Package: fastml (via r-universe)

November 25, 2024

Type Package

Title Fast Machine Learning Model Training and Evaluation

Version 0.1.0

Description Streamlines the training, evaluation, and comparison of multiple machine learning models with minimal code by providing comprehensive data preprocessing and support for a wide range of algorithms with hyperparameter tuning. It offers performance metrics and visualization tools to facilitate efficient and effective machine learning workflows.

License GPL (>= 2)

Encoding UTF-8

Imports caret, pROC, reshape2, ggplot2, foreach, doParallel

Suggests knitr, rmarkdown, randomForest, kernlab, gbm, nnet, rpart, glmnet, xgboost, pls, C50, mboost, arm, ipred, caTools, e1071, MASS

RoxygenNote 7.3.2

NeedsCompilation no

Author Selcuk Korkmaz [aut, cre] (<`https://orcid.org/0000-0003-4632-6850`>), Dincer Goksuluk [aut] (<`https://orcid.org/0000-0002-2752-7668`>)

Maintainer Selcuk Korkmaz <`selcukorkmaz@gmail.com`>

Repository CRAN

Date/Publication 2024-11-12 14:30:02 UTC

Config/pak/sysreqs libicu-dev

Contents

`evaluate_models` *Evaluate Models Function*

Description

Evaluates the trained models on the test data and computes performance metrics.

Usage

```
evaluate_models(models, test_data, label, metric = "Accuracy")
```

Arguments

`models`	A list of trained model objects.
`test_data`	Preprocessed test data frame.
`label`	Name of the target variable.
`metric`	The performance metric to optimize (e.g., "Accuracy", "ROC").

Value

A list of performance metrics for each model.

`fastml` *Fast Machine Learning Function*

Description

Trains and evaluates multiple classification models.

Usage

```
fastml(
  data,
  label,
  algorithms = c("xgboost", "random_forest", "svm_radial"),
  test_size = 0.2,
  resampling_method = "cv",
  folds = 5,
  tune_params = NULL,
  metric = "Accuracy",
  n_cores = 1,
  stratify = TRUE,
  impute_method = NULL,
  encode_categoricals = TRUE,
  scaling_methods = c("center", "scale"),
  summaryFunction = NULL,
  seed = 123
)
```

Arguments

`data`	A data frame containing the features and target variable.
`label`	A string specifying the name of the target variable.
`algorithms`	A vector of algorithm names to use. Default is `c("xgboost", "random_forest", "svm_radial")`. Use `"all"` to run all supported algorithms.
`test_size`	A numeric value between 0 and 1 indicating the proportion of the data to use for testing. Default is `0.2`.
`resampling_method`	A string specifying the resampling method for cross-validation. Default is `"cv"` (cross-validation). Other options include `"none"`, `"boot"`, `"repeatedcv"`, etc.
`folds`	An integer specifying the number of folds for cross-validation. Default is `5`.
`tune_params`	A list specifying hyperparameter tuning ranges. Default is `NULL`.
`metric`	The performance metric to optimize during training. Default is `"Accuracy"`.
`n_cores`	An integer specifying the number of CPU cores to use for parallel processing. Default is `1`.
`stratify`	Logical indicating whether to use stratified sampling when splitting the data. Default is `TRUE`.
`impute_method`	Method for missing value imputation. Default is `NULL`.
`encode_categoricals`	Logical indicating whether to encode categorical variables. Default is `TRUE`.
`scaling_methods`	Vector of scaling methods to apply. Default is `c("center", "scale")`.
`summaryFunction`	A custom summary function for model evaluation. Default is `NULL`.
`seed`	An integer value specifying the random seed for reproducibility.

Value

An object of class `fastml_model` containing the best model, performance metrics, and other information.

