Package 'csmpv'

May 8, 2026

Type Package

Title Biomarker Confirmation, Selection, Modelling, Prediction, and Validation

Version 1.0.5

Maintainer Aixiang Jiang <aijiang@bccrc.ca>

Depends R (>= 4.4.0), stats

Imports survival, glmnet, Hmisc, rms, forestmodel, ggplot2, ggpubr, survminer, xgboost, scales, Matrix

Suggests knitr, rmarkdown, devtools

VignetteBuilder knitr

Description There are diverse purposes such as biomarker confirmation, novel biomarker discovery, constructing predictive models, model-based prediction, and validation.
It handles binary, continuous, and time-to-event outcomes at the sample or patient level.
- Biomarker confirmation utilizes established functions like glm() from 'stats', coxph() from 'survival', surv_fit(), and ggsurvplot() from 'survminer'.
- Biomarker discovery and variable selection are facilitated by three LASSO-related functions LASSO2(), LASSO_plus(), and LASSO2plus(), leveraging the 'glmnet' R package with additional steps.
- Eight versatile modeling functions are offered, each designed for predictive models across various outcomes and data types.
1) LASSO2(), LASSO_plus(), LASSO2plus(), and LASSO2_reg() perform variable selection using LASSO methods and construct predictive models based on selected variables.
2) XGBtraining() employs 'XGBoost' for model building and is the only function not involving variable selection.
3) Functions like LASSO2_XGBtraining(), LASSO-plus_XGBtraining(), and LASSO2plus_XGBtraining() combine LASSO-related variable selection with 'XGBoost' for model construction.
- All models support prediction and validation, requiring a testing dataset comparable to the training dataset.
Additionally, the package introduces XGpred() for risk prediction based on survival data, with the XGpred_predict() function available for predicting risk groups in new datasets.
The methodology is based on our new algorithms and various references:
- Hastie et al. (1992, ISBN 0 534 16765-9),
- Therneau et al. (2000, ISBN 0-387-98784-3),

- Kassambara et al. (2021) <https://CRAN.R-project.org/package=survminer>,
- Friedman et al. (2010) <doi:10.18637/jss.v033.i01>,
- Simon et al. (2011) <doi:10.18637/jss.v039.i05>,
- Harrell (2023) <https://CRAN.R-project.org/package=rms>,
- Harrell (2023) <https://CRAN.R-project.org/package=Hmisc>,
- Chen and Guestrin (2016) <doi:10.48550/arXiv.1603.02754>,
- Aoki et al. (2023) <doi:10.1200/JCO.23.01115>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Author Aixiang Jiang [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0002-6153-7595>)

Repository CRAN

Date/Publication 2025-12-12 00:20:02 UTC

Contents

confirmVars *Biomarker Confirmation Function*

Description

This function confirms and validates known biomarkers in a given dataset.

Usage

```
confirmVars(
  data = NULL,
  standardization = FALSE,
  columnWise = TRUE,
  biomks = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  Y = NULL,
  time = NULL,
  event = NULL,
  outfile = "nameWithPath",
  timeUnits = "years"
)
```

Arguments

data	A data matrix or data frame with samples in rows and features/traits (including outcome and biomarkers) in columns.
standardization	Logical; indicates if standardization is needed before biomarker confirmation/validation. Default is FALSE.
columnWise	Logical; indicates if column-wise or row-wise normalization is needed for standardization. Default is TRUE.
biomks	A vector of biomarker names to confirm/validate. Subset of column names in the data parameter.
outcomeType	The type of the outcome variable. It has three choices: "binary" (default), "continuous", and "time-to-event".
Y	The outcome variable name when the outcome type is either "binary" or "continuous".
time	The time variable name when the outcome type is "time-to-event".
event	The event variable name when the outcome type is "time-to-event".
outfile	A string representing the output file, including the path if necessary, but without the file type extension.
timeUnits	A character vector specifying the units in which time is measured for the survival data. Default is "years".

Details

Use this function to assess whether individual variables or groups of variables have an impact on an outcome variable within a dataset. The outcome variable can be binary, continuous, or time-to-event. Note that this function is not intended for model confirmation, as it doesn't incorporate coefficients from previous research.

Value

A list containing:

`fit`	A model with selected variables for the given outcome variable.
`allplot`	A list of plots generated during the confirmation/validation process.

There might be extra plots in the list for time-to-event outcome

Author(s)

Aixiang Jiang

References

Hastie, T. J. and Pregibon, D. (1992) Generalized linear models. Chapter 6 of Statistical Models in S eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

Therneau, T., Grambsch, P., Modeling Survival Data: Extending the Cox Model. Springer-Verlag, 2000.

Kassambara A, Kosinski M, Biecek P (2021). survminer: Drawing Survival Curves using 'ggplot2', R package version 0.4.9, <https://CRAN.R-project.org/package=survminer>.

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training

# The confirmVars function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()

# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI","B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")

# confirmVars can work with three different outcome types.
# Here, we use binary as an example:
bconfirm = confirmVars(data = tdat, biomks = Xvars, Y = "DZsig",
                       outfile = paste0(temp_dir, "/confirmBinary"))
# You might save the files to the directory you want.

# To delete the temp_dir, use the following:
unlink(temp_dir)
```

csmpvModelling *All-in-one Modelling with csmpv R package*

Description

This function is designed to simplify the process of building, evaluating and comparing different modelling methods. It offers the flexibility to perform one or all of the following modelling methods: LASSO2, LASSO2 + regression, LASSO_plus, LASSO2plus, XGBoost, LASSO2 + XGBoost, LASSO_plus + XGBoost, and LASSO2plus + XGBoost. The models are trained on the training data, and their performance is validated on a separate validation dataset.

Usage

