Package ‘iai’

June 18, 2024

Type Package

Title Interface to 'Interpretable AI' Modules

Version 1.10.1

Description An interface to the algorithms of 'Interpretable AI' from the R programming language. 'Interpretable AI' provides various modules, including 'Optimal Trees' for classification, regression, prescription and survival analysis, 'Optimal Imputation' for missing data imputation and outlier detection, and 'Optimal Feature Selection' for exact sparse regression. The 'iai' package is an open-source project. The 'Interpretable AI' software modules are proprietary products, but free academic and evaluation licenses are available.

URL https://www.interpretable.ai

SystemRequirements Julia (>= 1.0) and Interpretable AI System Image (>= 1.0.0)

License MIT + file LICENSE

Imports JuliaCall (>= 0.17.5), stringr, rlang, lifecycle, rappdirs, ggplot2, cowplot, rjson

RoxygenNote 7.2.3

Suggests testthat, covr, xml2, withr

NeedsCompilation no

Author Jack Dunn [aut, cre], Ying Zhuo [aut], Interpretable AI LLC [cph]

Maintainer Jack Dunn <jack@interpretable.ai>

Repository CRAN

Date/Publication 2024-06-18 04:00:02 UTC

Contents

acquire_license .......................................................... 7
add_julia_processes ......................................................... 7
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>glmnetcv_regressor</td>
<td>62</td>
</tr>
<tr>
<td>glmnetcv_survival_learner</td>
<td>63</td>
</tr>
<tr>
<td>grid_search</td>
<td>63</td>
</tr>
<tr>
<td>iai_setup</td>
<td>64</td>
</tr>
<tr>
<td>imputation_learner</td>
<td>64</td>
</tr>
<tr>
<td>impute</td>
<td>65</td>
</tr>
<tr>
<td>impute_cv</td>
<td>66</td>
</tr>
<tr>
<td>install_julia</td>
<td>66</td>
</tr>
<tr>
<td>install_system_image</td>
<td>67</td>
</tr>
<tr>
<td>is_categorical_split</td>
<td>68</td>
</tr>
<tr>
<td>is_hyperplane_split</td>
<td>68</td>
</tr>
<tr>
<td>is_leaf</td>
<td>69</td>
</tr>
<tr>
<td>is_mixed_ordinal_split</td>
<td>69</td>
</tr>
<tr>
<td>is_mixed_parallel_split</td>
<td>70</td>
</tr>
<tr>
<td>is_ordinal_split</td>
<td>70</td>
</tr>
<tr>
<td>is_parallel_split</td>
<td>71</td>
</tr>
<tr>
<td>load_graphviz</td>
<td>71</td>
</tr>
<tr>
<td>mean_imputation_learner</td>
<td>72</td>
</tr>
<tr>
<td>missing_goes_lower</td>
<td>72</td>
</tr>
<tr>
<td>multi_questionnaire</td>
<td>73</td>
</tr>
<tr>
<td>multi_questionnaire.default</td>
<td>73</td>
</tr>
<tr>
<td>multi_questionnaire.grid_search</td>
<td>74</td>
</tr>
<tr>
<td>multi_tree_plot</td>
<td>75</td>
</tr>
<tr>
<td>multi_tree_plot.default</td>
<td>75</td>
</tr>
<tr>
<td>multi_tree_plot.grid_search</td>
<td>76</td>
</tr>
<tr>
<td>numeric_classification_reward_estimator</td>
<td>77</td>
</tr>
<tr>
<td>numeric_regression_reward_estimator</td>
<td>77</td>
</tr>
<tr>
<td>numeric_reward_estimator</td>
<td>78</td>
</tr>
<tr>
<td>numeric_survival_reward_estimator</td>
<td>79</td>
</tr>
<tr>
<td>optimal_feature_selection_classifier</td>
<td>79</td>
</tr>
<tr>
<td>optimal_feature_selection_regressor</td>
<td>80</td>
</tr>
<tr>
<td>optimal_tree_classifier</td>
<td>81</td>
</tr>
<tr>
<td>optimal_tree_multi_classifier</td>
<td>81</td>
</tr>
<tr>
<td>optimal_tree_multi_regressor</td>
<td>82</td>
</tr>
<tr>
<td>optimal_tree_policy_maximizer</td>
<td>82</td>
</tr>
<tr>
<td>optimal_tree_policy_minimizer</td>
<td>83</td>
</tr>
<tr>
<td>optimal_tree_prescription_maximizer</td>
<td>83</td>
</tr>
<tr>
<td>optimal_tree_prescription_minimizer</td>
<td>84</td>
</tr>
<tr>
<td>optimal_tree_regressor</td>
<td>84</td>
</tr>
<tr>
<td>optimal_tree_survival_learner</td>
<td>85</td>
</tr>
<tr>
<td>optimal_tree_survivor</td>
<td>85</td>
</tr>
<tr>
<td>opt_knn_imputation_learner</td>
<td>86</td>
</tr>
<tr>
<td>opt_svm_imputation_learner</td>
<td>86</td>
</tr>
<tr>
<td>opt_tree_imputation_learner</td>
<td>87</td>
</tr>
<tr>
<td>plot.grid_search</td>
<td>87</td>
</tr>
<tr>
<td>plot.roc_curve</td>
<td>88</td>
</tr>
<tr>
<td>plot.similarity_comparison</td>
<td>88</td>
</tr>
<tr>
<td>plot.stability_analysis</td>
<td>89</td>
</tr>
<tr>
<td>Function</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>predict</td>
<td>90</td>
</tr>
<tr>
<td>predict.categorical_reward_estimator</td>
<td>90</td>
</tr>
<tr>
<td>predict.glmnetcv_learner</td>
<td>91</td>
</tr>
<tr>
<td>predict.numeric_reward_estimator</td>
<td>91</td>
</tr>
<tr>
<td>predict.optimal_feature_selection_learner</td>
<td>92</td>
</tr>
<tr>
<td>predict.supervised_learner</td>
<td>93</td>
</tr>
<tr>
<td>predict.supervised_multi_learner</td>
<td>93</td>
</tr>
<tr>
<td>predict.survival_learner</td>
<td>94</td>
</tr>
<tr>
<td>predict.expected_survival_time</td>
<td>95</td>
</tr>
<tr>
<td>predict.expected_survival_time.glmnetcv_survival_learner</td>
<td>95</td>
</tr>
<tr>
<td>predict.expected_survival_time.survival_curve</td>
<td>96</td>
</tr>
<tr>
<td>predict.expected_survival_time.survival_learner</td>
<td>96</td>
</tr>
<tr>
<td>predict_hazard</td>
<td>97</td>
</tr>
<tr>
<td>predict_hazard.glmnetcv_survival_learner</td>
<td>97</td>
</tr>
<tr>
