Package 'PheCAP'

July 21, 2025

Type Package

Title High-Throughput Phenotyping with EHR using a Common Automated Pipeline

Version 1.2.1

Description Implement surrogate-assisted feature extraction (SAFE) and common machine learning approaches to train and validate phenotyping models. Background and details about the methods can be found at Zhang et al. (2019) <doi:10.1038/s41596-019-0227-6>, Yu et al. (2017) <doi:10.1093/jamia/ocw135>, and Liao et al. (2015) <doi:10.1136/bmj.h1885>.

URL https://celehs.github.io/PheCAP/, https://github.com/celehs/PheCAP

BugReports https://github.com/celehs/PheCAP/issues

License GPL-3

Encoding UTF-8

ByteCompile yes

Imports graphics, methods, stats, utils, glmnet, RMySQL

Suggests ggplot2, e1071, randomForestSRC, xgboost, knitr, rmarkdown

VignetteBuilder knitr

Depends R (>= 3.3.0)

RoxygenNote 7.1.1

LazyData true

NeedsCompilation no

Author Yichi Zhang [aut],
Chuan Hong [aut],
Tianxi Cai [aut],
PARSE LTD [aut, cre]

Maintainer PARSE LTD <software@parse-health.org>

Repository CRAN

Date/Publication 2020-09-17 09:30:18 UTC

Contents

`PheCAP-package` *High-Throughput Phenotyping with EHR using a Common Automated Pipeline*

Description

Implement surrogate-assisted feature extraction (SAFE) and common machine learning approaches to train and validate phenotyping models. Background and details about the methods can be found at Zhang et al. (2019) <doi:10.1038/s41596-019-0227-6>, Yu et al. (2017) <doi:10.1093/jamia/ocw135>, and Liao et al. (2015) <doi:10.1136/bmj.h1885>.

Details

The DESCRIPTION file:

Package:	PheCAP
Type:	Package
Title:	High-Throughput Phenotyping with EHR using a Common Automated Pipeline
Version:	1.2.1
Authors@R:	c(person("Yichi", "Zhang", role = "aut"), person("Chuan", "Hong", role = "aut"), person("Tianxi", "Cai", r
Description:	Implement surrogate-assisted feature extraction (SAFE) and common machine learning approaches to train
URL:	https://celehs.github.io/PheCAP/, https://github.com/celehs/PheCAP
BugReports:	https://github.com/celehs/PheCAP/issues
License:	GPL-3
Encoding:	UTF-8
ByteCompile:	yes
Imports:	graphics, methods, stats, utils, glmnet, RMySQL
Suggests:	ggplot2, e1071, randomForestSRC, xgboost, knitr, rmarkdown
VignetteBuilder:	knitr
Depends:	R (>= 3.3.0)
RoxygenNote:	7.1.1
LazyData:	true

Author: Yichi Zhang [aut], Chuan Hong [aut], Tianxi Cai [aut], PARSE LTD [aut, cre]
Maintainer: PARSE LTD <software@parse-health.org>

Index of help topics:

PheCAP provides a straightforward interface for conducting phenotyping on eletronic health records. One can specify the data via PhecapData, define surrogate using PhecapSurrogate. Next, one may run surrogate-assisted feature extraction (SAFE) by calling phecap_run_feature_extraction, and then train and validate phenotyping models via phecap_train_phenotyping_model and phecap_validate_phenotypi The predictive performance can be visualized using phecap_plot_roc_curves. Predicted phenotype is provided by phecap_predict_phenotype.

Author(s)

NA

Maintainer: NA

References

Yu, S., Chakrabortty, A., Liao, K. P., Cai, T., Ananthakrishnan, A. N., Gainer, V. S., ... & Cai, T. (2016). Surrogate-assisted feature extraction for high-throughput phenotyping. Journal of the American Medical Informatics Association, 24(e1), e143-e149.

Liao, K. P., Cai, T., Savova, G. K., Murphy, S. N., Karlson, E. W., Ananthakrishnan, A. N., ... & Churchill, S. (2015). Development of phenotype algorithms using electronic medical records and incorporating natural language processing. bmj, 350, h1885.

