on Tissue-specific Local Epigenetic marks, in preparation

Examples

library("REPTILE")
data("rsd")

## Training (needs a few minutes and ~1.8 Gb memory)
reptile.model <- reptile_train(rsd$training_data$region_epimark,
rsd$training_data$region_label,
rsd$training_data$DMR_epimark,
rsd$training_data$DMR_label,
ntree=50)

## Prediction
## - REPTILE
pred <- reptile_predict(reptile.model,
rsd$test_data$region_epimark,
rsd$test_data$DMR_epimark)

## - Random guessing
pred_guess = runif(length(pred$D))
names(pred_guess) = names(pred$D)

## Evaluation
res_reptile <- reptile_eval_prediction(pred$D,
rsd$test_data$region_label)
res_guess <- reptile_eval_prediction(pred_guess,
rsd$test_data$region_label)

## - Print AUROC and AUPR
cat(paste0("REPTILE\n"," AUROC = ",round(res_reptile$AUROC,digit=3),
"\n"," AUPR = ",round(res_reptile$AUPR,digit=3)))
cat(paste0("Random guessing\n"," AUROC = ",round(res_guess$AUROC,digit=3),
"\n",})
calculate_epimark_deviation

Description

Internal function used to calculate the intensity deviation features. It is based on the epigenomic signatures of a given region in target sample, where prediction will be generated, and reference samples. Intensity deviation is defined as the intensity in target sample subtracted by the mean intensity in reference samples (i.e. reference epigenome) and it captures the tissue-specificity of each epigenetic mark.

Usage

```
calculate_epimark_deviation(data_info, x, query_sample, ref_sample = NULL)
```

Arguments

- `data_info` data.frame instance generated by reading data information file specifying the samples and marks used in the analysis. The data.frame includes at least two columns named "sample" and "mark", corresponding to the samples and marks included.
- `x` data.frame instance generated by reading epimark file. The first four columns of the data.frame are "chr", "start", "end" and "id" of each region in the epimark file. The rest columns contain values of epigenetic marks in samples as specified in data_info and column names are under MARK_SAMPLE format, such as "H3K4me1_mESC".
- `query_sample` name of the target sample
- `ref_sample` a vector of names of the reference sample(s)

Value

data.frame instance containing intensity deviation values of each mark

Author(s)

Yupeng He <yupeng.he.bioinfo@gmail.com>

See Also

- `read_epigenomic_data`
get_option_parser_compute_score

*Internal - parsing options for REPTILE_compute_score.R*

**Description**

Internal function used to parsing options for "REPTILE_compute_score.R" script in the REPTILE enhancer prediction pipeline:

https://github.com/yupenghe/REPTILE

**Usage**

get_option_parser_compute_score()

**Value**

An instance of the OptionParser class.

**Author(s)**

Yupeng He <yupeng.he.bioinfo@gmail.com>

get_option_parser_evaluation

*Internal - parsing options for REPTILE_evaluate_prediction.R*

**Description**

Internal function used to parsing options for "REPTILE_evaluate_prediction.R" script in the REPTILE enhancer prediction pipeline:

https://github.com/yupenghe/REPTILE

**Usage**

get_option_parser_evaluation()

**Value**

An instance of the OptionParser class.

**Author(s)**

Yupeng He <yupeng.he.bioinfo@gmail.com>
get_option_parser_training

*Internal - parsing options for REPTILE_train.R*

**Description**

Internal function used to parsing options for "REPTILE_train.R" script in the REPTILE enhancer prediction pipeline:

https://github.com/yupenghe/REPTILE

**Usage**

get_option_parser_training()

**Value**

An instance of the OptionParser class.

**Author(s)**

Yupeng He <yupeng.he.bioinfo@gmail.com>

---

read_epigenomic_data  
*Reading epigenomic data from epimark file*

**Description**

Function to read epimark file from disk and generate data.frame instance. It is used to read epigenomic data from file on disk and generate the input data.frame instance to fuel the model training, prediction and other following steps. Epimark file is a tab-separated file with a header. The first four columns are "chr", "start", "end" and "id", specifying the chromosome, start, end and id of regions. Each of the remaining columns contain values of one epigenetic mark in one sample (condition, cell or tissue type, etc) and the column name follows "MARKSAMPLE" format, such as "H3K4me1_mESC".