<td>predict_hazard.survival_learner</td>
<td>98</td>
</tr>
<tr>
<td>predict_outcomes</td>
<td>99</td>
</tr>
<tr>
<td>predict_outcomes.policy_learner</td>
<td>99</td>
</tr>
<tr>
<td>predict_outcomes.prescription_learner</td>
<td>100</td>
</tr>
<tr>
<td>predict_proba</td>
<td>100</td>
</tr>
<tr>
<td>predict_proba.classification_learner</td>
<td>101</td>
</tr>
<tr>
<td>predict_proba.classification_multi_learner</td>
<td>101</td>
</tr>
<tr>
<td>predict_proba.glmnetcv_classifier</td>
<td>102</td>
</tr>
<tr>
<td>predict_reward</td>
<td>103</td>
</tr>
<tr>
<td>predict_reward.categorical_reward_estimator</td>
<td>103</td>
</tr>
<tr>
<td>predict_reward.numeric_reward_estimator</td>
<td>104</td>
</tr>
<tr>
<td>predict_shap</td>
<td>104</td>
</tr>
<tr>
<td>predict_treatment_outcome</td>
<td>105</td>
</tr>
<tr>
<td>predict_treatment_outcome.standard_error</td>
<td>105</td>
</tr>
<tr>
<td>predict_treatment_rank</td>
<td>106</td>
</tr>
<tr>
<td>print_path</td>
<td>107</td>
</tr>
<tr>
<td>prune_trees</td>
<td>108</td>
</tr>
<tr>
<td>questionnaire</td>
<td>108</td>
</tr>
<tr>
<td>questionnaire.optimal_feature_selection_learner</td>
<td>109</td>
</tr>
<tr>
<td>questionnaire.tree_learner</td>
<td>109</td>
</tr>
<tr>
<td>random_forest_classifier</td>
<td>110</td>
</tr>
<tr>
<td>random_forest_regressor</td>
<td>110</td>
</tr>
<tr>
<td>random_forest_survival_learner</td>
<td>111</td>
</tr>
<tr>
<td>rand_imputation_learner</td>
<td>112</td>
</tr>
<tr>
<td>read_json</td>
<td>112</td>
</tr>
<tr>
<td>refit_leaves</td>
<td>113</td>
</tr>
<tr>
<td>release_license</td>
<td>113</td>
</tr>
<tr>
<td>reset_display_label</td>
<td>114</td>
</tr>
<tr>
<td>resume_from_checkpoint</td>
<td>114</td>
</tr>
<tr>
<td>reward_estimator</td>
<td>115</td>
</tr>
<tr>
<td>roc_curve</td>
<td>115</td>
</tr>
<tr>
<td>roc_curve.classification_learner</td>
<td>116</td>
</tr>
<tr>
<td>roc_curve.classification_multi_learner</td>
<td>116</td>
</tr>
<tr>
<td>roc_curve.default</td>
<td>117</td>
</tr>
</tbody>
</table>
roc_curve.glmnetcv_classifier ................................. 118
score ......................................................... 118
score.categorical_reward_estimator ..................... 119
score.default ........................................... 119
score.glmnetcv_learner .................................. 120
score.numeric_reward_estimator ......................... 121
score.optimal_feature_selection_learner ............... 121
score.supervised_learner ................................ 122
score.supervised_multi_learner ......................... 123
set_display_label ....................................... 123
set_julia_seed ........................................... 124
set_params ................................................ 124
set_reward_kernel_bandwidth ............................. 125
set_rich_output_param ................................ 125
set_threshold ............................................. 126
show_in_browser .......................................... 126
show_in_browser.abstract_visualization .................. 127
show_in_browser.roc_curve ................................ 127
show_in_browser.tree_learner ............................. 128
show_questionnaire ...................................... 128
show_questionnaire.optimal_feature_selection_learner .. 129
show_questionnaire.tree_learner .......................... 129
similarity_comparison ................................... 130
single_knn_imputation_learner ......................... 131
split_data .................................................. 131
stability_analysis ....................................... 132
transform .................................................... 133
transform_and_expand ................................... 133
tree_plot .................................................... 134
tune_reward_kernel_bandwidth ............................ 134
variable_importance .................................... 135
variable_importance.learner ............................. 135
variable_importance.optimal_feature_selection_learner .. 136
variable_importance.tree_learner ....................... 136
variable_importance_similarity ......................... 137
write_booster ............................................. 138
write_dot .................................................... 138
write_html .................................................. 139
write_html.abstract_visualization ..................... 139
write_html.roc_curve .................................... 140
write_html.tree_learner ................................ 140
write_json .................................................. 141
write_pdf ................................................... 142
write_png ................................................... 142
write_questionnaire ................................... 143
write_questionnaire.optimal_feature_selection_learner ... 143
write_questionnaire.tree_learner ....................... 144
write_svg .................................................... 145
acquire_license