Examples

```
# Simulate an EHR dataset
size <- 2000
latent <- rgamma(size, 0.3)
latent2 <- rgamma(size, 0.3)
ehr_data <- data.frame(
  ICD1 = rpois(size, 7 * (rgamma(size, 0.2) + latent) / 0.5),
  ICD2 = rpois(size, 6 * (rgamma(size, 0.8) + latent) / 1.1),
  ICD3 = rpois(size, 1 * rgamma(size, 0.5 + latent2) / 0.5),
  ICD4 = rpois(size, 2 * rgamma(size, 0.5) / 0.5),
  NLP1 = rpois(size, 8 * (rgamma(size, 0.2) + latent) / 0.6),
  NLP2 = rpois(size, 2 * (rgamma(size, 1.1) + latent) / 1.5),
  NLP3 = rpois(size, 5 * (rgamma(size, 0.1) + latent) / 0.5),
  NLP4 = rpois(size, 11 * rgamma(size, 1.9 + latent) / 1.9),
  NLP5 = rpois(size, 3 * rgamma(size, 0.5 + latent2) / 0.5),
  NLP6 = rpois(size, 2 * rgamma(size, 0.5) / 0.5),
  NLP7 = rpois(size, 1 * rgamma(size, 0.5) / 0.5),
  HU = rpois(size, 30 * rgamma(size, 0.1) / 0.1),
  label = NA)
ii <- sample.int(size, 400)
ehr_data[ii, "label"] <- with(
  ehr_data[ii, ], rbinom(400, 1, plogis(
    -5 + 1.5 * log1p(ICD1) + log1p(NLP1) +
      0.8 * log1p(NLP3) - 0.5 * log1p(HU))))

# Define features and labels used for phenotyping.
data <- PhecapData(ehr_data, "HU", "label", validation = 0.4)
data

# Specify the surrogate used for
# surrogate-assisted feature extraction (SAFE).
# The typical way is to specify a main ICD code, a main NLP CUI,
# as well as their combination.
# The default lower_cutoff is 1, and the default upper_cutoff is 10.
# In some cases one may want to define surrogate through lab test.
# Feel free to change the cutoffs based on domain knowledge.
surrogates <- list(
  PhecapSurrogate(
    variable_names = "ICD1",
    lower_cutoff = 1, upper_cutoff = 10),
  PhecapSurrogate(
    variable_names = "NLP1",
    lower_cutoff = 1, upper_cutoff = 10))

# Run surrogate-assisted feature extraction (SAFE)
# and show result.
feature_selected <- phecap_run_feature_extraction(
  data, surrogates, num_subsamples = 50, subsample_size = 200)
```

```
feature_selected

# Train phenotyping model and show the fitted model,
# with the AUC on the training set as well as random splits.
model <- phecap_train_phenotyping_model(
  data, surrogates, feature_selected, num_splits = 100)
model

# Validate phenotyping model using validation label,
# and show the AUC and ROC.
validation <- phecap_validate_phenotyping_model(data, model)
validation

phecap_plot_roc_curves(validation)

# Apply the model to all the patients to obtain predicted phenotype.
phenotype <- phecap_predict_phenotype(data, model)


# A more complicated example

# Load Data.
data(ehr_data)
data <- PhecapData(ehr_data, "healthcare_utilization", "label", 0.4)
data

# Specify the surrogate used for
# surrogate-assisted feature extraction (SAFE).
# The typical way is to specify a main ICD code, a main NLP CUI,
# as well as their combination.
# In some cases one may want to define surrogate through lab test.
# The default lower_cutoff is 1, and the default upper_cutoff is 10.
# Feel free to change the cutoffs based on domain knowledge.
surrogates <- list(
  PhecapSurrogate(
    variable_names = "main_ICD",
    lower_cutoff = 1, upper_cutoff = 10),
  PhecapSurrogate(
    variable_names = "main_NLP",
    lower_cutoff = 1, upper_cutoff = 10),
  PhecapSurrogate(
    variable_names = c("main_ICD", "main_NLP"),
    lower_cutoff = 1, upper_cutoff = 10))

# Run surrogate-assisted feature extraction (SAFE)
# and show result.
feature_selected <- phecap_run_feature_extraction(data, surrogates)
feature_selected

# Train phenotyping model and show the fitted model,
# with the AUC on the training set as well as random splits
model <- phecap_train_phenotyping_model(data, surrogates, feature_selected)
model
```

```
# Validate phenotyping model using validation label,
# and show the AUC and ROC
validation <- phecap_validate_phenotyping_model(data, model)
validation
phecap_plot_roc_curves(validation)

# Apply the model to all the patients to obtain predicted phenotype.
phenotype <- phecap_predict_phenotype(data, model)
```

