Package: PIE (via r-universe)

January 27, 2025

Type Package

Title A Partially Interpretable Model with Black-Box Refinement

Version 1.0.0

Date 2025-01-20

Description Implements a novel predictive model, Partially Interpretable Estimators (PIE), which jointly trains an interpretable model and a black-box model to achieve high predictive performance as well as partial model. See the paper, Wang, Yang, Li, and Wang (2021) <doi:10.48550/arXiv.2105.02410>.

Depends R (>= 3.5.0), gglasso, xgboost

Imports splines, stats

Encoding UTF-8

License GPL-2

VignetteBuilder knitr

Suggests knitr, rmarkdown

Repository CRAN

RoxygenNote 7.3.2

NeedsCompilation no

Author Tong Wang [aut], Jingyi Yang [aut, cre], Yunyi Li [aut], Boxiang Wang [aut]

Maintainer Jingyi Yang <jy4057@stern.nyu.edu>

Date/Publication 2025-01-27 18:30:09 UTC

Config/pak/sysreqs make

Contents

`data_process` *data_process: process tabular data into the format for the PIE model.*

Description

This function take tabular dataset and meta-data (such as numerical columns and categorical columns), then output k fold cross validation dataset with splines on numerical features in order to capture the non-linear relationship among numerical features. Within this function, numerical features and target variable are normalized and reorganize into order: (numerical features, categorical features, target).

Usage

```
data_process(
  X,
  y,
  num_col,
  cat_col,
  y_col,
  k = 5,
  validation_rate = 0.2,
  spline_num = 5,
  random_seed = 1
)
```

Arguments

`X`	Feature columns in dataset
`y`	Target column in dataset
`num_col`	Index of the columns that are numerical features
`cat_col`	Index of the columns that are categorical features.
`y_col`	Index of the column that is the response.
`k`	Number of fold for cross validation dataset setup. By default ‘k = 5‘.
`validation_rate`	Validation ratio within training dataset. By default ‘validation_rate = 0.2‘
`spline_num`	The degree of freedom for natural splines. By default ‘spline_num = 5‘
`random_seed`	Random seed for cross validation data split. By default ‘random_seed = 1‘

Details

The function generates a suitable cross-validation dataset for PIE model. It contains training dataset, validation dataset, testing dataset and also group indicator for group lasso. When ‘k=5‘, the training testing splits in 80/20. When ‘validation_rate=0.2‘, 20 Setting ‘validation_rate=0‘ will only generate training and testing data without validation data.

Value

A list containing:

`spl_train_X` A list of splined training dataset where all numerical features are splined into ‘spline_num‘ columns. The number of element in list equals ‘k‘ the number of fold.

`orig_train_X` A list of original training dataset where the numerical features remains the original format. The number of element in list equals ‘k‘ the number of fold.

`train_y` A list of vectors representing target variable for training dataset. The number of element in list equals ‘k‘ the number of fold.

`spl_validation_X` A list of splined validation dataset where all numerical features are splined into ‘spline_num‘ columns. The number of element in list equals ‘k‘ the number of fold. It could be None, when ‘validation_rate == 0‘

`orig_validation_X` A list of original validation dataset where the numerical features remains the original format. The number of element in list equals ‘k‘ the number of fold. It could be None, when ‘validation_rate == 0‘

`validation_y` A list of vectors representing target variable for validation dataset. The number of element in list equals ‘k‘ the number of fold. It could be None, when ‘validation_rate == 0‘

`spl_test_X` A list of splined testing dataset where all numerical features are splined into ‘spline_num‘ columns. The number of element in list equals ‘k‘ the number of fold.

`orig_test_X` A list of original testing dataset where the numerical features remains the original format. The number of element in list equals ‘k‘ the number of fold.

`test_y` A list of vectors representing target variable for testing dataset. The number of element in list equals ‘k‘ the number of fold.

`lasso_group` A vector of consecutive integers describing the grouping of the coefficients

Examples

```
# Load the training data
data("winequality")

# Which columns are numerical?
num_col <- 1:11
# Which columns are categorical?
cat_col <- 12
# Which column is the response?
```

```
y_col <- ncol(winequality)

# Data Processing (the first 200 rows are sampled for demonstration)
dat <- data_process(X = as.matrix(winequality[1:200, -y_col]),
  y = winequality[1:200, y_col],
  num_col = num_col, cat_col = cat_col, y_col = y_col)
```

MAE *MAE: Mean Absolute Error*

Description

This function calculates the mean absolute error between the predicted values and the true values. The formula for MAE is:

$$MAE = \frac{1}{n}\sum_{i}|y_i - \hat{y}_i|$$

Usage

```
MAE(pred, true_label)
```

Arguments

pred	The predicted values of the dataset.
true_label	The actual target values of the dataset.

Value

A numeric value representing the mean absolute error (MAE).