    xgboost_classifier ................................................. 145
    xgboost_regressor .................................................. 146
    xgboost_survival_learner ......................................... 146
    zero_imputation_learner .......................................... 147

Index 148

acquire_license  Acquire an IAI license for the current session.

Description
    Julia Equivalent: IAI.acquire_license

Usage
    acquire_license(...)  

Arguments
    ...  Refer to the Julia documentation for available parameters

IAI Compatibility
    Requires IAI version 3.1 or higher.

Examples
    ## Not run: iai::acquire_license()

add_julia_processes  Add additional Julia worker processes to parallelize workloads

Description
    Julia Equivalent: Distributed.addprocs!

Usage
    add_julia_processes(...)  

Arguments
    ...  Refer to the Julia documentation for available parameters
Details
For more information, refer to the documentation on parallelization

Examples
```r
## Not run: iai::add_julia_processes(3)
```

## all_treatment_combinations
Return a dataframe containing all treatment combinations of one or more treatment vectors, ready for use as treatment candidates in 'fit_predict' or 'predict'

### Description
Julia Equivalent: `IAI.all_treatment_combinations`

### Usage
```r
all_treatment_combinations(...)```

### Arguments
- `...` A vector of possible options for each treatment

### Examples
```r
## Not run: iai::all_treatment_combinations(c(1, 2, 3))```

## apply
Return the leaf index in a tree model into which each point in the features falls

### Description
Julia Equivalent: `IAI.apply`

### Usage
```r
apply(lnr, X)`

### Arguments
- `lnr` The learner or grid to query.
- `X` The features of the data.
**apply_nodes**

**Examples**

```r
## Not run: iai::apply(lnr, X)
```

---

**Description**

Julia Equivalent: `IAI.apply_nodes`

**Usage**

`apply_nodes(lnr, X)`

**Arguments**

- `lnr`: The learner or grid to query.
- `X`: The features of the data.

**Examples**

```r
## Not run: iai::apply_nodes(lnr, X)
```

---

**as.mixeddata**

*Convert a vector of values to IAI mixed data format*

**Description**

Julia Equivalent: `IAI.make_mixed_data`

**Usage**

`as.mixeddata(values, categorical_levels, ordinal_levels = c())`

**Arguments**

- `values`: The vector of values to convert
- `categorical_levels`: The values in `values` to treat as category levels
- `ordinal_levels` (optional): The values in `values` to treat as ordinal levels, in the order supplied
Examples

```r
## Not run:
df <- iris
set.seed(1)
df$mixed <- rnorm(150)
df$mixed[1:5] <- NA # Insert some missing values
df$mixed[6:10] <- "Not graded"
df$mixed <- iai::as.mixeddata(df$mixed, c("Not graded"))

## End(Not run)
```

---

### autoplot.grid_search

**Description**

Construct a `ggplot2::ggplot` object plotting grid search results for Optimal Feature Selection learners.

**Usage**

```r
type = stop("type is required"), ...
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>The grid search to plot</td>
</tr>
<tr>
<td>type</td>
<td>The type of plot to construct (either &quot;validation&quot; or &quot;importance&quot;, for more information refer to the Julia documentation for plotting grid search results)</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments (unused)</td>
</tr>
</tbody>
</table>

**IAI Compatibility**

Requires IAI version 2.2 or higher.

**Examples**

```r
## Not run: ggplot2::autoplot(grid)
```
autoplot.roc_curve

### Description

Construct a `ggplot2::ggplot` object plotting the ROC curve

### Usage

```r
## S3 method for class 'roc_curve'
autoplot(object, ...)
```

### Arguments

- `object`: The ROC curve to plot
- `...`: Additional arguments (unused)

### IAI Compatibility

Requires IAI version 2.1 or higher.

### Examples

```r
## Not run: ggplot2::autoplot(roc)
```

autoplot.similarity_comparison

### Description

Construct a `ggplot2::ggplot` object plotting the results of the similarity comparison

### Usage

```r
## S3 method for class 'similarity_comparison'
autoplot(object, ...)
```

### Arguments

- `object`: The similarity comparison to plot
- `...`: Additional arguments (unused)
IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

```r
## Not run: ggplot2::autoplot(similarity)
```

---

**Description**

Construct a `ggplot2::ggplot` object plotting the results of the stability analysis

**Usage**

```r
## S3 method for class 'stability_analysis'
autoplot(object, ...)
```

**Arguments**

- `object`: The stability analysis to plot
- `...`: Additional arguments (unused)

**IAI Compatibility**

Requires IAI version 2.2 or higher.

**Examples**

```r
## Not run: ggplot2::autoplot(stability)
```
categorical_classification_reward_estimator

Learner for conducting reward estimation with categorical treatments and classification outcomes

Description

Julia Equivalent: IAI.CategoricalClassificationRewardEstimator

Usage

categorical_classification_reward_estimator(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: lnr <- iai::categorical_classification_reward_estimator()

categorical_regression_reward_estimator

Learner for conducting reward estimation with categorical treatments and regression outcomes

Description

Julia Equivalent: IAI.CategoricalRegressionRewardEstimator

Usage

categorical_regression_reward_estimator(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.
categorical_reward_estimator

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

```r
## Not run: lnr <- iai::categorical_regression_reward_estimator()
```

categorical_reward_estimator

Learner for conducting reward estimation with categorical treatments

Description

This function was deprecated in iai 1.6.0, and [categorical_classification_reward_estimator()] or [categorical_classification_reward_estimator()] should be used instead.

Usage

categorical_reward_estimator(...)  

Arguments

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Details

This deprecation is no longer supported as of the IAI v3 release.

IAI Compatibility

Requires IAI version 2.0, 2.1 or 2.2.

Examples

```r
## Not run: lnr <- iai::categorical_reward_estimator()
```
categorical_survival_reward_estimator

Learner for conducting reward estimation with categorical treatments and survival outcomes

Description

Julia Equivalent: `IAI.CategoricalSurvivalRewardEstimator`

Usage

`categorical_survival_reward_estimator(...)`

Arguments

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

```r
## Not run: lnr <- iai::categorical_survival_reward_estimator()
```

cleanup_installation

Remove all traces of automatic Julia/IAI installation

Description

Removes files created by `install_julia` and `install_system_image`

Usage

`cleanup_installation()`

Examples

```r
## Not run: iai::cleanup_installation()
```
**clone**

Return an unfitted copy of a learner with the same parameters.

**Description**

Julia Equivalent: `IAI.clone`

**Usage**

```r
clone(lnr)
```

**Arguments**

- `lnr` The learner to copy.

**Examples**

```r
## Not run: new_lnr <- iai::clone(lnr)
```

**convert_treatments_to_numeric**

Convert 'treatments' from symbol/string format into numeric values.

**Description**

Julia Equivalent: `IAI.convert_treatments_to_numeric`

**Usage**

```r
convert_treatments_to_numeric(treatments)
```

**Arguments**

- `treatments` The treatments to convert

**Examples**

```r
## Not run: iai::convert_treatments_to_numeric(c("1", "2", "3"))
```
**copy_splits_and_refit_leaves**

*Copy the tree split structure from one learner into another and refit the models in each leaf of the tree using the supplied data*

**Description**

Julia Equivalent: `IAI.copy_splits_and_refit_leaves!`

**Usage**

```
copy_splits_and_refit_leaves(new_lnr, orig_lnr, ...)
```

**Arguments**

- `new_lnr`: The learner to modify and refit
- `orig_lnr`: The learner from which to copy the tree split structure
- `...`: Refer to the Julia documentation for available parameters

**IAI Compatibility**

Requires IAI version 3.0 or higher.

**Examples**

```julia
## Not run: iai::copy_splits_and_refit_leaves(new_lnr, orig_lnr, ...)
```

**decision_path**

*Return a matrix where entry \((i, j)\) is true if the \(i\)th point in the features passes through the \(j\)th node in a trained tree model.*

**Description**

Julia Equivalent: `IAI.decision_path`

**Usage**

```
decision_path(lnr, X)
```

**Arguments**

- `lnr`: The learner or grid to query.
- `X`: The features of the data.
## delete_rich_output_param

*Delete a global rich output parameter*

### Description

Julia Equivalent: `IAI.delete_rich_output_param!`

### Usage

```julia
delete_rich_output_param(key)
```

### Arguments

- **key**
  
  The parameter to delete.