```
csmpvModelling(
  tdat = NULL,
  vdat = NULL,
  Ybinary = NULL,
  varsBinary = NULL,
  Ycont = NULL,
  varsCont = NULL,
  time = NULL,
  event = NULL,
  varsSurvival = NULL,
 methods = c("all", "LASSO2", "LASSO2_reg", "LASSO_plus", "LASSO2plus", "XGBoost",
    "LASSO2_XGBoost", "LASSO_plus_XGBoost", "LASSO2plus_XGBoost"),
  outfileName = NULL
)
```

Arguments

tdat	Training data. It can not be null.
vdat	Validation data. It should contain the same variables as in the training data, including outcome variables. No validation result is saved if it is NULL.
Ybinary	Binary outcome variable for classification.
varsBinary	Names of binary predictors.
Ycont	Continuous outcome variable for regression.
varsCont	Names of continuous predictors.
time	Time-to-event variable for survival analysis.
event	Event/censoring indicator for survival analysis.
varsSurvival	Names of predictors for survival analysis.
methods	Method(s) to use for modeling. If "all," models for all eight methods will be built. Otherwise, provide one of the following method names: - "LASSO2": Variable selection using LASSO2 with a minimum of two remaining variables.

- "LASSO2_reg": Variables selected from LASSO2, followed by regular regression. - "LASSO_plus": Variables selected from LASSO_plus, followed by regular regression. - "LASSO2plus": Variables selected from LASSO2plus, followed by regular regression. - "XGBoost": XGBoost model built without variable selection. - "LASSO2_XGBoost": Variables selected from LASSO2, followed by XGBoost. - "LASSO_plus_XGBoost": Variables selected from LASSO_plus, followed by XGBoost. - "LASSO2plus_XGBoost": Variables selected from LASSO2plus, followed by XGBoost.

`outfileName` Prefix for output file names.

Details

By default, this function runs all eight different modeling methods. However, users can specify the "methods" parameter to choose and run a specific modelling method of their choice. For clarity, when providing a 'vdat' argument, the function assumes that it contains the outcome variable, and it proceeds with model validation.

Value

A list of trained models and prediction objects. Results are saved to local files.

Author(s)

Aixiang Jiang

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training
vdat = datlist$validation

# The confirmVars function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()

# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# The default setting of this single function generates all models and provides predictions
# and validations for each of them.
# Of course, we can also use this all-in-one function to work on one outcome type
# and one model at a time, for example:
DZlassoreg = csmpvModelling(tdat = tdat, vdat = vdat,
                            Ybinary = "DZsig", varsBinary = Xvars,
                            methods = "LASSO2_reg",
                            outfileName= paste0(temp_dir, "/just_one"))
# This is equivalent to using LASSO2_reg for modeling, followed by prediction and validation
# with rms_model for the classification task "DZsig".
# Six result files are then saved locally.
# You might want to save the files to the directory you prefer.
```

```
# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

`datlist` *This is an example data in csmpv*

Description

This dataset contains sample data for csmpv.

Usage

```
datlist
```

Format

A list with training and a validation datasets.

Author(s)

Aixiang Jiang

`LASSO2` *Variable Selection using Modified LASSO with a Minimum of Two Remaining Variables*

Description

This function conducts variable selection using LASSO (Least Absolute Shrinkage and Selection Operator) with a minor adaptation. It calculates the mean lambda value from multiple cv.glmnet runs and ensures the selection of at least two variables.

Usage

```
LASSO2(
  data = NULL,
  standardization = FALSE,
  columnWise = TRUE,
  biomks = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  Y = NULL,
  time = NULL,
  event = NULL,
  nfolds = 10,
  outfile = "nameWithPath"
)
```

Arguments

data	A data matrix or a data frame where samples are in rows and features/traits are in columns.
standardization	A logical variable indicating if standardization is needed before variable selection. The default is FALSE.
columnWise	A logical variable indicating if column-wise or row-wise normalization is needed. The default is TRUE, which means column-wise normalization is performed. This is only meaningful when "standardization" is TRUE.
biomks	A vector of potential biomarkers for variable selection. They should be a subset of the column names in the "data" variable.
outcomeType	The outcome variable type. There are three choices: "binary" (default), "continuous", and "time-to-event".
Y	The outcome variable name when the outcome type is either "binary" or "continuous".
time	The time variable name when the outcome type is "time-to-event".
event	The event variable name when the outcome type is "time-to-event".
nfolds	The number of folds for cross-validation. The default is 10.
outfile	A string representing the output file, including the path if necessary, but without the file type extension.

Details

The function utilizes glmnet::cv.glmnet for cross-validation-based variable selection with the largest value of lambda such that error is within 1 standard error of the minimum. To mitigate randomness from cross-validation splits, it conducts 10 runs (this number can later be parameterized) of n-fold cv.glmnet. The resulting average lambda value across these runs serves as the final lambda. Subsequently, the final regularization regression is performed on the complete dataset using this mean lambda value. Following this, the function assesses the count of remaining variables. If only one or none are selected, the function defaults to selecting the first lambda that results in at least two chosen variables on the full dataset. This function is designed to handle three types of outcome variables: continuous, binary, and time-to-event.

Value

A list is returned:

coefs	A vector of LASSO coefficients
h0	Cumulative baseline hazard table, for time to event outcome only
Y	The outcome variable name when the outcome type is either "binary" or "continuous".
time	The time variable name when the outcome type is "time-to-event".
event	The event variable name when the outcome type is "time-to-event".
standardization	A logical variable indicating if standardization is needed before variable selection.

columnWise	A logical variable indicating if column-wise or row-wise normalization is needed.
outcomeType	The outcome variable type.
allplot	A plot object

A shrunken coefficient vector is returned

Author(s)

Aixiang Jiang

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) Regularization Paths for Generalized Linear Models via Coordinate Descent (2010), Journal of Statistical Software, Vol. 33(1), 1-22, doi:10.18637/jss.v033.i01.

Simon, N., Friedman, J., Hastie, T. and Tibshirani, R. (2011) Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent, Journal of Statistical Software, Vol. 39(5), 1-13, doi:10.18637/jss.v039.i05.

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# The function can work with three different outcome types.
# Here, we use time-to-event as an example:
# tl = LASSO2(data = tdat, biomks = Xvars,
#             outcomeType = "time-to-event",
#             time = "FFP..Years.",event = "Code.FFP",
#             outfile = paste0(temp_dir, "/survivalLASSO2"))
# You might save the files to the directory you want.

# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

LASSO2plus *Variable Selection and Modeling with LASSO2plus*

Description

This function performs variable selection using the LASSO2plus algorithm and subsequently builds a model.

Usage