**Usage**

read_epigenomic_data(data_info, epimark_file, query_sample, 
ref_sample = NULL, incl_dev = T)
Arguments

data_info data.frame generated by reading data information file specifying the samples and marks used in the analysis. The data.frame includes at least two columns named "sample" and "mark", corresponding to the samples and marks included.

epimark_file name of epimark file

query_sample name of the target sample

ref_sample a vector of names of the reference sample(s)

incl_dev logical value indicates whether to calculate the intensity deviation feature. Intensity deviation is defined as the intensity in target sample subtracted by the mean intensity in reference samples (i.e. reference epigenome) and it captures the tissue-specificity of each epigenetic mark.

Value

data.frame instance containing intensity and intensity deviation values of each mark for each region

Author(s)

Yupeng He <yupeng.he.bioinfo@gmail.com>

See Also

read_label

Description

Function to read epimark file from disk and generate data.frame instance. It is used to read epigenomic data from file on disk and generate the input data.frame instance to fuel the model training, prediction and other following steps. Label file is a tab-separated file with a header. The first column contains the id of each region. The second or more columns specify whether a certain region is enhancer (1) or not (0) in a specific sample. Each of these columns corresponds to one sample and the name of the column is the sample name.

Usage

read_label(label_file, query_sample)

Arguments

label_file name of label file on disk

query_sample name(s) of sample(s), in which you would like to have label information
reptile_eval_prediction

Evaluating the prediction results

Description

Function used to evaluate the predictions by comparing enhancer scores from reptile_predict or reptile_predict_genome_wide and the correct labels. Area under the Receiver Operating Characteristic (ROC) curve (AUROC) and Area under the Precision-Recall curve (AUPR) will be calculated.

Usage

reptile_eval_prediction(predictions, annotations)

Arguments

predictions vector of enhancer scores for regions. The name of each value (score) corresponds to the id of the region.

annotations vector of labels for regions with the same length as predictions. The name of each value (label) corresponds to the id of the region. Only two values are allowed in annotations: 0 (negative) and 1 (positive). No NA is allowed.

Value

A list containing two numbers

AUROC Area under the Receiver Operating Characteristic (ROC) curve

AUPR Area under the Precision-Recall curve
Author(s)
Yupeng He <yupeng.he.bioinfo@gmail.com>

See Also
reptile_predict, reptile_predict_genome_wide

Examples

library("REPTILE")
data("rsd")

## Training
rsd_model <- reptile_train(rsd$training_data$region_epimark,
                         rsd$training_data$region_label,
                         rsd$training_data$DMR_epimark,
                         rsd$training_data$DMR_label,
                         ntree=50)

## Prediction
## - REPTILE
pred <- reptile_predict(rsd_model,
                         rsd$test_data$region_epimark,
                         rsd$test_data$DMR_epimark)

## - Random guessing
pred_guess = runif(length(pred$D))
names(pred_guess) = names(pred$D)

## Evaluation
res_reptile <- reptile_eval_prediction(pred$D,
                                         rsd$test_data$region_label)
res_guess <- reptile_eval_prediction(pred_guess,
                                      rsd$test_data$region_label)

## - Print AUROC and AUPR
cat(paste0("REPTILE\n",
           " AUROC = ",round(res_reptile$AUROC,digit=3),
           "\n",
           " AUPR = ",round(res_reptile$AUPR,digit=3)))

"\n")
cat(paste0("Random guessing\n",
           " AUROC = ",round(res_guess$AUROC,digit=3),
           "\n",
           " AUPR = ",round(res_guess$AUPR,digit=3)))
"\n")

reptile_predict  Predicting enhancer activity of given regions
Description
Predicting enhancer activities of query regions based on the enhancer model from reptile_train in training step. This function calculates the combined enhancer score for each query region (given region) as the maximum among the score of whole query region and the scores of DMRs within it. This function is for generating genome-wide enhancer predictions.

Usage
reptile_predict(reptile_model, epimark_region, epimark_DMR = NULL, family = "randomForest")

Arguments
reptile_model Enhancer model from reptile_train. It is a list containing two objects of class randomForest or glm when family is set to be "Logistic"
epimark_region data.frame instance from read_epigenomic_data, which containing intensity and intensity deviation values of each mark for each query region
epimark_DMR data.frame instance from read_epigenomic_data, which containing intensity and intensity deviation values of each mark for each DMR
family classifier family used in the enhancer model
  Default: RandomForest
  Classifiers available:
  - RandomForest: random forest
  - Logistic: logistic regression

Value
A list containing three vectors
D Combined enhancer score of each query region
R Enhancer score of each query region
DMR Enhancer score of each DMR