### Examples

```julia
## Not run: iai::delete_rich_output_param("simple_layout")
```

## equal_propensity_estimator

*Learner that estimates equal propensity for all treatments.*

### Description

For use with data from randomized experiments where treatments are known to be randomly assigned.

### Usage

```julia
equal_propensity_estimator(...)```

### Arguments

```julia
... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.
```

### Details

Julia Equivalent: `IAI.EqualPropensityEstimator`
IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: lnr <- iai::equal_propensity_estimator()
```
fit.imputation_learner

Fits an imputation learner to the training data.

Description

Additional keyword arguments are available for fitting imputation learners - please refer to the Julia documentation.

Usage

## S3 method for class 'imputation_learner'
fit(obj, X, ...)

Arguments

obj The learner or grid to fit.
X The features of the data.
... Refer to the Julia documentation for available parameters.

Details

Julia Equivalent: IAI.fit!

Examples

## Not run: iai::fit(lnr, X)
### fit.learner

_Fits a model to the training data_

**Description**

Julia Equivalent: _IAI.fit!

**Usage**

```r
## S3 method for class 'learner'
fit(obj, X, ...)
```

**Arguments**

- `obj`: The learner to fit.
- `X`: The features of the data.
- `...`: Other parameters, including zero or more target vectors as required by the problem type. Refer to the Julia documentation for available parameters.

**Examples**

```r
## Not run: iai::fit(lnr, X, y)
```

### fit.optimal_feature_selection_learner

_Fits an Optimal Feature Selection learner to the training data_

**Description**

When the `coordinated_sparsity` parameter of the learner is TRUE, additional keyword arguments are required - please refer to the Julia documentation.

**Usage**

```r
## S3 method for class 'optimal_feature_selection_learner'
fit(obj, X, ...)
```

**Arguments**

- `obj`: The learner or grid to fit.
- `X`: The features of the data.
- `...`: Other parameters, including zero or more target vectors as required by the problem type. Refer to the Julia documentation for available parameters.
fit_and_expand

Details
Julia Equivalent: `IAI.fit!`

IAI Compatibility
Requires IAI version 1.1 or higher.

Examples
## Not run: iai::fit(lnr, X)

---

fit_and_expand

Fit an imputation learner with training features and create adaptive indicator features to encode the missing pattern

Description
Julia Equivalent: `IAI.fit_and_expand!`

Usage

```r
fit_and_expand(lnr, X, ...)
```

Arguments

- `lnr` The learner to use for imputation.
- `X` The dataframe in which to impute missing values.
- `...` Refer to the Julia documentation for available parameters.

IAI Compatibility
Requires IAI version 3.0 or higher.

Examples
## Not run: lnr <- iai::fit_and_expand(lnr, X, type = "finite")
fit_cv

Fits a grid search to the training data with cross-validation

Description

Julia Equivalent: IAI.fit_cv!

Usage

fit_cv(grid, X, ...)

Arguments

grid The grid to fit.
X The features of the data.
... Other parameters, including zero or more target vectors as required by the problem type. Refer to the Julia documentation for available parameters.

Examples

## Not run:
X <- iris[, 1:4]
y <- iris$Species
grid <- iai::grid_search(
    iai::optimal_tree_classifier(max_depth = 1),
)
iai::fit_cv(grid, X, y)
## End(Not run)

fit_predict

Generic function for fitting a reward estimator on features, treatments and returning predicted counterfactual rewards and scores of the internal estimators.

Description

Julia Equivalent: IAI.fit_predict!

Usage

fit_predict(obj, ...)

Arguments

obj The object controlling which method is used
... Arguments depending on the specific method used
fit_predict.categorical_reward_estimator

Fit a categorical reward estimator on features, treatments and outcomes and return predicted counterfactual rewards for each observation, under each treatment observed in the data, as well as the scores of the internal estimators.

Description

Julia Equivalent: IAI.fit_predict!

Usage

```r
## S3 method for class 'categorical_reward_estimator'
fit_predict(obj, X, treatments, ...)
```

Arguments

- `obj` The learner or grid to use for estimation
- `X` The features of the data.
- `treatments` The treatment applied to each point in the data.
- `...` Additional arguments depending on the treatment and outcome types. Refer to the Julia documentation for more information.

IAI Compatibility

Requires IAI version 2.0 or higher.

Examples

```r
## Not run: iai::fit_predict(obj, X, treatments, outcomes)
```

fit_predict.numeric_reward_estimator

Fit a numeric reward estimator on features, treatments and outcomes and return predicted counterfactual rewards for each observation, under each treatment candidate, as well as the scores of the internal estimators.

Description

Julia Equivalent: IAI.fit_predict!
Usage

```r
## S3 method for class 'numeric_reward_estimator'
fit_predict(obj, X, treatments, ...)
```

Arguments

- `obj`: The learner or grid to use for estimation
- `X`: The features of the data.
- `treatments`: The treatment applied to each point in the data.
- `...`: Additional arguments depending on the treatment and outcome types. Refer to the Julia documentation for more information.

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: iai::fit_predict(obj, X, treatments, outcomes)
```

---

fit_transform

Fit an imputation model using the given features and impute the missing values in these features

Description

Similar to calling `fit.imputation_learner` followed by `transform`

Usage

```r
fit_transform(lnr, X, ...)```

Arguments

- `lnr`: The learner or grid to use for imputation
- `X`: The features of the data.
- `...`: Refer to the Julia documentation for available parameters.

Details

Julia Equivalent: `IAI.fit_transform!`
Examples

```r
## Not run:
X <- iris
X[1, 1] <- NA
grid <- iai::grid_search(
  iai::imputation_learner(),
  method = c("opt_knn", "opt_tree"),
)
iai::fit_transform(grid, X)
## End(Not run)
```

Description

Julia Equivalent: `IAI.fit_transform_cv!`

Usage

`fit_transform_cv(grid, X, ...)`

Arguments

- `grid` The grid to use for imputation
- `X` The features of the data.
- `...` Refer to the Julia documentation for available parameters.