```
LASSO2plus(
  data = NULL,
  standardization = FALSE,
  columnWise = TRUE,
  biomks = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  Y = NULL,
  time = NULL,
  event = NULL,
  outfile = "nameWithPath",
  height = 6
)
```

Arguments

data	A data matrix or a data frame, samples are in rows, and features/traits are in columns.
standardization	A logic variable to indicate if standardization is needed before variable selection, the default is FALSE.
columnWise	A logic variable to indicate if column wise or row wise normalization is needed, the default is TRUE, which is to do column-wise normalization. This is only meaningful when "standardization" is TRUE.
biomks	A vector of potential biomarkers for variable selection, they should be a subset of "data" column names.
outcomeType	Outcome variable type. There are three choices: "binary" (default), "continuous", and "time-to-event".
Y	Outcome variable name when the outcome type is either "binary" or "continuous".
time	Time variable name when outcome type is "time-to-event".
event	Event variable name when outcome type is "time-to-event".
outfile	A string for the output file including path if necessary but without file type extension.
height	An integer to indicate the forest plot height in inches

Details

The LASSO2plus algorithm begins with variable selection using LASSO2, typically involving multiple cross-validation-based LASSO regressions. However, if only one or no variables are selected, the cross-validation results are ignored, and the algorithm ensures a minimum of two remaining variables through full-data lambda simulations. Additionally, it conducts variable selection through single-variable regression for each candidate variable. The variables selected from both LASSO2 and single-variable approaches are then combined to perform traditional variable selection using stepwise regression. This function is designed to handle outcome variables of binary, continuous, or time-to-event type. Following variable selection, a model is constructed using standard R functions such as lm, glm, or coxph, depending on the type of outcome variable.

Value

A list is returned:

`fit`	A model with selected variables for the given outcome variable
`outplot`	A forest plot

Author(s)

Aixiang Jiang

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) Regularization Paths for Generalized Linear Models via Coordinate Descent (2010), Journal of Statistical Software, Vol. 33(1), 1-22, doi:10.18637/jss.v033.i01.

Simon, N., Friedman, J., Hastie, T. and Tibshirani, R. (2011) Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent, Journal of Statistical Software, Vol. 39(5), 1-13, doi:10.18637/jss.v039.i05.

Hastie, T. J. and Pregibon, D. (1992) Generalized linear models. Chapter 6 of Statistical Models in S eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

Therneau, T., Grambsch, P., Modeling Survival Data: Extending the Cox Model. Springer-Verlag, 2000.

Kassambara A, Kosinski M, Biecek P (2021). survminer: Drawing Survival Curves using 'ggplot2'_. R package version 0.4.9, <https://CRAN.R-project.org/package=survminer>.

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# The function can work with three different outcome types.
# Here, we use continuous as an example:
c2fit = LASSO2plus(data = tdat, biomks = Xvars,
                   outcomeType = "continuous", Y = "Age",
                   outfile = paste0(temp_dir, "/continuousLASSO2plus"))
# You might save the files to the directory you want.

# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

LASSO2plus_XGBtraining *XGBoost Modeling after Variable Selection with LASSO2plus*

Description

This function performs variable selection using LASSO2plus and then builds an XGBoost model.

Usage

```
LASSO2plus_XGBtraining(
  data = NULL,
  standardization = FALSE,
  columnWise = TRUE,
  biomks = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  Y = NULL,
  time = NULL,
  event = NULL,
  nrounds = 5,
  nthread = 2,
  gamma = 1,
  max_depth = 3,
  eta = 0.3,
  outfile = "nameWithPath",
  height = 6
)
```

Arguments

data	A data matrix or a data frame where samples are in rows, and features/traits are in columns.
standardization	A logical variable to indicate if standardization is needed before variable selection. The default is FALSE.
columnWise	A logical variable indicating whether column-wise or row-wise normalization is needed. The default is TRUE, which is used to perform column-wise normalization. This is only meaningful when "standardization" is TRUE.
biomks	A vector of potential biomarkers for variable selection. They should be a subset of "data" column names.
outcomeType	Outcome variable type. There are three choices: "binary" (default), "continuous", and "time-to-event".
Y	Outcome variable name when the outcome type is either "binary" or "continuous".
time	Time variable name when outcome type is "time-to-event".

event	Event variable name when outcome type is "time-to-event".
nrounds	Max number of boosting iterations.
nthread	Number of parallel threads used to run XGBoost.
gamma	Minimum loss reduction required to make a further partition on a leaf node of the tree.
max_depth	Maximum depth of a tree. Increasing this value will make the model more complex and more likely to overfit.
eta	The learning rate for the XGBoost model.
outfile	A string for the output file including path if necessary but without file type extension.
height	An integer to indicate the forest plot height in inches.

Details

The first part of LASSO2plus_XGBtraining involves variable selection with LASSO2plus. The LASSO2plus algorithm begins with variable selection using LASSO2, followed by variable selection through single-variable regression for each candidate variable. Finally, the two sets of selected variables are combined and processed to obtain the final list through stepwise variable selection. The second part of LASSO2plus_XGBtraining involves using the final variable list obtained above to build an XGBoost model. It is suitable for three types of outcomes: continuous, binary, and time-to-event.

Value

A list is returned:

XGBoost_model	An XGBoost model
XGBoost_model_score	Model scores for the given training data set. For a continuous outcome variable, this is a vector of the estimated continuous values; for a binary outcome variable, this is a vector representing the probability of the positive class; for time-to-event outcome, this is a vector of risk scores

Author(s)

Aixiang Jiang

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) Regularization Paths for Generalized Linear Models via Coordinate Descent (2010), Journal of Statistical Software, Vol. 33(1), 1-22, doi:10.18637/jss.v033.i01.

Simon, N., Friedman, J., Hastie, T. and Tibshirani, R. (2011) Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent, Journal of Statistical Software, Vol. 39(5), 1-13, doi:10.18637/jss.v039.i05.

Tianqi Chen and Carlos Guestrin, "XGBoost: A Scalable Tree Boosting System", 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016, https://arxiv.org/abs/1603.02754

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# The function can work with three different outcome types.
# Here, we use time-to-event as an example:
# tl2xfit = LASSO2plus_XGBtraining(data = tdat, biomks = Xvars,
#                                  outcomeType = "time-to-event",
#                                 time = "FFP..Years.", event = "Code.FFP",
#                               outfile = paste0(temp_dir, "/survival_LASSO2plus_XGBoost"))
#
# You might save the files to the directory you want.

# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

LASSO2_predict *Predict and Validate LASSO2 Model Scores*

Description

This function predicts model scores, positive group probabilities, and risk scores for different outcome types based on a LASSO2 object and a new data set. It also performs validation of predictions when the true outcome variable is available.

Usage