Author(s)
Yupeng He <yupeng.he.bioinfo@gmail.com>

See Also
reptile_predict_genome_wide
reptile_train
read_epigenomic_data
read_label
Examples

```r
library("REPTILE")
data("rsd")

## Training
rsd_model <- reptile_train(rsd$training_data$region_epimark,
rsd$training_data$region_label,
rsd$training_data$DMR_epimark,
rsd$training_data$DMR_label,
ntree=50)

## Prediction
## - REPTILE
pred <- reptile_predict(rsd_model,
rsd$test_data$region_epimark,
rsd$test_data$DMR_epimark)

## - Random guessing
pred_guess = runif(length(pred$D))
names(pred_guess) = names(pred$D)

## Evaluation
res_reptile <- reptile_eval_prediction(pred$D,
rsd$test_data$region_label)
res_guess <- reptile_eval_prediction(pred_guess,
rsd$test_data$region_label)

## - Print AUROC and AUPR
cat(paste0("REPTILE
", " AUROC = ",round(res_reptile$AUROC,digit=3),
"\n", " AUPR = ",round(res_reptile$AUPR,digit=3))
,"\n")
cat(paste0("Random guessing\n", " AUROC = ",round(res_guess$AUROC,digit=3),
"\n", " AUPR = ",round(res_guess$AUPR,digit=3))
,"\n")
```

reptile_predict_genome_wide

*Predicting enhancer activity*

Description

Predicting enhancer activities of query regions based on the enhancer model from `reptile_train` in training step. This function calculates the enhancer scores of DMRs and query regions. It does not try to generate combined enhancer scores.
Usage

```r
reptile_predict_genome_wide(reptile_model,
epimark_region,
epimark_DMR = NULL,
family = "randomForest")
```

Arguments

- `reptile_model`: Enhancer model from `reptile_train`. It is a list containing two objects of class `randomForest` or `glm` when `family` is set to be "Logistic".
- `epimark_region`: data.frame instance from `read_epigenomic_data`, which containing intensity and intensity deviation values of each mark for each query region.
- `epimark_DMR`: data.frame instance from `read_epigenomic_data`, which containing intensity and intensity deviation values of each mark for each DMR.
- `family`: classifier family used in the enhancer model. Default: RandomForest. Classifiers available:
  - RandomForest: random forest
  - Logistic: logistic regression

Value

A list containing two vectors

- `R`: Enhancer score of each query region
- `DMR`: Enhancer score of each DMR

Author(s)

Yupeng He <yupeng.he.bioinfo@gmail.com>

See Also

- `reptile_predict`
- `reptile_train`
- `read_epigenomic_data`
- `read_label`

Examples

```r
library("REPTILE")
data("rsd")

## Training
rsd_model <- reptile_train(rsd$training_data$region_epimark,
rsd$training_data$region_label,
rsd$training_data$DMR_epimark,
rsd$training_data$DMR_label)
```
## Prediction

**- REPTILE**

```r
pred <- reptile_predict(rsd_model,
                        rsd$test_data$region_epimark,
                        rsd$test_data$DMR_epimark)
```

**- Random guessing**

```r
pred_guess = runif(length(pred$D))
names(pred_guess) = names(pred$D)
```

## Evaluation

```r
res_reptile <- reptile_eval_prediction(pred$D,
                                        rsd$test_data$region_label)
res_guess <- reptile_eval_prediction(pred_guess,
                                     rsd$test_data$region_label)
```

**- Print AUROC and AUPR**

```r
cat(paste0("REPTILE
",  " AUROC = ",round(res_reptile$AUROC,digit=3),
       "\n",  " AUPR = ",round(res_reptile$AUPR,digit=3)))

cat(paste0("Random guessing\n",  " AUROC = ",round(res_guess$AUROC,digit=3),
       "\n",  " AUPR = ",round(res_guess$AUPR,digit=3)))
```

---

**reptile_predict_one_mode**

*Internal - predicting enhancer activity of DMRs or query regions*

### Description

Internal function used to predict the enhancer activity of either DMRs or query regions.

### Usage

```r
reptile_predict_one_mode(reptile_classifier,
                         epimark,
                         family)
```

### Arguments

- **reptile_classifier**
  
  An object of class `randomForest` or `glm` when `family` is set to be "Logistic".