---

ehr_data *A Synthetic EHR Dataset*

Description

This dataset gives a sample dataset for EHR phenotyping. It contains counts for ICD codes, counts for NLP mentions, healthcare utilization (HU) features for all observations. It also contains the accurate phenotypes for 181 observations.

Usage

```
data(ehr_data)
```

Format

A data.frame with 10000 observations of 588 variables.

PhecapData *Define or Read Datasets for Phenotyping*

Description

Specify the data to be used for phenotyping.

Usage

```
PhecapData(
  data, hu_feature, label, validation,
  patient_id = NULL, subject_weight = NULL,
  seed = 12300L, feature_transformation = log1p)
```

Arguments

data	A data.frame consisting of all the variables needed for phenotyping, or a character scalar of the path to the data, or a list consisting of either character scalar or data.frame. If a list is given, patient_id cannot be NULL. All the datasets in the list will be joined into a single dataset according to the columns specified by patient_id.
hu_feature	A character scalar or vector specifying the names of one of more healthcare utilization (HU) variables. There variables are always included in the phenotyping model.
label	A character scalar of the column name that gives the phenotype status (1 or TRUE: present, 0 or FALSE: absent). If label is not ready yet, just put a column filled with NA in data. In such cases only the feature extraction step can be done.
validation	A character scalar, a real number strictly between 0 and 1, or an integer not less than 2. If a character scalar is used, it is treated as the column name in the data that specifies whether this observation belongs to the validation samples (1 or TRUE: validation, 0 or FALSE: training). If a real number strictly between 0 and 1 is used, it is treated as the proportion of the validation samples. The actual validation samples will be drawn from all labeled samples. If an integer not less than 2 is used, it is treated as the size of the validation samples. The actual validation samples will be drawn from all labeled samples.
patient_id	A character vector for the column names, if any, that uniquely identifies each patient. Such variables must appear in the data. patient_id can be NULL if such fields are not contained in the data.
subject_weight	An optional numeric vector of weights for observations.
seed	If validation samples need to be drawn from all labeled samples, seed specifies the random seed for sampling.
feature_transformation	A function that will be applied to all the features. Since count data are typically right-skewed, by default `log1p` will be used. feature_transformation can be NULL, in which case no transformation will be done on any of the feature.

Value

An object of class `PhecapData`.

See Also

See `PheCAP-package` for code examples.

`PhecapSurrogate` *Define a Surrogate Variable used in Surrogate-Assisted Feature Extraction (SAFE)*

Description

Define a surrogate varible from existing features, and specify associated lower and upper cutoffs.

Usage

```
PhecapSurrogate(variable_names, lower_cutoff = 1L, upper_cutoff = 10L)
```

Arguments

variable_names	a character scalar or vector consisting of variable names. If a vector is given, the value of the surrogate is defined as the sum of the values of each variable.
lower_cutoff	a numeric scalar. If the surrogate value of a patient is less than or equal to this cutoff, then this patient is treated as a control in SAFE.
upper_cutoff	a numeric scalar. If the surrogate value of a patient is greater than or equal to this cutoff, then this patient is treated as a case in SAFE.

Details

This function only stores the definition. No calculation is done.

Value

An object of class PhecapSurrogate.

See Also

See PheCAP-package for code examples.

---

phecap_generate_dictionary_file

Generate a Dictionary File for Note Parsing

Description

Given a list of CUIs, connect to the UMLS database stored in MySQL, extract CUIs and associated terms, and write a dictionary file for use in note parsing.