```
LASSO2_predict(
  lassoObj,
  newdata = NULL,
  newY = FALSE,
  u = 2,
  outfile = "nameWithPath"
)
```

Arguments

lassoObj	A LASSO2 object.
newdata	A new data matrix or data frame where samples are in rows and features/traits are in columns.
newY	The outcome variable for the new data set.

u	A single numeric follow-up time for survival outcomes.
outfile	A string representing the output file, including the path if necessary, but without the file type extension.

Value

A vector of predicted values is returned.

Author(s)

Aixiang Jiang

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) Regularization Paths for Generalized Linear Models via Coordinate Descent (2010), Journal of Statistical Software, Vol. 33(1), 1-22, doi:10.18637/jss.v033.i01.

Simon, N., Friedman, J., Hastie, T. and Tibshirani, R. (2011) Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent, Journal of Statistical Software, Vol. 39(5), 1-13, doi:10.18637/jss.v039.i05.

Harrell Jr F (2023). rms: Regression Modeling Strategies_. R package version 6.7-1, <https://CRAN.R-project.org/package=rms>

Harrell Jr F (2023). Hmisc: Harrell Miscellaneous_. R package version 5.1-1, <https://CRAN.R-project.org/package=Hmisc>

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training
vdat = datlist$validation

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# The function can work with three different outcome types.
# Here, we use time-to-event as an example:
# tl = LASSO2(data = tdat, biomks = Xvars,
#             outcomeType = "time-to-event",
#             time = "FFP..Years.",event = "Code.FFP",
#             outfile = paste0(temp_dir, "/survivalLASSO2"))
# To predict the model in a new data set:
# ptl = LASSO2_predict(tl, newdata = vdat,
#                      outfile = paste0(temp_dir, "/pred_LASSO2_time_to_event"))
# You might save the files to the directory you want.

# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

LASSO2_reg *LASSO2 Variable Selection and Regular Regression Modeling*

Description

This function performs variable selection with LASSO2 but ignores the shrunk LASSO coefficients and builds a regular regression model.

Usage

```
LASSO2_reg(
  data = NULL,
  standardization = FALSE,
  columnWise = TRUE,
  biomks = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  Y = NULL,
  time = NULL,
  event = NULL,
  nfolds = 10,
  outfile = "nameWithPath"
)
```

Arguments

data	A data matrix or a data frame where samples are in rows, and features/traits are in columns.
standardization	A logical variable to indicate if standardization is needed before variable selection. The default is FALSE.
columnWise	A logical variable to indicate if column-wise or row-wise normalization is needed. The default is TRUE, which performs column-wise normalization. This is only meaningful when "standardization" is TRUE.
biomks	A vector of potential biomarkers for variable selection. They should be a subset of column names in "data".
outcomeType	The outcome variable type. There are three choices: "binary" (default), "continuous", and "time-to-event".
Y	The outcome variable name when the outcome type is either "binary" or "continuous".
time	The time variable name when the outcome type is "time-to-event".
event	The event variable name when the outcome type is "time-to-event".
nfolds	The number of folds for cross-validation. The default value is 10.
outfile	A string for the output file including the path, if necessary, but without the file type extension.

Details

The first part of LASSO2_reg involves variable selection with LASSO2, typically based on the mean lambda.1se from 10 iterations of n-fold cross-validation-based LASSO regression. In each iteration, a lambda.1se refers to the largest value of lambda such that the error is within 1 standard error of the minimum. However, if only one or no variable is selected, the cross-validation results are ignored, and a minimum of two remaining variables is ensured through full-data lambda simulations. The second part of LASSO2_reg involves ignoring the shrunk LASSO coefficients and building a regular regression model. It is suitable for three types of outcomes: continuous, binary, and time-to-event.

Value

A list is returned with the same output as from confirmVars.

`fit`	A model with selected variables for the given outcome variable.
`allplot`	A list with all plots.

Author(s)

Aixiang Jiang

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) Regularization Paths for Generalized Linear Models via Coordinate Descent (2010), Journal of Statistical Software, Vol. 33(1), 1-22, doi:10.18637/jss.v033.i01.

Simon, N., Friedman, J., Hastie, T. and Tibshirani, R. (2011) Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent, Journal of Statistical Software, Vol. 39(5), 1-13, doi:10.18637/jss.v039.i05.

Hastie, T. J. and Pregibon, D. (1992) Generalized linear models. Chapter 6 of Statistical Models in S eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

Therneau, T., Grambsch, P., Modeling Survival Data: Extending the Cox Model. Springer-Verlag, 2000.

Kassambara A, Kosinski M, Biecek P (2021). survminer: Drawing Survival Curves using 'ggplot2'_. R package version 0.4.9,

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")

# The function can work with three different outcome types.
# Here, we use time-to-event as an example:
```

```
# tlr = LASSO2_reg(data = tdat, biomks = Xvars,
#                  outcomeType = "time-to-event",
#                  time = "FFP..Years.",event = "Code.FFP",
#                  outfile = paste0(temp_dir, "/survivalLASSO2_reg"))
# You might save the files to the directory you want.
# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

LASSO2_XGBtraining *Variable Selection with LASSO2 and Modeling with XGBoost*

Description

This function performs a two-step process: variable selection using LASSO2 and building a predictive model using XGBoost.

Usage

