Package 'interpret'

October 13, 2019

Title Fit Interpretable Models and Explain Blackbox Machine Learning

Version 0.1.22

Date 2019-10-10

Description Machine Learning package for training interpretable models and explaining blackbox systems. Historically, the most intelligible models were not very accurate, and the most accurate models were not intelligible. Microsoft Research has developed an algorithm called the Explainable Boosting Machine (EBM) which has both high accuracy and intelligibility. EBM uses machine learning techniques like bagging and boosting to breathe new life into traditional GAMs (Generalized Additive Models). This makes them as accurate as random forests and gradient boosted trees, and also enhances their intelligibility and editability. Details on the EBM algorithm can be found in the paper by Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad (2015, <doi:10.1145/2783258.2788613>).

URL https://github.com/microsoft/interpret

BugReports https://github.com/microsoft/interpret/issues

License MIT + file LICENSE

Depends R (>= 3.0.0)

NeedsCompilation yes

SystemRequirements C++11

Author Samuel Jenkins [aut],
Harsha Nori [aut],
Paul Koch [aut],
Rich Caruana [aut, cre],
Microsoft Corporation [cph]

Maintainer Rich Caruana <interpretml@outlook.com>

Repository CRAN

Date/Publication 2019-10-13 14:40:02 UTC

R topics documented:

ebm_feature 2

ebm_feature_combination 3
get_best_model_feature_combination 3
get_current_model_feature_combination 4
get_interaction_score 5
initialize_interaction_classification 6
initialize_interaction_regression 7
initialize_training_classification 7
initialize_training_regression 9
training_step 10

Index **12**

`ebm_feature` *Create ebm_feature*

Description

Creates an ebm_feature

Usage

```
ebm_feature(
  count_bins,
  has_missing,
  feature_type
)
```

Arguments

`count_bins`	count bins
`has_missing`	has missing
`feature_type`	feature type

Value

Returns an S3 ebm_feature class that contains information about a single machine learning feature

Examples

```
feature <- ebm_feature(1, FALSE, "ordinal")
```

ebm_feature_combination *Create ebm_feature_combination*

Description

Creates an ebm_feature_combination

Usage

```
ebm_feature_combination(
  count_features_in_combination
)
```

Arguments

count_features_in_combination
: count features in combination

Value

Returns an S3 ebm_feature_combination class that contains information about a combination of ebm_feature objects

Examples

```
feature_combination <- ebm_feature_combination(1)
```

get_best_model_feature_combination *Get Best Model Feature Combination*

Description

Get Best Model Feature Combination

Usage

```
get_best_model_feature_combination(
  ebm_training,
  index_feature_combination
)
```

Arguments

ebm_training	ebm training
index_feature_combination	index feature combination

Value

Returns the best model tensor for a single explainable boosting machine feature combination

Examples

```
training_ptr <- initialize_training_regression(
  1L,
  list(ebm_feature(2)),
  list(ebm_feature_combination(1)),
  c(0),
  c(10, 10), c(0, 1), c(0, 0),
  c(12), c(1), c(0),
  0L)

validation_metric <- training_step(training_ptr, 0, 0.01, 2, 2, NULL, NULL)

model_feature_combination_tensor <- get_best_model_feature_combination(training_ptr, 0)
```

---

get_current_model_feature_combination *Get Current Model Feature Combination*

Description

Get Current Model Feature Combination

Usage

```
get_current_model_feature_combination(
  ebm_training,
  index_feature_combination
)
```

Arguments

ebm_training	ebm training
index_feature_combination	index feature combination

Value

Returns the current model tensor for a single explainable boosting machine feature combination

Examples

```
training_ptr <- initialize_training_regression(
  1L,
  list(ebm_feature(2)),
  list(ebm_feature_combination(1)),
  c(0),
  c(10, 10), c(0, 1), c(0, 0),
  c(12), c(1), c(0),
  0L)

validation_metric <- training_step(training_ptr, 0, 0.01, 2, 2, NULL, NULL)

model_feature_combination_tensor <- get_current_model_feature_combination(training_ptr, 0)
```

---

get_interaction_score *Get Interaction Score*

Description

Get Interaction Score

Usage

```
get_interaction_score(
  ebm_interaction,
  feature_indexes
)
```

Arguments

ebm_interaction	ebm interaction
feature_indexes	feature indexes

Value

Returns an interaction score that indicates the relative benefit of a proposed set of features as inputs to explainable boosting machine training

Examples

```
interaction_ptr <- initialize_interaction_regression(
  list(ebm_feature(1)),
  c(0), c(0), c(0))

interaction_score <- get_interaction_score(interaction_ptr, 0)
```

`initialize_interaction_classification` *Initializes Classification Interaction*

Description

Initializes Classification Interaction

Usage

```
initialize_interaction_classification(
  features,
  count_target_classes,
  targets,
  binned_data,
  predictor_scores
)
```