Examples

```r
## Not run:
X <- iris
X[1, 1] <- NA
grid <- iai::grid_search(
  iai::imputation_learner(),
  method = c("opt_knn", "opt_tree"),
)
iai::fit_transform_cv(grid, X)
## End(Not run)
```
**get_best_params**

Return the best parameter combination from a grid

---

**get_best_params**

Return the best parameter combination from a grid

**Description**

Julia Equivalent: `IAI.get_best_params`

**Usage**

```
get_best_params(grid)
```

**Arguments**

- `grid` The grid search to query.

**Examples**

```r
## Not run: iai::get_best_params(grid)
```

---

**get_classification_label**

Generic function for returning the predicted label in the node of a classification tree

---

**Description**

Generic function for returning the predicted label in the node of a classification tree

**Usage**

```
get_classification_label(obj, ...)
```

**Arguments**

- `obj` The object controlling which method is used
- `...` Arguments depending on the specific method used
get_classification_label.classification_tree_learner

Return the predicted label at a node of a tree

Description

Julia Equivalent: `IAI.get_classification_label`

Usage

```r
## S3 method for class 'classification_tree_learner'
get_classification_label(obj, node_index, ...)
```

Arguments

- `obj` - The learner to query.
- `node_index` - The node in the tree to query.
- `...` - Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: iai::get_classification_label(lnr, 1)
```

---

get_classification_label.classification_tree_multi_learner

Return the predicted label at a node of a multi-task tree

Description

Julia Equivalent: `IAI.get_classification_label` and `IAI.get_classification_label`

Usage

```r
## S3 method for class 'classification_tree_multi_learner'
get_classification_label(obj, node_index, ...)
```

Arguments

- `obj` - The learner to query.
- `node_index` - The node in the tree to query.
- `...` - Refer to the Julia documentation for available parameters.
**get_classification_proba**

**IAI Compatibility**

Requires IAI version 3.2 or higher.

**Examples**

```r
## Not run: iai::get_classification_label(lnr, 1)
```

---

**get_classification_proba**

*Generic function for returning the probabilities of class membership at a node of a classification tree*

---

**Description**

Generic function for returning the probabilities of class membership at a node of a classification tree.

**Usage**

```r
get_classification_proba(obj, ...)
```

**Arguments**

- `obj` The object controlling which method is used
- `...` Arguments depending on the specific method used

---

**get_classification_proba.classification_tree_learner**

*Return the predicted probabilities of class membership at a node of a tree*

---

**Description**

Julia Equivalent: `IAI.get_classification_proba`

**Usage**

```r
## S3 method for class 'classification_tree_learner'
get_classification_proba(obj, node_index, ...)
```

**Arguments**

- `obj` The learner to query.
- `node_index` The node in the tree to query.
- `...` Refer to the Julia documentation for available parameters.
Examples

```r
## Not run: iai::get_classification_proba(lnr, 1)
```

---

`get_classification_proba.classification_tree_multi_learner`

Return the predicted probabilities of class membership at a node of a multi-task tree

Description

Julia Equivalent: `IAI.get_classification_proba` and `IAI.get_classification_proba`

Usage

```r
## S3 method for class 'classification_tree_multi_learner'
get_classification_proba(obj, node_index, ...)
```

Arguments

- `obj`: The learner to query.
- `node_index`: The node in the tree to query.
- `...`: Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

```r
## Not run: iai::get_classification_proba(lnr, 1)
```

---

`get_cluster_assignments`

Return the indices of the trees assigned to each cluster, under the clustering of a given number of trees

Description

Julia Equivalent: `IAI.get_cluster_assignments`

Usage

```r
get_cluster_assignments(stability, num_trees)
```
**get_cluster_details**

Arguments

- **stability**  
  The stability analysis to query

- **num_trees**  
  The number of trees to include in the clustering

**IAI Compatibility**

Requires IAI version 2.2 or higher.

**Examples**

```r
## Not run: iai::get_cluster_assignments(stability, num_trees)
```

---

**get_cluster_details**  
Return the centroid information for each cluster, under the clustering of a given number of trees

**Description**

Julia Equivalent: `IAI.get_cluster_details`

**Usage**

```r
get_cluster_details(stability, num_trees)
```

**Arguments**

- **stability**  
  The stability analysis to query

- **num_trees**  
  The number of trees to include in the clustering

**IAI Compatibility**

Requires IAI version 2.2 or higher.

**Examples**

```r
## Not run: iai::get_cluster_details(stability, num_trees)
```
get_cluster_distances  Return the distances between the centroids of each pair of clusters, under the clustering of a given number of trees

Description
Julia Equivalent: `IAI.get_cluster_distances`

Usage
```julia
get_cluster_distances(stability, num_trees)
```

Arguments
- **stability**: The stability analysis to query.
- **num_trees**: The number of trees to include in the clustering.

IAI Compatibility
Requires IAI version 2.2 or higher.

Examples
```julia
## Not run: iai::get_cluster_distances(stability, num_trees)
```

get_depth  Get the depth of a node of a tree

Description
Julia Equivalent: `IAI.get_depth`

Usage
```julia
get_depth(lnr, node_index)
```

Arguments
- **lnr**: The learner to query.
- **node_index**: The node in the tree to query.

Examples
```julia
## Not run: iai::get_depth(lnr, 1)
```
get_estimation_densities

Return the total kernel density surrounding each treatment candidate for the propensity/outcome estimation problems in a fitted learner.

Description

Julia Equivalent: IAI.get_estimation_densities

Usage

get_estimation_densities(lnr, ...)

Arguments

lnr The learner from which to extract densities
...

Refer to the Julia documentation for other parameters

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: iai::get_estimation_densities(lnr, ...)

get_features_used

Return the names of the features used by the learner

Description

Julia Equivalent: IAI.get_features_used

Usage

get_features_used(lnr)

Arguments

lnr The learner to query.

IAI Compatibility

Requires IAI version 2.2 or higher.
### get_grid_results

Return a summary of the results from the grid search

**Description**

This function was deprecated and renamed to `get_grid_result_summary()` in iai 1.5.0. This is for consistency with the IAI v2.2.0 Julia release.

**Usage**

```julia
get_grid_results(grid)
```

**Arguments**

- `grid`  
  The grid search to query.

**Examples**

```julia
## Not run: iai::get_grid_results(grid)
```

### get_grid_result_details

Return a vector of lists detailing the results of the grid search

**Description**

Julia Equivalent: `IAI.get_grid_result_details`

**Usage**

```julia
get_grid_result_details(grid)
```

**Arguments**

- `grid`  
  The grid search to query.

**IAI Compatibility**

Requires IAI version 2.2 or higher.

**Examples**

```julia
## Not run: iai::get_grid_result_details(grid)
```
get_grid_result_summary

Return a summary of the results from the grid search

Description

Julia Equivalent: `IAI.get_grid_result_summary`

Usage