```
LASSO2_XGBtraining(
  data = NULL,
  standardization = FALSE,
  columnWise = TRUE,
  biomks = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  Y = NULL,
  time = NULL,
  event = NULL,
  nfolds = 10,
  nrounds = 5,
  nthread = 2,
  gamma = 1,
  max_depth = 3,
  eta = 0.3,
  outfile = "nameWithPath"
)
```

Arguments

data	A data matrix or data frame containing samples in rows and features/traits in columns.
standardization	A logical value indicating if standardization is needed before variable selection. Default is FALSE.
columnWise	A logical value indicating if column-wise or row-wise normalization is needed for standardization. Default is TRUE. This parameter is only meaningful when standardization is TRUE.

biomks	A vector of potential biomarkers for variable selection. These should be a subset of the column names in the data parameter.
outcomeType	The type of the outcome variable: "binary" (default), "continuous", or "time-to-event".
Y	The name of the outcome variable when the outcome type is either "binary" or "continuous".
time	The name of the time variable when the outcome type is "time-to-event".
event	The name of the event variable when the outcome type is "time-to-event".
nfolds	The number of folds for cross-validation. The default is 10.
nrounds	The maximum number of boosting iterations for the XGBoost model.
nthread	The number of parallel threads used for running XGBoost.
gamma	The minimum loss reduction required to make a further partition on a leaf node of the tree.
max_depth	The maximum depth of a tree in the XGBoost model.
eta	The learning rate for the XGBoost model.
outfile	A string for the output file, including the path if necessary, but without the file type extension.

Details

The first part of LASSO2_XGBtraining involves variable selection with LASSO2, typically based on the mean lambda.1se from 10 iterations of n-fold cross-validation-based LASSO regression. In each iteration, a lambda.1se refers to the largest value of lambda such that the error is within 1 standard error of the minimum. However, if only one or no variable is selected, the cross-validation results are ignored, and a minimum of two remaining variables is ensured through full-data lambda simulations.

The second part of LASSO2_XGBtraining involves ignoring the shrunk LASSO coefficients and building an XGBoost model. It is suitable for three types of outcomes: continuous, binary, and time-to-event.

Value

A list is returned:

XGBoost_model	An XGBoost model
XGBoost_model_score	Model scores for the given training data set. For a continuous outcome variable, this is a vector of the estimated continuous values; for a binary outcome variable, this is a vector representing the probability of the positive class; for time-to-event outcome, this a vector of risk scores

Author(s)

Aixiang Jiang

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) Regularization Paths for Generalized Linear Models via Coordinate Descent (2010), Journal of Statistical Software, Vol. 33(1), 1-22, doi:10.18637/jss.v033.i01.

Simon, N., Friedman, J., Hastie, T. and Tibshirani, R. (2011) Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent, Journal of Statistical Software, Vol. 39(5), 1-13, doi:10.18637/jss.v039.i05.

Tianqi Chen and Carlos Guestrin, "XGBoost: A Scalable Tree Boosting System", 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016, https://arxiv.org/abs/1603.02754

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# The function can work with three different outcome types.
# Here, we use binary as an example:
blxfit = LASSO2_XGBtraining(data = tdat, biomks = Xvars, Y = "DZsig",
                            outfile = paste0(temp_dir, "/binary_LASSO2_XGBoost"))
# You might save the files to the directory you want.

# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

LASSO_plus *LASSO_plus Variable Selection and Modeling*

Description

This function performs variable selection using the LASSO_plus algorithm and builds a model afterward.

Usage

```
LASSO_plus(
  data = NULL,
  standardization = FALSE,
  columnWise = TRUE,
  biomks = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  Y = NULL,
  time = NULL,
```

```
  event = NULL,
  topN = 10,
  outfile = "nameWithPath",
  height = 6
)
```

Arguments

data	A data matrix or a data frame, samples are in rows, and features/traits are in columns.
standardization	A logic variable to indicate if standardization is needed before variable selection, the default is FALSE.
columnWise	A logic variable to indicate if column wise or row wise normalization is needed, the default is TRUE, which is to do column-wise normalization. This is only meaningful when "standardization" is TRUE.
biomks	A vector of potential biomarkers for variable selection, they should be a subset of "data" column names.
outcomeType	Outcome variable type. There are three choices: "binary" (default), "continuous", and "time-to-event".
Y	Outcome variable name when the outcome type is either "binary" or "continuous".
time	Time variable name when outcome type is "time-to-event".
event	Event variable name when outcome type is "time-to-event".
topN	An integer indicating the desired number of variables to be selected.
outfile	A string representing the output file, including the path if necessary, but without the file type extension
height	An integer to indicate the forest plot height in inches

Details

The LASSO_plus algorithm combines LASSO, single variable regression, and stepwise regression to select variables associated with an outcome variable in a given dataset. The outcome variable can be binary, continuous, or time-to-event. After variable selection, a model is built using common R functions such as lm, glm, and coxph, depending on the outcome type.

Value

A list is returned:

fit	A model with selected variables for the given outcome variable
outplot	A forest plot

Author(s)

Aixiang Jiang

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) Regularization Paths for Generalized Linear Models via Coordinate Descent (2010), Journal of Statistical Software, Vol. 33(1), 1-22, doi:10.18637/jss.v033.i01.

Simon, N., Friedman, J., Hastie, T. and Tibshirani, R. (2011) Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent, Journal of Statistical Software, Vol. 39(5), 1-13, doi:10.18637/jss.v039.i05.

Hastie, T. J. and Pregibon, D. (1992) Generalized linear models. Chapter 6 of Statistical Models in S eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

Therneau, T., Grambsch, P., Modeling Survival Data: Extending the Cox Model. Springer-Verlag, 2000.

Kassambara A, Kosinski M, Biecek P (2021). survminer: Drawing Survival Curves using 'ggplot2', R package version 0.4.9, <https://CRAN.R-project.org/package=survminer>.

Aoki T, Jiang A, Xu A et al.,(2023) Spatially Resolved Tumor Microenvironment Predicts Treatment Outcomes in Relapsed/Refractory Hodgkin Lymphoma. J Clin Oncol. 2023 Dec 19:JCO2301115. doi: 10.1200/JCO.23.01115. Epub ahead of print. PMID: 38113419.

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# The function can work with three different outcome types.
# Here, we use binary as an example:
bfit = LASSO_plus(data = tdat, biomks = Xvars, Y = "DZsig", topN = 5,
                  outfile = paste0(temp_dir, "/binaryLASSO_plus"))
# You might save the files to the directory you want.

# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

LASSO_plus_XGBtraining

LASSO_plus_XGBtraining: Variable Selection and XGBoost Modeling

---

Description

This function performs variable selection using LASSO_plus, followed by building an XGBoost model. LASSO_plus is a method for variable selection, and XGBoost is a gradient boosting algorithm for modeling.

Usage