Usage

```
phecap_generate_dictionary_file(
  cui_list, dict_file,
  user = "username", password = "password",
  host = "localhost", dbname = "umls", ...)
```

Arguments

cui_list	a character vector consisting of CUIs of interest.
dict_file	a character scalar for the path to the dictionary file that will be generated.
user	a character scalar for the username for database connection; passed to RMySQL::dbConnect as it is.
password	a character scalar for the password for database connection; passed to RMySQL::dbConnect as it is.
host	a character scalar for the host (or URL) for database connection; passed to RMySQL::dbConnect as it is.
dbname	a character scalar for the database name for database connection; passed to RMySQL::dbConnect as it is.
...	Other arguments passed to RMySQL::dbConnect as they are.

Value

The dictionary will be written to the location given by dict_file. Return the dictionary invisibly.

---

phecap_perform_majority_voting

Perform Majority Voting on the CUIs from Multiple Knowledge Sources

---

Description

Read parsed knowledge sources and identify CUIs. Generate a list of CUIs that appear in at least half of the sources.

Usage

```
phecap_perform_majority_voting(
  input_folder)
```

Arguments

input_folder	a character scalar for the path to the folder that contains the parsed knowledge sources

Value

A character vector consisting of CUIs that pass the majority voting criterion.

phecap_plot_roc_curves *Plot ROC and Related Curves for Phenotyping Models*

Description

Plot ROC-like curves to illustrate phenotyping accuracy.

Usage

```
phecap_plot_roc_curves(
  x, axis_x = "1 - spec", axis_y = "sen",
  what = c("training", "random-splits", "validation"),
  ggplot = TRUE, ...)
```

Arguments

x	either a single object of class PhecapModel or PhecapValidation (returned from phecap_train_phenotyping_model or phecap_validate_phenotyping_model), or a named list of such objects
axis_x	an expression that leads to the x coordinate. Recognized quantities include: cut (probability cutoff), pct (percent of predicted cases), acc (accuracy), tpr (true positive rate), fpr (false positive rate), tnr (true negative rate), ppv (positive predictive value), fdr (false discovery rate), npv (negative predictive value), sen (sensitivity), spec (specificity), prec (precision), rec (recall), f1 (F1 score).
axis_y	an expression that leads to the y coordinate. Recognized quantities are the same as those in axis_x.
what	The curves to be included in the figure.
ggplot	if TRUE and ggplot2 is installed, ggplot will be used for the figure. Otherwise, the base R graphics functions will be used.
...	arguments to be ignored.

See Also

See PheCAP-package for code examples.

`phecap_predict_phenotype` *Predict Phenotype*

Description

Compute predicted probability of having the phenotype for each patient in the dataset.

Usage

```
phecap_predict_phenotype(data, model)
```

Arguments

`data`	an object of class `PhecapData`, obtained by calling `PhecapData(...)`.
`model`	an object of class `PhecapModel`, probably returned from `phecap_train_phenotyping_model`.

Value

A `data.frame` with two columns:

`patient_index` patient identifier

,

`prediction` predicted phenotype

.

See Also

See `PheCAP-package` for code examples.

`phecap_run_feature_extraction` *Run Surrogate-Assisted Feature Extraction (SAFE)*

Description

Run surrogate-assisted feature extraction (SAFE) using unlabeled data and subsampling.

Usage

```
phecap_run_feature_extraction(
  data, surrogates,
  subsample_size = 1000L, num_subsamples = 200L,
  dropout_proportion = 0, frequency_cutoff = 0.5,
  start_seed = 45600L, verbose = 0L)
```

Arguments

`data`	An object of class PhecapData, obtained by calling PhecapData(...)
`surrogates`	A list of objects of class PhecapSurrogate, obtained by something like list(PhecapSurrogate(...), PhecapSurrogate(...))
`subsample_size`	An integer scalar giving the size of each subsample
`num_subsamples`	The number of subsamples drawn for each surrogate
`dropout_proportion`	A scalar between 0 and 1. If it is positive, for each predictor a random subset of observations will be set to zero
`frequency_cutoff`	A scalar between 0 and 1. Variables selected in at least this proportion of the subsamples are the variables finally selected
`start_seed`	in the i-th subsample, the seed is set to start_seed + i
`verbose`	print progress every `verbose` subsample if `verbose` is positive, or remain quiet if `verbose` is zero

Details

In this unlabeled setting, the extremes of each surrogate are used to define cases and controls. The variables selected are those selected in at least half (or the proportion specified) of the subsamples.