- **epimark**
  
  data.frame instance from `read_epigenomic_data`, which containing intensity and intensity deviation values of each mark for each query region
**reptile_train**

Learn a REPTILE enhancer model

Description

Learn a REPTILE enhancer model based on epigenomic signature of known enhancers.

Usage

```r
reptile_train(epimark_region, label_region,
epimark_DMR = NULL, label_DMR = NULL,
family = "randomForest", ntree = 2000,
nodeSize = 1)
```

Arguments

- **epimark_region**: data.frame instance from read_epigenomic_data, which containing intensity and intensity deviation values of each mark for each query region.
- **label_region**: factor instance from read_label, containing the label of each query region. The possible values and their meanings of a label are: 0 (not enhancer), 1 (enhancer) and NA (unknown and it will be ignored).
- **epimark_DMR**: data.frame instance from read_epigenomic_data, which containing intensity and intensity deviation values of each mark for each DMR. If either this value or label_DMR is NULL, the output enhancer model will not include a classifier for predicting the enhancer activities of DMRs. Default: NULL.
- **family**: classifier family used in the enhancer model
  Default: RandomForest
  Classifiers available:
  - RandomForest: random forest
  - Logistic: logistic regression

Value

A vector of enhancer score of each query region or DMR

Author(s)

Yupeng He <yupeng.he.bioinfo@gmail.com>

See Also

- `reptile_predict`
- `reptile_predict_genome_wide`
factor instance from read_label, containing the label of each DMR. The possible
values and their meanings of a label are: 0 (not enhancer), 1 (enhancer) and
NA (unknown and it will be ignored). If either this value or label_DMR is
NULL, the output enhancer model will not include a classifier for predicting
the enhancer activities of DMRs. Default: NULL

classifier family used in the enhancer model
Default: RandomForest
Classifiers available:
- RandomForest: random forest
- Logistic: logistic regression

Number of tree to be constructed in the random forest model. See the function
randomForest() in "randomForest" package for more information. Default: 2000

Minimum size of terminal nodes. See the function randomForest() in "random-
Forest" package for more information. Default: 1

A list containing two objects of class randomForest.

Classifier for DMRs. It is an randomForest object or glm object when family
is set to be "Logistic".

Classifier for query regions. It is an randomForest object or glm object when
family is set to be "Logistic".

Yupeng He <yupeng.he.bioinfo@gmail.com>


read_epigenomic_data, read_label, reptile_predict

library("REPTILE")
data("rsd")

## Training
rsd_model <- reptile_train(rsd$training_data$region_epimark,
rsd$training_data$region_label,
rsd$training_data$DMR_epimark,
rsd$training_data$DMR_label,
```r
reptile_train_one_mode

ntree=5)
print(rsd_model$D)
print(rsd_model$R)
```

---

**Description**

Internal function to learn a random forest classifier.

**Usage**

```r
t reptile_train_one_mode(epimark, label,
 family, ntree, nodesize)
```

**Arguments**

- **epimark**: data.frame instance from read_epigenomic_data, which containing intensity and intensity deviation values of each mark for each DMR or query region.
- **label**: factor instance from read_label, containing the label of each query region. The possible values and their meanings of a label are: 0 (not enhancer), 1 (enhancer) and NA (unknown and will be ignored).
- **family**: Classifier family used in the enhancer model. Default: RandomForest Classifiers available: - RandomForest: random forest - Logistic: logistic regression
- **ntree**: Number of tree to be constructed in the random forest model. See the function randomForest() in "randomForest" package for more information. Default: 2000
- **nodesize**: Minimum size of terminal nodes. See the function randomForest() in "randomForest" package for more information. Default: 1

**Value**

An randomForest object or glm object when family is set to be "Logistic".

**Author(s)**

Yupeng He <yupeng.he.bioinfo@gmail.com>

**References**


See Also

reptile_train

---

rsd  
REPTILE sample data (rsd)

Description

sample data for testing REPTILE training, prediction and evaluation.

Usage

data(rsd)

Format

A list containing two lists.

training_data is the data used for training REPTILE enhancer model. This list has four elements: region_epimark, DMR_epimark, region_label and DMR_label. The former two store the epigenomic signatures of query regions and DMRs. The latter two label which a certain query region or DMR is enhancer (1) or negative instance (0)

test_data is for training REPTILE enhancer model and it has four elements: region_epimark, DMR_epimark and region_label. The former two store the epigenomic signatures of query regions and DMRs. The region_label indicates whether a certain query region or DMR is enhancer (1) or negative instance (0)

Author(s)

Yupeng He <yupeng.he.bioinfo@gmail.com>

Source

training_data was based on the EP300 binding sites (positives), promoters (negatives) and randomly chosen genomic loci (negatives) in mouse embryonic stem cells.