```
LASSO_plus_XGBtraining(
  data = NULL,
  standardization = FALSE,
  columnWise = TRUE,
  biomks = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  Y = NULL,
  time = NULL,
  event = NULL,
  topN = 10,
  nrounds = 5,
  nthread = 2,
  gamma = 1,
  max_depth = 3,
  eta = 0.3,
  outfile = "nameWithPath",
  height = 6
)
```

Arguments

data	A data matrix or a data frame where samples are in rows, and features/traits are in columns.
standardization	A logical variable to indicate if standardization is needed before variable selection. The default is FALSE.
columnWise	A logical variable to indicate if column-wise or row-wise normalization is needed. The default is TRUE, which is to do column-wise normalization. This is only meaningful when "standardization" is TRUE.
biomks	A vector of potential biomarkers for variable selection. They should be a subset of "data" column names.
outcomeType	Outcome variable type. There are three choices: "binary" (default), "continuous", and "time-to-event".
Y	Outcome variable name when the outcome type is either "binary" or "continuous".
time	Time variable name when outcome type is "time-to-event".
event	Event variable name when outcome type is "time-to-event".
topN	An integer to indicate how many variables we intend to select.
nrounds	Max number of boosting iterations.
nthread	Number of parallel threads used to run XGBoost.
gamma	Minimum loss reduction required to make a further partition on a leaf node of the tree.
max_depth	Maximum depth of a tree. Increasing this value will make the model more complex and more likely to overfit.

eta	The learning rate for the XGBoost model.
outfile	A string representing the output file, including the path if necessary, but without the file type extension.
height	An integer to indicate the forest plot height in inches.

Value

A list is returned:

XGBoost_model	An XGBoost model
XGBoost_model_score	Model scores for the given training data set. For a continuous outcome variable, this is a vector of the estimated continuous values; for a binary outcome variable, this is a vector representing the probability of the positive class; for time-to-event outcome, this is a vector of risk scores

Author(s)

Aixiang Jiang

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) Regularization Paths for Generalized Linear Models via Coordinate Descent (2010), Journal of Statistical Software, Vol. 33(1), 1-22, doi:10.18637/jss.v033.i01.

Simon, N., Friedman, J., Hastie, T. and Tibshirani, R. (2011) Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent, Journal of Statistical Software, Vol. 39(5), 1-13, doi:10.18637/jss.v039.i05.

Tianqi Chen and Carlos Guestrin, "XGBoost: A Scalable Tree Boosting System", 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016, https://arxiv.org/abs/1603.02754

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# The function can work with three different outcome types.
# Here, we use continuous as an example:
clpxfit = LASSO_plus_XGBtraining(data = tdat, biomks = Xvars,
          outcomeType = "continuous", Y = "Age", topN = 5,
          outfile = paste0(temp_dir, "/continuous_LASSO_plus_XGBoost"))
# You might save the files to the directory you want.

# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

rms_model *A Wrapper for Building Predictive Models using the rms Package*

Description

This wrapper function harnesses the capabilities of the rms package to construct robust predictive models using various modeling techniques. Whether you're working with linear regression (lm), generalized linear models (glm), or Cox proportional hazards models (coxph) directly or generating these objects from other functions, this function streamlines the model-building process and enhances analysis. The function features built-in bootstrap-based internal validation, model score generation, and result storage.

Usage

```
rms_model(afit, data = NULL, newdata = NULL, newY = FALSE, u = 2, outfile)
```

Arguments

afit	A model fit from lm, glm, or coxph.
data	A data frame used to obtain the 'afit' object. This parameter is only required when the outcome variable is time-to-event; otherwise, the data is extracted from 'afit'.
newdata	A new data frame for prediction.
newY	A logical variable indicating whether 'newdata' contains the outcome variable.
u	A single numeric value representing follow-up time for survival outcomes, used to estimate the survival probability for the given time point.
outfile	A string indicating the output file name without the file type extension, but including the complete path information.

Details

The wrapper function capitalizes on the power of the rms package, providing a seamless interface for creating predictive models. With the rms package, the function conducts efficient bootstrap-based internal validation, enabling thorough model evaluation. Nomograph plots and C-index calculations are also at your disposal. Furthermore, the function adapts to your needs by predicting model scores for new datasets. In the absence of new data, it generates predictions based on the training data. However, when you provide an independent dataset complete with outcome variable information, the function extends to external validation. This enables assessing model performance in an independent context using the rms R package.

All results, from the model itself to internal validation metrics and predictions, are conveniently stored locally for easy access and further analysis.

Value

Predictions are returned.

Author(s)

Aixiang Jiang

References

Harrell Jr F (2023). rms: Regression Modeling Strategies. R package version 6.7-1, <https://CRAN.R-project.org/package=rms>

Harrell Jr F (2023). Hmisc: Harrell Miscellaneous. R package version 5.1-1, <https://CRAN.R-project.org/package=Hmisc>

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training
vdat = datlist$validation

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")

# The function can work with multiple models and multiple outcome types.
# Here, we use continuous as an example:
clr = LASSO2_reg(data = tdat, biomks = Xvars,
                 outcomeType = "continuous", Y = "Age",
                 outfile = paste0(temp_dir, "/continuousLASSO2_reg"))
pclr = rms_model(clr$fit, newdata = vdat,
                 outfile = paste0(temp_dir, "/pred_LASSO2reg_continuous"))
# You might save the files to the directory you want.

# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

---

validation *Validate Model Predictions*

Description

This function is designed to perform model validation when the corresponding outcome variable is available. It facilitates the comparison of model predictions with the provided outcome variable, which can be continuous, binary, or time-to-event.

Usage