```
get_grid_result_summary(grid)
```

Arguments

- `grid`: The grid search to query.

Examples

```r
## Not run: iai::get_grid_result_summary(grid)
```

get_learner

Return the fitted learner using the best parameter combination from a grid

Description

Julia Equivalent: `IAI.get_learner`

Usage

```
get_learner(grid)
```

Arguments

- `grid`: The grid to query.

Examples

```r
## Not run: lnr <- iai::get_learner(grid)
```
get_lower_child

Get the index of the lower child at a split node of a tree

Description

Julia Equivalent: `IAI.get_lower_child`

Usage

`get_lower_child(lnr, node_index)`

Arguments

- `lnr` The learner to query.
- `node_index` The node in the tree to query.

Examples

```r
## Not run: iai::get_lower_child(lnr, 1)
```

---

get_machine_id

Return the machine ID for the current computer.

Description

This ID ties the IAI license file to your machine.

Usage

`get_machine_id()`

Examples

```r
## Not run: iai::get_machine_id()
```
**get_num_fits**

Generic function for returning the number of fits in a trained learner

**Description**

Generic function for returning the number of fits in a trained learner

**Usage**

```
get_num_fits(obj, ...)
```

**Arguments**

- `obj`: The object controlling which method is used
- `...`: Arguments depending on the specific method used

**get_num_fits.glmnetcv_learner**

Return the number of fits along the path in a trained GLMNet learner

**Description**

Julia Equivalent: `IAI.get_num_fits`

**Usage**

```
## S3 method for class 'glmnetcv_learner'
get_num_fits(obj, ...)
```

**Arguments**

- `obj`: The GLMNet learner to query.
- `...`: Additional arguments (unused)

**IAI Compatibility**

Requires IAI version 2.1 or higher.

**Examples**

```
## Not run: lnr <- iai::get_num_fits(lnr)
```
get_num_fits.optimal_feature_selection_learner

Return the number of fits along the path in a trained Optimal Feature Selection learner

Description

Julia Equivalent: `IAI.get_num_fits`

Usage

```r
## S3 method for class 'optimal_feature_selection_learner'
get_num_fits(obj, ...)
```

Arguments

- `obj` The Optimal Feature Selection learner to query.
- `...` Additional arguments (unused)

IAI Compatibility

Requires IAI version 3.0 or higher.

Examples

```r
## Not run: iai::get_num_fits(lnr)
```

get_num_nodes

Return the number of nodes in a trained learner

Description

Julia Equivalent: `IAI.get_num_nodes`

Usage

```r
get_num_nodes(lnr)
```

Arguments

- `lnr` The learner to query.

Examples

```r
## Not run: iai::get_num_nodes(lnr)
```
### get_num_samples

Get the number of training points contained in a node of a tree

**Description**

Julia Equivalent: `IAI.get_num_samples`

**Usage**

```julia
get_num_samples(lnr, node_index)
```

**Arguments**

- `lnr` The learner to query.
- `node_index` The node in the tree to query.

**Examples**

```julia
## Not run: iai::get_num_samples(lnr, 1)
```

### get_params

Return the value of all parameters on a learner

**Description**

Julia Equivalent: `IAI.get_params`

**Usage**

```julia
get_params(lnr)
```

**Arguments**

- `lnr` The learner to query.

**Examples**

```julia
## Not run: iai::get_params(lnr)
```
get_parent

*Get the index of the parent node at a node of a tree*

**Description**

Julia Equivalent: \texttt{IAI.get\_parent}

**Usage**

\texttt{get\_parent(lnr, node\_index)}

**Arguments**

- \texttt{lnr} \hspace{0.5cm} The learner to query.
- \texttt{node\_index} \hspace{0.5cm} The node in the tree to query.

**Examples**

\texttt{## Not run: iai::get\_parent(lnr, 2)}

get_policy_treatment_outcome

*Return the quality of the treatments at a node of a tree*

**Description**

Julia Equivalent: \texttt{IAI.get_policy\_treatment\_outcome}

**Usage**

\texttt{get_policy\_treatment\_outcome(lnr, node\_index, ...)}

**Arguments**

- \texttt{lnr} \hspace{0.5cm} The learner to query.
- \texttt{node\_index} \hspace{0.5cm} The node in the tree to query.
- \texttt{...} \hspace{0.5cm} Refer to the Julia documentation for available parameters.

**IAI Compatibility**

Requires IAI version 2.1 or higher.

**Examples**

\texttt{## Not run: iai::get\_policy\_treatment\_outcome(lnr, 1)}
Description

Julia Equivalent: `IAI.get_policy_treatment_outcome_standard_error`

Usage

```
get_policy_treatment_outcome_standard_error(lnr, node_index, ...)
```

Arguments

- `lnr`: The learner to query.
- `node_index`: The node in the tree to query.
- `...`: Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

```
## Not run: iai::get_policy_treatment_outcome_standard_error(lnr, 1)
```

Description

Julia Equivalent: `IAI.get_policy_treatment_rank`

Usage

```
get_policy_treatment_rank(lnr, node_index, ...)
```

Arguments

- `lnr`: The learner to query.
- `node_index`: The node in the tree to query.
- `...`: Refer to the Julia documentation for available parameters.
get_prediction_constant

Generic function for returning the prediction constant in a trained learner

Description

Generic function for returning the prediction constant in a trained learner

Usage

get_prediction_constant(obj, ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>The object controlling which method is used</td>
</tr>
<tr>
<td>...</td>
<td>Arguments depending on the specific method used</td>
</tr>
</tbody>
</table>

get_prediction_constant.glmnetcv_learner

Return the constant term in the prediction in a trained GLMNet learner

Description

Julia Equivalent: IAI.get_prediction_constant

Usage

## S3 method for class 'glmnetcv_learner'
get_prediction_constant(obj, fit_index = NULL, ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>The learner to query.</td>
</tr>
<tr>
<td>fit_index</td>
<td>The index of the fit in the path to use for prediction, defaulting to the best fit if not supplied.</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments (unused)</td>
</tr>
</tbody>
</table>
get_prediction_constant.optimal_feature_selection_learner

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: iai::get_prediction_constant(lnr)
```

---

**get_prediction_constant.optimal_feature_selection_learner**

*Return the constant term in the prediction in a trained Optimal Feature Selection learner*

Description

Julia Equivalent: `IAI.get_prediction_constant`

Usage

```r
## S3 method for class 'optimal_feature_selection_learner'
get_prediction_constant(obj, fit_index = NULL, ...)
```

Arguments

- `obj` The learner to query.
- `fit_index` The index of the cluster to use for prediction, if the `coordinated_sparsity` parameter on the learner is TRUE.
- `...` Additional arguments (unused)

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

```r
## Not run: iai::get_prediction_constant(lnr)
```
get_prediction_weights

Generic function for returning the prediction weights in a trained learner

Description

Generic function for returning the prediction weights in a trained learner

Usage

get_prediction_weights(obj, ...)