Examples

```
# Load the training data
data("winequality")

# Which columns are numerical?
num_col <- 1:11
# Which columns are categorical?
cat_col <- 12
# Which column is the response?
y_col <- ncol(winequality)

# Data Processing (the first 200 rows are sampled for demonstration)
dat <- data_process(X = as.matrix(winequality[1:200, -y_col]),
  y = winequality[1:200, y_col],
  num_col = num_col, cat_col = cat_col, y_col = y_col)

# Fit a PIE model
```

```
fold <- 1
fit <- PIE_fit(
  X = dat$spl_train_X[[fold]],
  y = dat$train_y[[fold]],
  lasso_group = dat$lasso_group,
  X_orig = dat$orig_train_X[[fold]],
  lambda1 = 0.01, lambda2 = 0.01, iter = 5, eta = 0.05, nrounds = 200
)

# Prediction
pred <- predict(fit,
  X = dat$spl_validation_X[[fold]],
  X_orig = dat$orig_validation_X[[fold]]
)

# Validation
val_rrmae_test <- MAE(pred$total, dat$validation_y[[fold]])
```

PIE *PIE: A Partially Interpretable Model with Black-box Refinement*

Description

The PIE package implements a novel Partially Interpretable Model (PIE) framework introduced by Wang et al. <arxiv:2105.02410>. This framework jointly train an interpretable model and a black-box model to achieve high predictive performance as well as partial model transparency.

Functions

- `predict.PIE()`: Main function for generating predictions with the PIE model on dataset. - `PIE()`: Main function for training the PIE model with dataset. - `data_process()`: Process data into the format that can be used by PIE model. - `sparsity_count()`: Counts the number of features used in group lasso. - `RPE()`: Evaluate the RPE of a PIE model. - `MAE()`: Evaluate the MAE of a PIE model.

For more details, see the documentation for individual functions.

Author(s)

Maintainer: Jingyi Yang <jy4057@stern.nyu.edu>

Authors:

- Tong Wang
- Yunyi Li
- Boxiang Wang

PIE_fit *PIE: Partially Interpretable Model*

Description

Partially Interpretable Estimators (PIE), which jointly train an interpretable model and a black-box model to achieve high predictive performance as well as partial model transparency. PIE is designed to attribute a prediction to contribution from individual features via a linear additive model to achieve interpretability while complementing the prediction by a black-box model to boost the predictive performance. Experimental results show that PIE achieves comparable accuracy to the state-of-the-art black-box models on tabular data. In addition, the understandability of PIE is close to linear models as validated via human evaluations.

Usage

```
PIE_fit(X, y, lasso_group, X_orig, lambda1, lambda2, iter, eta, nrounds, ...)
```

Arguments

X	A matrix for the dataset features with numerical splines.
y	A vector for the dataset target label.
lasso_group	A vector that indicates groups
X_orig	A matrix for the dataset features without numerical splines.
lambda1	A numeric number for group lasso penalty. The larger the value, the larger the penalty.
lambda2	A numeric number for black-box model. The larger the value, the larger contribution of XGBoost model.
iter	A numeric number for iterations.
eta	A numeric number for learning rate of XGBoost model.
nrounds	A numeric number for number of rounds of XGBoost model.
...	Additional arguments passed to the XGBoost function.

Details

The PIE_fit function use training dataset to train the PIE model through jointly train an interpretable model and a black-box model to achieve high predictive performance as well as partial model transparency.

Value

An object of class PIE containing the following components:

Betas	The coefficient of group lasso model
Trees	The coefficients of XGBoost trees

rrMSE_fit	A matrix containing the evaluation between group lasso and y, and evaluation between full model and y for each iteration.
GAM_pred	A matrix containing the contribution of group lasso in each iteration.
Tree_pred	A matrix containing the contribution of XGBoost model in each iteration.
best_iter	The number of the best iteration.
lambda1	The lambda1 tuning parameter used in PIE.
lambda2	The lambda2 tuning parameter used in PIE.

Examples

```
# Load the training data
data("winequality")

# Which columns are numerical?
num_col <- 1:11
# Which columns are categorical?
cat_col <- 12
# Which column is the response?
y_col <- ncol(winequality)

# Data Processing (the first 200 rows are sampled for demonstration)
dat <- data_process(X = as.matrix(winequality[1:200, -y_col]),
  y = winequality[1:200, y_col],
  num_col = num_col, cat_col = cat_col, y_col = y_col)

# Fit a PIE model
fold <- 1
fit <- PIE_fit(
  X = dat$spl_train_X[[fold]],
  y = dat$train_y[[fold]],
  lasso_group = dat$lasso_group,
  X_orig = dat$orig_train_X[[fold]],
  lambda1 = 0.01, lambda2 = 0.01, iter = 5, eta = 0.05, nrounds = 200
)
```

---

predict.PIE *Make Predictions for PIE*

Description

predicts the response of a PIE object using new data.

Usage

```
## S3 method for class 'PIE'
predict(object, X, X_orig, ...)
```

Arguments

object	A fitted PIE object.
X	A matrix for the dataset with features expanded using numerical splines.
X_orig	A matrix for the dataset with original features without numerical splines.
...	Not used. Other arguments to predict.