```
validation(
  predicted = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  trueY = NULL,
  time = NULL,
  trueEvent = NULL,
  baseHz = NULL,
  u = 2,
  outfile = "nameWithPath"
)
```

Arguments

predicted	A vector of model prediction values, which can be generated by prediction functions in this package such as LASSO2_predict and XGBtraining_predict. For continuous outcomes, this vector represents model scores; for binary outcomes, it represents the probability of the positive class; for time-to-event outcomes, it contains risk scores, which will later be transformed into estimated survival probabilities corresponding to times in the new data.
outcomeType	Outcome variable type. There are three choices: "binary" (default), "continuous", and "time-to-event".
trueY	A vector of the outcome variable when it is continuous or binary.
time	A vector of time for time-to-event outcome.
trueEvent	A vector of the event for time-to-event outcome.
baseHz	A table for accumulating baseline hazard for multiple time points, usually generated based on a training data set.
u	A single numeric follow-up time for survival outcomes.
outfile	A string for the output file, including the path if necessary but without the file type extension.

Details

This function is invoked by multiple prediction functions within this package when an outcome variable is available for a new dataset. However, users can also directly call this function if needed.

Value

A vector of model prediction values from the input

References

Harrell Jr F (2023). rms: Regression Modeling Strategies_. R package version 6.7-1, <https://CRAN.R-project.org/package=rms>

Harrell Jr F (2023). Hmisc: Harrell Miscellaneous_. R package version 5.1-1, <https://CRAN.R-project.org/package=Hmisc>

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training
vdat = datlist$validation

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")

# The function can work with multiple models and multiple outcome types.
# Here, we use XGBoost model with binary outcome as an example:
bxfit = XGBtraining(data = tdat, biomks = Xvars, Y = "DZsig",
                    outfile = paste0(temp_dir, "/binary_XGBoost"))
pbxfit = XGBtraining_predict(bxfit, newdata = vdat,
                    outfile = paste0(temp_dir, "/pred_binary_XGBoost"))
Y = bxfit$Y
outs = validation(predicted = pbxfit, outcomeType = "binary", trueY = vdat[,Y],
                  outfile = paste0(temp_dir, "/binary_XGBoost_validate"))
# You might save the files to the directory you want.

# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

`XGBtraining` *A Wrapper Function for xgboost::xgboost*

Description

This wrapper function streamlines the process of utilizing the xgboost package for model training. It takes care of converting the data format to xgb.DMatrix, handling xgboost's specific settings, and invoking xgboost::xgboost. The function is suitable for all three outcome types: binary, continuous, and time-to-event. It returns both the trained model and the model scores for the training dataset.

It's important to note that all independent variables (X variables) should already be selected and in numeric format when passed to this function. Additionally, this function does not perform variable selection or automatically convert categorical variables to numeric format.

Usage

```
XGBtraining(
  data,
  biomks = NULL,
  outcomeType = c("binary", "continuous", "time-to-event"),
  Y = NULL,
  time = NULL,
  event = NULL,
```

```
  nrounds = 5,
  nthread = 2,
  gamma = 1,
  max_depth = 3,
  eta = 0.3,
  outfile = "nameWithPath"
)
```

Arguments

data	A data matrix or a data frame where samples are in rows and features/traits are in columns.
biomks	A vector of potential biomarkers for variable selection. They should be a subset of the column names in the "data" variable.
outcomeType	The outcome variable type. There are three choices: "binary" (default), "continuous", and "time-to-event".
Y	The outcome variable name when the outcome type is either "binary" or "continuous". When Y is binary, it should be in 0-1 format.
time	The time variable name when the outcome type is "time-to-event".
event	The event variable name when the outcome type is "time-to-event".
nrounds	The maximum number of boosting iterations.
nthread	The number of parallel threads used to run XGBoost.
gamma	The minimum loss reduction required to make a further partition on a leaf node of the tree.
max_depth	The maximum depth of a tree. Increasing this value will make the model more complex and more likely to overfit.
eta	The step size shrinkage used in the update to prevent overfitting.
outfile	A string for the output file, including the path if necessary but without the file type extension.

Value

A list is returned:

XGBoost_model	An XGBoost model
XGBoost_score	Scores for the given training data set. For a continuous outcome variable, this is a vector of the estimated continuous values; for a binary outcome variable, this is a vector representing the probability of the positive class; for a time-to-event outcome, this is a vector of risk scores.
h0	Cumulative baseline hazard table, for time to event outcome only.
Y	The outcome variable name when the outcome type is either "binary" or "continuous".
time	The time variable name when the outcome type is "time-to-event".
event	The event variable name when the outcome type is "time-to-event".

Author(s)

Aixiang Jiang

References

Tianqi Chen and Carlos Guestrin, "XGBoost: A Scalable Tree Boosting System", 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016, https://arxiv.org/abs/1603.02754

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training
# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")

# The function can work with three outcome types.
# Here, we use time-to-event outcome as an example:
txfit = XGBtraining(data = tdat, biomks = Xvars,
                    outcomeType = "time-to-event",
                    time = "FFP..Years.",event = "Code.FFP",
                    outfile = paste0(temp_dir, "/survival_XGBoost"))
# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

XGBtraining_predict *Predicting XGBoost Model Scores and Performing Validation*

Description

This function predicts XGBoost model scores using an XGBtraining object and a new dataset. It converts the input data to the required xgb.DMatrix format and returns the model scores. If the new dataset includes an outcome variable, the function also performs validation, comparing predictions with observed outcomes.

Usage

```
XGBtraining_predict(
  xgbtrainingObj = NULL,
  newdata = NULL,
  newY = FALSE,
  outfile = "nameWithPath"
)
```

Arguments

xgbtrainingObj	An XGBtraining object returned from the XGBtraining function.
newdata	A data matrix or a data frame, samples are in rows, and features/traits are in columns.
newY	A logical variable indicating if 'newdata' contains the outcome variable.
outfile	A string for the output file including path if necessary but without file type extension.

Value

A vector of predicted values is return. If an outcome variable is available for the new dataset, validation is performed. For continuous outcome, this is a vector of model scores. For binary outcome, this is a vector representing the probability of the positive class. For time to event outcome, this is a vector of risk scores.

Author(s)

Aixiang Jiang

References

Tianqi Chen and Carlos Guestrin, "XGBoost: A Scalable Tree Boosting System", 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016, https://arxiv.org/abs/1603.02754

Harrell Jr F (2023). rms: Regression Modeling Strategies_. R package version 6.7-1, <https://CRAN.R-project.org/package=rms>

Harrell Jr F (2023). Hmisc: Harrell Miscellaneous_. R package version 5.1-1, <https://CRAN.R-project.org/package=Hmisc>

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training
vdat = datlist$validation

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")

# The function can work with multiple models and multiple outcome types.
# Here, we provide an example using the XGBoost model with a time-to-event outcome:
txfit = XGBtraining(data = tdat, biomks = Xvars,
                    outcomeType = "time-to-event",
                    time = "FFP..Years.",event = "Code.FFP",
                    outfile = paste0(temp_dir, "/survival_XGBoost"))
ptxfit = XGBtraining_predict(txfit, newdata = vdat,
                    outfile = paste0(temp_dir, "/pred_XGBoost_time_to_event"))
```

```
# To delete the "temp_dir", use the following:
unlink(temp_dir)
```

XGpred *XGpred: Building Risk Classification Predictive Models using Survival Data*

Description

The XGpred function is designed to generate an empirical Bayesian-based binary risk classification model with survival data based on our novel XGpred algorithm, combining XGBoost and traditional survival analysis.