Examples

```
 # Example 1: Using the iris dataset for binary classification (excluding 'setosa')
data(iris)
iris <- iris[iris$Species != "setosa", ]  # Binary classification
iris$Species <- factor(iris$Species)

# Train models
model <- fastml(
  data = iris,
  label = "Species"
)

# View model summary
summary(model)


# Example 2: Using the mtcars dataset for binary classification
data(mtcars)
mtcars$am <- factor(mtcars$am)  # Convert transmission (0 = automatic, 1 = manual) to a factor

# Train models with a different resampling method and specific algorithms
model2 <- fastml(
  data = mtcars,
  label = "am",
  algorithms = c("random_forest", "svm_radial"),
  resampling_method = "repeatedcv",
  folds = 3,
  test_size = 0.25
)

# View model performance
summary(model2)


# Example 3: Using the airquality dataset with missing values
data(airquality)
airquality <- na.omit(airquality)  # Simple example to remove missing values for demonstration
airquality$Month <- factor(airquality$Month)

# Train models with categorical encoding and scaling
model3 <- fastml(
  data = airquality,
  label = "Month",
  encode_categoricals = TRUE,
  scaling_methods = c("center", "scale")
)
```

```
# Evaluate and compare models
summary(model3)


# Example 4: Custom hyperparameter tuning for a random forest
data(iris)
iris <- iris[iris$Species != "setosa", ]  # Filter out 'setosa' for binary classification
iris$Species <- factor(iris$Species)
custom_tuning <- list(
  random_forest = expand.grid(mtry = c(1:10))
)

model4 <- fastml(
  data = iris,
  label = "Species",
  algorithms = c("random_forest"),
  tune_params = custom_tuning,
  metric = "Accuracy"
)

# View the results
summary(model4)
```

`load_model` *Load Model Function*

Description

Loads a trained model object from a file.

Usage

```
load_model(filepath)
```

Arguments

`filepath`	A string specifying the file path to load the model from.

Value

An object of class `fastml_model`.

plot.fastml_model *Plot Function for fastml_model*

Description

Generates plots to compare the performance of different models.

Usage

```
## S3 method for class 'fastml_model'
plot(x, ...)
```

Arguments

`x`	An object of class `fastml_model`.
`...`	Additional arguments (not used).

Value

Displays comparison plots of model performances.

predict.fastml_model *Predict Function for fastml_model*

Description

Makes predictions on new data using the trained model.

Usage

```
## S3 method for class 'fastml_model'
predict(object, newdata, ...)
```

Arguments

`object`	An object of class `fastml_model`.
`newdata`	A data frame containing new data for prediction.
`...`	Additional arguments (not used).

Value

A vector of predictions.

`save_model` *Save Model Function*

Description

Saves the trained model object to a file.

Usage

```
save_model(model, filepath)
```

Arguments

`model`	An object of class `fastml_model`.
`filepath`	A string specifying the file path to save the model.

Value

No return value, called for its side effect of saving the model object to a file.

`summary.fastml_model` *Summary Function for fastml_model*

Description

Provides a detailed summary of the models' performances.

Usage

```
## S3 method for class 'fastml_model'
summary(object, sort_metric = NULL, ...)
```

Arguments

`object`	An object of class `fastml_model`.
`sort_metric`	A string specifying which metric to sort the models by. Default is `NULL`, which prioritizes the optimized metric.
`...`	Additional arguments (not used).

Value

Prints a summary of the models' performances and displays comparison plots.

train_models *Train Specified Machine Learning Algorithms on the Training Data*

Description

Trains specified machine learning algorithms on the preprocessed training data.

Usage

```
train_models(
  train_data,
  label,
  algorithms,
  resampling_method,
  folds,
  repeats = NULL,
  tune_params,
  metric,
  summaryFunction = NULL,
  seed = 123
)
```

Arguments

`train_data`	Preprocessed training data frame.
`label`	Name of the target variable.
`algorithms`	Vector of algorithm names to train.
`resampling_method`	Resampling method for cross-validation (e.g., "cv", "repeatedcv").
`folds`	Number of folds for cross-validation.
`repeats`	Number of times to repeat cross-validation (only applicable for methods like "repeatedcv").
`tune_params`	List of hyperparameter tuning ranges.
`metric`	The performance metric to optimize.
`summaryFunction`	A custom summary function for model evaluation. Default is `NULL`.
`seed`	An integer value specifying the random seed for reproducibility.

Value

A list of trained model objects.

`validate_tuneGrid` *Validate and Complete the tuneGrid*

Description

Ensures that the tuneGrid includes all required hyperparameters and adjusts it based on cross-validation.

Usage

```
validate_tuneGrid(tuneGrid, default_params, required_params, resampling_method)
```

Arguments

`tuneGrid`	User-provided tuning grid.
`default_params`	Default hyperparameter ranges.
`required_params`	Required hyperparameters for the algorithm.
`resampling_method`	Logical indicating whether cross-validation is enabled.

Value

A validated and possibly modified tuneGrid.

Index