Value

An object of class `PhecapFeatureExtraction`, with components

`selected`	the names of selected features
`frequency`	the proportion of being selected for each feature

See Also

See `PheCAP-package` for code examples.

`phecap_train_phenotyping_model` *Train Phenotyping Model using the Training Labels*

Description

Train the phenotyping model on the training dataset, and evaluate its performance via random splits of the training dataset.

Usage

```
phecap_train_phenotyping_model(
  data, surrogates, feature_selected,
  method = "lasso_bic",
  train_percent = 0.7, num_splits = 200L,
  start_seed = 78900L, verbose = 0L)
```

Arguments

data	an object of class PhecapData, obtained by calling PhecapData(...).
surrogates	a list of objects of class PhecapSurrogate, obtained by something like list(PhecapSurrogate(...), PhecapSurrogate(...)). The surrogates used here might be different from that used in feature extraction.
feature_selected	a character vector of the features that should be included in the model, probably returned by phecap_run_feature_extraction (but not necessary). The features listed here might be different from those returned from feature extraction.
method	Either a character vector or a list of two components. If a character vector is used, possible entries are given below. When at least two methods are specified, the predicted probability is the simple average of the predicted probabilities from each method. • 'plain' (logistic regression without penalty) • 'ridge_cv' (logistic regression with ridge penalty and CV tuning) • 'lasso_cv' (logistic regression with lasso penalty and CV tuning) • 'lasso_bic' (logistic regression with lasso penalty and BIC tuning) • 'alasso_cv' (logistic regression with adaptive lasso penalty and CV tuning) • 'alasso_bic' (logistic regression with adaptive lasso penalty and BIC tuning) • 'svm' (support vector machine with CV tuning, package e1071 needed, subject_weight not supported) • 'rf' (random forest with default parameters, package randomForestSRC needed) • 'xgb' (extreme gradient boosting with default parameters, package xgboost needed) If a list is used, it should contain two named components as follows. • fit (a function for model fitting, with arguments x, y, subject_weight, penalty_weight) • predict (a function for prediction, with arguments object which was returned by fit, x which was used as the new data to predict on)
train_percent	The percentage (between 0 and 1) of labels that are used for model training during random splits
num_splits	The number of random splits.
start_seed	in the i-th split, the seed is set to start_seed + i.
verbose	print progress every verbose splits if verbose is positive, or remain quiet if verbose is zero

Value

An object of class `PhecapModel`, with components

`coefficients`	the fitted object
`method`	the method used for model training
`feature_selected`	the feature selected by SAFE
`train_roc`	ROC on training dataset
`train_auc`	AUC on training dataset
`split_roc`	average ROC on random splits of training dataset
`split_auc`	average AUC on random splits of training dataset
`fit_function`	the function used for fitting
`predict_function`	the function used for prediction

See Also

See `PheCAP-package` for code examples.

`phecap_validate_phenotyping_model` *Validate the Phenotyping Model using the Validation Labels*

Description

Apply the trained model to all patients in the validation dataset, and measure the prediction accuracy via ROC and AUC.

Usage

```
phecap_validate_phenotyping_model(data, model)
```

Arguments

`data`	an object of class `PhecapData`, obtained by calling `PhecapData(...)`
`model`	an object of class `PhecapModel`, obtained by calling `phecap_train_phenotyping_model`.

Value

An object of class `PhecapValidation`, with components

`method`	the method used for model training
`train_roc`	ROC on training dataset
`train_auc`	AUC on training dataset

split_roc	average ROC on random splits of training dataset
split_auc	average AUC on random splits of training dataset
valid_roc	ROC on validation dataset
valid_auc	AUC on validation dataset

See Also

See PheCAP-package for code examples.

Index