Arguments

`features`	features
`count_target_classes`	count target classes
`targets`	targets
`binned_data`	binned data
`predictor_scores`	predictor scores

Value

Returns an opaque externalptr object that can be passed to the function get_interaction_score to obtain the interaction score for any given set of features

Examples

```
interaction_ptr <- initialize_interaction_classification(
  list(ebm_feature(1)),
  2,
  c(0), c(0), c(0))
```

initialize_interaction_regression
Initializes Regression Interaction

Description

Initializes Regression Interaction

Usage

```
initialize_interaction_regression(
  features,
  targets,
  binned_data,
  predictor_scores
)
```

Arguments

features	features
targets	targets
binned_data	binned data
predictor_scores	predictor scores

Value

Returns an opaque externalptr object that can be passed to the function get_interaction_score to obtain the interaction score for any given set of features

Examples

```
interaction_ptr <- initialize_interaction_regression(
  list(ebm_feature(1)),
  c(0), c(0), c(0))
```

initialize_training_classification
Initializes Classification Training

Description

Initializes Classification Training

Usage

```
initialize_training_classification(
  random_seed,
  features,
  feature_combinations,
  feature_combination_indexes,
  count_target_classes,
  training_targets,
  training_binned_data,
  training_predictor_scores,
  validation_targets,
  validation_binned_data,
  validation_predictor_scores,
  count_inner_bags
)
```

Arguments

`random_seed`	random seed
`features`	features
`feature_combinations`	feature combinations
`feature_combination_indexes`	feature combination indexes
`count_target_classes`	count target classes
`training_targets`	training targets
`training_binned_data`	training binned data
`training_predictor_scores`	training predictor scores
`validation_targets`	validation targets
`validation_binned_data`	validation binned data
`validation_predictor_scores`	validation predictor scores
`count_inner_bags`	count inner bags

Value

Returns an opaque externalptr object that can be passed to the function training_step to iteratively boost on features in the explainable boosting machine model, or to the functions get_best_model_feature_combination, or get_current_model_feature_combination to obtain the explainable boosting machine tensors

Examples

```
training_ptr <- initialize_training_classification(
  1L,
  list(ebm_feature(1)),
  list(ebm_feature_combination(1)),
  c(0),
  3,
  c(0), c(0), c(0, 0, 0),
  c(0), c(0), c(0, 0, 0),
  0L)
```

`initialize_training_regression` *Initializes Regression Training*

Description

Initializes Regression Training

Usage

```
initialize_training_regression(
  random_seed,
  features,
  feature_combinations,
  feature_combination_indexes,
  training_targets,
  training_binned_data,
  training_predictor_scores,
  validation_targets,
  validation_binned_data,
  validation_predictor_scores,
  count_inner_bags
)
```

Arguments

`random_seed`	random seed
`features`	features
`feature_combinations`	feature combinations
`feature_combination_indexes`	feature combination indexes
`training_targets`	training targets
`training_binned_data`	training binned data

`training_predictor_scores`
: training predictor scores

`validation_targets`
: validation targets

`validation_binned_data`
: validation binned data

`validation_predictor_scores`
: validation predictor scores

`count_inner_bags`
: count inner bags

Value

Returns an opaque externalptr object that can be passed to the function training_step to iteratively boost on features in the explainable boosting machine model, or to the functions get_best_model_feature_combination, or get_current_model_feature_combination to obtain the explainable boosting machine tensors

Examples

```
training_ptr <- initialize_training_regression(
  1L,
  list(ebm_feature(1)),
  list(ebm_feature_combination(1)),
  c(0),
  c(0), c(0), c(0),
  c(0), c(0), c(0),
  0L)
```

---

training_step *Training Step*

Description

Takes one Training Step

Usage

```
training_step(
  ebm_training,
  index_feature_combination,
  learning_rate,
  count_tree_splits_max,
  count_instances_required_for_parent_split_min,
  training_weights,
  validation_weights
)
```

Arguments

ebm_training	ebm training
index_feature_combination	index feature combination
learning_rate	learning rate
count_tree_splits_max	count tree splits max
count_instances_required_for_parent_split_min	count instances required for parent split min
training_weights	training weights
validation_weights	validation weights

Value

Returns the root mean squared error when the model being trained is for a regression problem, or the log loss for a classification problem

Examples

```
training_ptr <- initialize_training_regression(
  1L,
  list(ebm_feature(2)),
  list(ebm_feature_combination(1)),
  c(0),
  c(10, 10), c(0, 1), c(0, 0),
  c(12), c(1), c(0),
  0L)

validation_metric <- training_step(training_ptr, 0, 0.01, 2, 2, NULL, NULL)
```

Index

ebm_feature, 2
ebm_feature_combination, 3

get_best_model_feature_combination, 3
get_current_model_feature_combination, 4
get_interaction_score, 5

initialize_interaction_classification, 6
initialize_interaction_regression, 7
initialize_training_classification, 7
initialize_training_regression, 9

training_step, 10