Arguments

obj The object controlling which method is used
...

Arguments depending on the specific method used

get_prediction_weights.glmnetcv_learner

Return the weights for numeric and categoric features used for prediction in a trained GLMNet learner

Description

Julia Equivalent: IAI.get_prediction_weights

Usage

## S3 method for class 'glmnetcv_learner'
get_prediction_weights(obj, fit_index = NULL, ...)

Arguments

obj The learner to query.
fit_index The index of the fit in the path to use for prediction, defaulting to the best fit if not supplied.
...
Additional arguments (unused)

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

## Not run: iai::get_prediction_weights(lnr)
get_prediction_weights.optimal_feature_selection_learner

Return the weights for numeric and categoric features used for prediction in a trained Optimal Feature Selection learner

Description

Julia Equivalent: `IAI.get_prediction_weights`

Usage

```r
## S3 method for class 'optimal_feature_selection_learner'
get_prediction_weights(obj, fit_index = NULL, ...)
```

Arguments

- `obj` The learner to query.
- `fit_index` The index of the cluster to use for prediction, if the `coordinated_sparsity` parameter on the learner is TRUE.
- `...` Additional arguments (unused)

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

```r
## Not run: iai::get_prediction_weights(lnr)
```

get_prescription_treatment_rank

Return the treatments ordered from most effective to least effective at a node of a tree

Description

Julia Equivalent: `IAI.get_prescription_treatment_rank`

Usage

```r
get_prescription_treatment_rank(lnr, node_index, ...)
```
Arguments

lnr The learner to query.
node_index The node in the tree to query.
...
Refer to the Julia documentation for available parameters.

Examples

## Not run: iai::get_prescription_treatment_rank(lnr, 1)

---

get_regression_constant

*Generic function for returning the constant term in the regression prediction at a node of a tree*

Description

Generic function for returning the constant term in the regression prediction at a node of a tree

Usage

```{r}
get_regression_constant(obj, ...)
```

Arguments

obj The object controlling which method is used
...
Arguments depending on the specific method used

---

get_regression_constant.classification_tree_learner

*Return the constant term in the logistic regression prediction at a node of a classification tree*

Description

Julia Equivalent: `IAI.get_regression_constant`

Usage

```{r}
## S3 method for class 'classification_tree_learner'
get_regression_constant(obj, node_index, ...)
```
get_regression_constant.classification_tree_multi_learner

Arguments

obj
The learner to query.
node_index
The node in the tree to query.
...
Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.0 or higher.

Examples

## Not run: iai::get_regression_constant(lnr, 1)

---

get_regression_constant.classification_tree_multi_learner

Return the constant term in the logistic regression prediction at a node of a multi-task classification tree

Description

Julia Equivalent: IAI.get_regression_constant and IAI.get_regression_constant

Usage

## S3 method for class 'classification_tree_multi_learner'
geget_regression_constant(obj, node_index, ...)

Arguments

obj
The learner to query.
node_index
The node in the tree to query.
...
Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

## Not run: iai::get_regression_constant(lnr, 1)
get_regression_constant.prescription_tree_learner

Return the constant term in the linear regression prediction at a node of a prescription tree

Description

Julia Equivalent: `IAI.get_regression_constant`

Usage

## S3 method for class 'prescription_tree_learner'
get_regression_constant(obj, node_index, treatment, ...)

Arguments

- `obj`: The learner to query.
- `node_index`: The node in the tree to query.
- `treatment`: The treatment to query.
- `...`: Refer to the Julia documentation for available parameters.

Examples

## Not run: iai::get_regression_constant(lnr, 1, "A")

get_regression_constant.regression_tree_learner

Return the constant term in the linear regression prediction at a node of a regression tree

Description

Julia Equivalent: `IAI.get_regression_constant`

Usage

## S3 method for class 'regression_tree_learner'
get_regression_constant(obj, node_index, ...)

Arguments

- `obj`: The learner to query.
- `node_index`: The node in the tree to query.
- `...`: Refer to the Julia documentation for available parameters.
get_regression_constant.regression_tree_multi_learner

Return the constant term in the linear regression prediction at a node of a multi-task regression tree

Description

Julia Equivalent: IAI.get_regression_constant and IAI.get_regression_constant

Usage

## S3 method for class 'regression_tree_multi_learner'
get_regression_constant(obj, node_index, ...)

Arguments

obj The learner to query.
node_index The node in the tree to query.
... Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

## Not run: iai::get_regression_constant(lnr, 1)

get_regression_constant.survival_tree_learner

Return the constant term in the cox regression prediction at a node of a survival tree

Description

Julia Equivalent: IAI.get_regression_constant

Usage

## S3 method for class 'survival_tree_learner'
get_regression_constant(obj, node_index, ...)

Arguments

obj The learner to query.
node_index The node in the tree to query.
... Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

## Not run: iai::get_regression_constant(lnr, 1)
**get_regression_weights.classification_tree_learner**

**Arguments**

- `obj` The learner to query.
- `node_index` The node in the tree to query.
- ... Refer to the Julia documentation for available parameters.

**IAI Compatibility**

Requires IAI version 3.0 or higher.

**Examples**

```r
## Not run: iai::get_regression_constant(lnr, 1)
```

---

**get_regression_weights**

*Generic function for returning the weights for each feature in the regression prediction at a node of a tree*

**Description**

Generic function for returning the weights for each feature in the regression prediction at a node of a tree

**Usage**

```
get_regression_weights(obj, ...)
```

**Arguments**

- `obj` The object controlling which method is used
- ... Arguments depending on the specific method used

---

**get_regression_weights.classification_tree_learner**

*Return the weights for each feature in the logistic regression prediction at a node of a classification tree*

**Description**

Julia Equivalent: `IAI.get_regression_weights`

**Usage**

```r
## S3 method for class 'classification_tree_learner'
get_regression_weights(obj, node_index, ...)
```
get_regression_weights.classification_tree_multi_learner

Arguments

  obj         The learner to query.
  node_index  The node in the tree to query.
  ...

Refer to the Julia documentation for available parameters.

IAI Compatibility

  Requires IAI version 3.0 or higher.