Details

Make Predictions for PIE

This function predicts the response of a PIE object.

The PIE_predict function use generate predictions on dataset given the coefficients of group lasso and coefficients for XGBoost Trees

Value

A list containing:

total	The predicted value of the whole model for given features
white_box	The contribution of group lasso for the given features
black_box	The contribution of XGBoost model for the given features

Examples

```
# Load the training data
data("winequality")

# Which columns are numerical?
num_col <- 1:11
# Which columns are categorical?
cat_col <- 12
# Which column is the response?
y_col <- ncol(winequality)

# Data Processing (the first 200 rows are sampled for demonstration)
dat <- data_process(X = as.matrix(winequality[1:200, -y_col]),
  y = winequality[1:200, y_col],
  num_col = num_col, cat_col = cat_col, y_col = y_col)

# Fit a PIE model
fold <- 1
fit <- PIE_fit(
  X = dat$spl_train_X[[fold]],
  y = dat$train_y[[fold]],
  lasso_group = dat$lasso_group,
  X_orig = dat$orig_train_X[[fold]],
  lambda1 = 0.01, lambda2 = 0.01, iter = 5, eta = 0.05, nrounds = 200
)

# Prediction
```

```
pred <- predict(fit,
  X = dat$spl_validation_X[[fold]],
  X_orig = dat$orig_validation_X[[fold]]
)
```

RPE *RPE: Relative Prediction Error*

Description

This function takes predicted values and target values to evaluate the performance of a PIE model. The formula for RPE is:

$$RPE = \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

where $\bar{y} = \frac{1}{n} \sum_i^n y_i$.

Usage

```
RPE(pred, true_label)
```

Arguments

pred	The predicted values of the dataset.
true_label	The actual target values of the dataset.

Value

A numeric value representing the relative prediction error (RPE).

Examples

```
# Load the training data
data("winequality")

# Which columns are numerical?
num_col <- 1:11
# Which columns are categorical?
cat_col <- 12
# Which column is the response?
y_col <- ncol(winequality)

# Data Processing (the first 200 rows are sampled for demonstration)
dat <- data_process(X = as.matrix(winequality[1:200, -y_col]),
  y = winequality[1:200, y_col],
  num_col = num_col, cat_col = cat_col, y_col = y_col)

# Fit a PIE model
fold <- 1
```

```
fit <- PIE_fit(
  X = dat$spl_train_X[[fold]],
  y = dat$train_y[[fold]],
  lasso_group = dat$lasso_group,
  X_orig = dat$orig_train_X[[fold]],
  lambda1 = 0.01, lambda2 = 0.01, iter = 5, eta = 0.05, nrounds = 200
)

# Prediction
pred <- predict(fit,
  X = dat$spl_validation_X[[fold]],
  X_orig = dat$orig_validation_X[[fold]]
)

# Validation
val_rrmse_test <- RPE(pred$total, dat$validation_y[[fold]])
```

`sparsity_count` *sparsity_count*

Description

This function counts the number of features used in group lasso of PIE model.

Usage

```
sparsity_count(Betas, lasso_group)
```

Arguments

`Betas`	The coefficient of group lasso model.
`lasso_group`	The group indicator for group lasso model

Value

An integer: The number of features used in group lasso

Examples

```
# Load the training data
data("winequality")

# Which columns are numerical?
num_col <- 1:11
# Which columns are categorical?
cat_col <- 12
# Which column is the response?
y_col <- ncol(winequality)
```

```
# Data Processing (the first 200 rows are sampled for demonstration)
dat <- data_process(X = as.matrix(winequality[1:200, -y_col]),
  y = winequality[1:200, y_col],
  num_col = num_col, cat_col = cat_col, y_col = y_col)

# Fit a PIE model
fold <- 1
fit <- PIE_fit(
  X = dat$spl_train_X[[fold]],
  y = dat$train_y[[fold]],
  lasso_group = dat$lasso_group,
  X_orig = dat$orig_train_X[[fold]],
  lambda1 = 0.01, lambda2 = 0.01, iter = 5, eta = 0.05, nrounds = 200
)

# Sparsity count
sparsity_count(fit$Betas, dat$lasso_group)
```

`winequality` *Wine Quality Data*

Description

This dataset contains 5197 data points. It is related to Portuguese 'Vinho Verdo' wine. Input variables are based on physicochemical tests. This dataset can also be found at [Wine Quality](https://archive.ics.uci.edu/dataset/186/wine+quality) in UC Irvine Machine Learning Repository.

Usage

```
data(winequality)
```

Details

Wine Quality Data

A benchmark data set.

Value

A matrix with 5197 rows and 13 columns. The first 11 columns are numerical variables, the 12th column contains categorical variable, and the last column is the response.

Source

The data were introduced in Cortez et al. (2009).

References

Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009). "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, **47**(4), 547-553.

Examples

```
# Load the PIE library
library(PIE)

# Load the dataset
data(winequality)
```

Index