The test_data data was constructed based on in vivo validated mouse sequences from VISTA enhancer browser (Oct 24th, 2015). The labels indicate the activity in mouse heart tissues from E11.5 embryo.

See the papers included in References for details.

References

He, Yupeng et al., REPTILE: Regulatory Element Prediction based on Tissue-specific Local Epigenetic marks, in preparation

Examples

```r
## Visualizing rsd data
library("REPTILE")
data(rsd)

## Epigenomic signature of query region grouped by labels
ind_pos = rsd$training_data$region_label == 1
pos_region = rsd$training_data$region_epimark[ind_pos,]
neg_region = rsd$training_data$region_epimark[!ind_pos,]

## Epigenomic signature of DMRs grouped by labels
ind_pos = rsd$training_data$DMR_label == 1
pos_DMR = rsd$training_data$DMR_epimark[ind_pos,]
neg_DMR = rsd$training_data$DMR_epimark[!ind_pos,]

## Prepare the data format required for plotting
n = ncol(rsd$training_data$DMR_epimark) ## Number of features
feature_data_DMR = list()
feature_data_region = list()
for(i in 1:n){
  feature_data_DMR <- append(feature_data_DMR,
                              list(neg_DMR[,i],pos_DMR[,i],
                                   NA,NA))
  feature_data_region <- append(feature_data_region,
                                list(neg_region[,i],pos_region[,i],
                                     NA,NA))
}

## Plot the feature distribution
par(mar=c(4,8,4,4))
## - query region
b <- boxplot(feature_data_region,
             xlab = "feature value",
             notch=TRUE,outline=FALSE,yaxt='n',
             xlim = c(1,n+4-2),ylim=c(-7,7),
             horizontal=TRUE,
             col=c(rgb(65,105,225,max=255),rgb(250,128,114,max=255)),
             main = "Feature value distribution in query regions"
)
text(par("usr")[1]-0.2, seq(1.5,n+4-2,by=4),
     labels=gsub("_","-",colnames(rsd$training_data$region_epimark)),
     xpd = TRUE,adj=1)
legend(-8,4+n+4,c("negative","enhancer"),ncol=2,
      fill = c(rgb(250,128,114,max=255),rgb(65,105,225,max=255)),
      xpd=TRUE,bty='n')

## - DMR
b <- boxplot(feature_data_DMR,
             xlab = "feature value",
             notch=TRUE,outline=FALSE,yaxt='n',
             xlim = c(1,n+4-2),ylim=c(-7,7),
             horizontal=TRUE,
             col=c(rgb(65,105,225,max=255),rgb(250,128,114,max=255)),
             main = "Feature value distribution in DMRs"
)
```

col=c(rgb(65,105,225,max=255),rgb(250,128,114,max=255)),
main = "Feature value distribution in DMRs"
)
text(par("usr")[1]-0.2, seq(1.5,n*4-2,by=4),
labels=gsub("_","-",colnames(rsd$training_data$DMR_epimark)),
xpd = TRUE,adj=1)
legend(-8,4*n+4,c("negative","enhancer"),ncol=2,
fill = c(rgb(250,128,114,max=255),rgb(65,105,225,max=255)),
xpd=TRUE,bty='n')
Index

* dataset
  rsd, 17
* model training
  reptile_train, 14
* option parser
  get_option_parser_compute_score, 5
  get_option_parser_evaluation, 5
  get_option_parser_training, 6
* prediction
  reptile_predict, 9
  reptile_predict_genome_wide, 11
* read input file
  read_epigenomic_data, 6
  read_label, 7
* result evaluation
  reptile_eval_prediction, 8

calculate_epimark_deviation, 4
get_option_parser_compute_score, 5
get_option_parser_evaluation, 5
get_option_parser_training, 6
read_epigenomic_data, 4, 6, 8, 10, 12, 15
read_label, 7, 7, 10, 12, 15
REPTILE (REPTILE-package), 2
REPTILE-package, 2
reptile_eval_prediction, 8
reptile_predict, 9, 9, 12, 14, 15
reptile_predict_genome_wide, 9, 10, 11, 14
reptile_predict_one_mode, 13
reptile_train, 10, 12, 14, 17
reptile_train_one_mode, 16
rsd, 17