Usage

```
XGpred(
  data = NULL,
  varsIn = NULL,
  selection = FALSE,
  vsMethod = c("LASSO2", "LASSO2plus", "LASSO_plus"),
  time = NULL,
  event = NULL,
  nrounds = 5,
  probcut = 0.8,
  nthread = 2,
  gamma = 1,
  max_depth = 3,
  eta = 0.3,
  nclass = c(2, 3),
  topN = 10,
  outfile = "nameWithPath"
)
```

Arguments

data	A data matrix or a data frame where samples are in rows and features/traits are in columns.
varsIn	A vector of variables used for the prediction model.
selection	Logical. Default is FALSE. If TRUE, three variable selection methods can be chosen.
vsMethod	When "selection" is set to TRUE, three variable selection methods can be chosen, with LASSO2 as the default method. The other two methods are "LASSO2plus" and "LASSO_plus."
time	Time variable name.
event	Event variable name.
nrounds	The maximum number of boosting iterations.

probcut	Probability cutoff for risk group classification. Default is set to 0.8.
nthread	The number of parallel threads used to run XGBoost.
gamma	The minimum loss reduction required to make a further partition on a leaf node of the tree.
max_depth	The maximum depth of a tree. Increasing this value will make the model more complex and more likely to overfit.
eta	The step size shrinkage used in the update to prevent overfitting.
nclass	Number of risk groups. By default, it is 2; any samples not classified into high-risk groups are classified into the low-risk group. When 3 is chosen, samples are classified into low, middle, and high-risk groups.
topN	An integer indicating how many variables to select if LASSO_plus is chosen as the variable selection method.
outfile	A string for the output file, including the path if necessary but without the file type extension.

Details

If variable selection is needed, three variable selection methods are provided. Either the given variable or the selected variable list is used to build both an XGBoost model and a traditional Cox model. Risk scores for each model are calculated and ranked, then averaged for each sample. The top 1/3 of samples are defined as the high-risk group, while the bottom 1/3 of samples are defined as the low-risk group. The binary risk classification model is built based on these two risk groups using either the given variable or the selected variable list. The model is a linear combination of these variables, with weights defined as t values derived from the single-variable linear model of each variable on the two groups. Finally, the classification is based on empirical Bayesian probabilities.

Value

A list is returned with the following seven items:

ranks	Ranks from XGboost and Cox
twoEnds	High and low risk group samples identified by mean ranks from XGBoost and Cox models
weights	Weights for each variables used in the model
modelPars	Mean and standard error of model scores for each risk group
nclass	Number of risk groups
XGpred_score	Model XGpred score
XGpred_prob	Empirical Bayesian probability based on model XGpred score
XGpred_prob_class	Risk group classification based on XGpred_prob for the given probability cutoff
probcut	Probability cutoff for risk group classification

Author(s)

Aixiang Jiang

References

Tianqi Chen and Carlos Guestrin (2016), "XGBoost: A Scalable Tree Boosting System", 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016, https://arxiv.org/abs/1603.02754

Aoki T, Jiang A, Xu A et al.,(2023) Spatially Resolved Tumor Microenvironment Predicts Treatment Outcomes in Relapsed/Refractory Hodgkin Lymphoma. J Clin Oncol. 2023 Dec 19:JCO2301115. doi: 10.1200/JCO.23.01115.

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# For given time-to-event outcome and Xvars, we can build up a binary risk classification:
 xgobj = XGpred(data = tdat, varsIn = Xvars,
                time = "FFP..Years.", event = "Code.FFP",
                outfile = paste0(temp_dir, "/XGpred"))
# You might save the files to the directory you want.

# To delete the temp_dir, use the following:
unlink(temp_dir)
```

XGpred_predict *Predicting Risk Group Classification for a New Data Set*

Description

The XGpred_predict function is designed to predict risk group classifications for a new data set. This prediction is based on the assumption that an XGpred object is available from a training data set and that the new data set is comparable to the training data set.

In scenarios where the new and training data sets are not directly comparable, a calibration cohort is required to ensure accurate risk group predictions.

Usage

```
XGpred_predict(newdat = NULL, XGpredObj = NULL, scoreShift = 0)
```

Arguments

newdat	A data matrix or a data frame where samples are in rows and features/traits are in columns. It should include all variables needed for the prediction model.
XGpredObj	An XGpred object returned by the XGpred function. Although not all items are needed for prediction purposes, using the XGpred object as input is convenient.
scoreShift	A calibration value to subtract from the current model score if needed.

Value

A data frame containing XGpred_score, XGpred_prob, and XGpred_prob_class

Author(s)

Aixiang Jiang

References

Aoki T, Jiang A, Xu A et al.,(2023) Spatially Resolved Tumor Microenvironment Predicts Treatment Outcomes in Relapsed/Refractory Hodgkin Lymphoma. J Clin Oncol. 2023 Dec 19:JCO2301115. doi: 10.1200/JCO.23.01115. PMID: 38113419.

Examples

```
# Load in data sets:
data("datlist", package = "csmpv")
tdat = datlist$training
vdat = datlist$validation

# The function saves files locally. You can define your own temporary directory.
# If not, tempdir() can be used to get the system's temporary directory.
temp_dir = tempdir()
# As an example, let's define Xvars, which will be used later:
Xvars = c("highIPI", "B.Symptoms", "MYC.IHC", "BCL2.IHC", "CD10.IHC", "BCL6.IHC")
# For given time-to-event outcome and Xvars, we can build up a binary risk classification:
 xgobj = XGpred(data = tdat, varsIn = Xvars,
                time = "FFP..Years.",
                 event = "Code.FFP", outfile = paste0(temp_dir, "/XGpred"))
 tdat$XGpred_class = xgobj$XGpred_prob_class
# You might save the files to the directory you want.

# Now, we can predict the risk classification for a new data set:
 xgNew = XGpred_predict(newdat = vdat, XGpredObj = xgobj)

#' # To delete the "temp_dir", use the following:
unlink(temp_dir)
```

Index