Examples

  ## Not run: iai::get_regression_weights(lnr, 1)

---

get_regression_weights.classification_tree_multi_learner

  Return the weights for each feature in the logistic regression prediction
  at a node of a multi-task classification tree

Description

  Julia Equivalent: IAI.get_regression_weights and IAI.get_regression_weights

Usage

  ## S3 method for class 'classification_tree_multi_learner'
  get_regression_weights(obj, node_index, ...)

Arguments

  obj         The learner to query.
  node_index  The node in the tree to query.
  ...

Refer to the Julia documentation for available parameters.

IAI Compatibility

  Requires IAI version 3.2 or higher.

Examples

  ## Not run: iai::get_regression_weights(lnr, 1)
get_regression_weights.regression_tree_learner

Return the weights for each feature in the linear regression prediction at a node of a prescription tree

Description

Julia Equivalent: IAI.get_regression_weights

Usage

## S3 method for class 'prescription_tree_learner'
get_regression_weights(obj, node_index, treatment, ...)

Arguments

- **obj**: The learner to query.
- **node_index**: The node in the tree to query.
- **treatment**: The treatment to query.
- **...**: Refer to the Julia documentation for available parameters.

Examples

## Not run: iai::get_regression_weights(lnr, 1, "A")

get_regression_weights.regression_tree_learner

Return the weights for each feature in the linear regression prediction at a node of a regression tree

Description

Julia Equivalent: IAI.get_regression_weights

Usage

## S3 method for class 'regression_tree_learner'
get_regression_weights(obj, node_index, ...)

Arguments

- **obj**: The learner to query.
- **node_index**: The node in the tree to query.
- **...**: Refer to the Julia documentation for available parameters.
get_regression_weights.regression_tree_multi_learner

Return the weights for each feature in the linear regression prediction at a node of a multi-task regression tree

Description

Julia Equivalent: IAI.get_regression_weights and IAI.get_regression_weights

Usage

## S3 method for class 'regression_tree_multi_learner'
get_regression_weights(obj, node_index, ...)

Arguments

obj The learner to query.
node_index The node in the tree to query.
... Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

## Not run: iai::get_regression_weights(lnr, 1)

get_regression_weights.survival_tree_learner

Return the weights for each feature in the cox regression prediction at a node of a survival tree

Description

Julia Equivalent: IAI.get_regression_weights

Usage

## S3 method for class 'survival_tree_learner'
get_regression_weights(obj, node_index, ...)

Arguments

obj The learner to query.
node_index The node in the tree to query.
... Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

## Not run: iai::get_regression_weights(lnr, 1)
get_roc_curve_data

Arguments

- obj: The learner to query.
- node_index: The node in the tree to query.
- ...
  Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.0 or higher.

Examples

```text
## Not run: iai::get_regression_weights(lnr, 1)
```

---

get_rich_output_params

Return the current global rich output parameter settings

Description

Julia Equivalent: `IAI.get_rich_output_params`

Usage

```text
get_rich_output_params()
```

Examples

```text
## Not run: iai::get_rich_output_params()
```

---

get_roc_curve_data

Extract the underlying data from an ROC curve

Description

ROC curves are returned by `roc_curve`, e.g. `roc_curve.classification_learner`

Usage

```text
get_roc_curve_data(curve)
```

Arguments

- curve: The curve to query.
Details

The data is returned as a list with two keys: `auc` giving the area-under-the-curve, and `coords` containing a vector of lists representing each point on the curve, each with keys `fpr` (the false positive rate), `tpr` (the true positive rate) and `threshold` (the threshold).

Julia Equivalent: `IAI.get_roc_curve_data`

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: iai::get_roc_curve_data(curve)
```

---

### get_split_categories

Return the categoric/ordinal information used in the split at a node of a tree

#### Description

Julia Equivalent: `IAI.get_split_categories`

#### Usage

```r
get_split_categories(lnr, node_index)
```

#### Arguments

- **lnr**: The learner to query.
- **node_index**: The node in the tree to query.

#### Examples

```r
## Not run: iai::get_split_categories(lnr, 1)
```
get_split_feature

Return the feature used in the split at a node of a tree

Description

Julia Equivalent: \texttt{IAI.get_split_feature}

Usage

\begin{verbatim}
get_split_feature(lnr, node_index)
\end{verbatim}

Arguments

- \texttt{lnr} The learner to query.
- \texttt{node_index} The node in the tree to query.

Examples

\begin{verbatim}
## Not run: iai::get_split_feature(lnr, 1)
\end{verbatim}

get_split_threshold

Return the threshold used in the split at a node of a tree

Description

Julia Equivalent: \texttt{IAI.get_split_threshold}

Usage

\begin{verbatim}
get_split_threshold(lnr, node_index)
\end{verbatim}

Arguments

- \texttt{lnr} The learner to query.
- \texttt{node_index} The node in the tree to query.

Examples

\begin{verbatim}
## Not run: iai::get_split_threshold(lnr, 1)
\end{verbatim}
get_split_weights

Return the weights for numeric and categoric features used in the hyperplane split at a node of a tree

Description

Julia Equivalent: IAI.get_split_weights

Usage

get_split_weights(lnr, node_index)

Arguments

lnr The learner to query.
node_index The node in the tree to query.

Examples

## Not run: iai::get_split_weights(lnr, 1)

get_stability_results

Return the trained trees in order of increasing objective value, along with their variable importance scores for each feature

Description

Julia Equivalent: IAI.get_stability_results

Usage

get_stability_results(stability)

Arguments

stability The stability analysis to query

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: iai::get_stability_results(stability)
get_survival_curve

Return the survival curve at a node of a tree

Description

Julia Equivalent: IAI.get_survival_curve

Usage

get_survival_curve(lnr, node_index, ...)

Arguments

lnr               The learner to query.
node_index        The node in the tree to query.
...               Refer to the Julia documentation for available parameters.

Examples

## Not run: iai::get_survival_curve(lnr, 1)

get_survival_curve_data

Extract the underlying data from a survival curve (as returned by predict.survival_learner or get_survival_curve)

Description

The data is returned as a list with two keys: times containing the time for each breakpoint on the curve, and coefs containing the probability for each breakpoint on the curve.

Usage

get_survival_curve_data(curve)

Arguments

curve               The curve to query.

Details

Julia Equivalent: IAI.get_survival_curve_data

Examples

## Not run: iai::get_survival_curve_data(curve)
get_survival_expected_time

Return the predicted expected survival time at a node of a tree

Description

Julia Equivalent: `IAI.get_survival_expected_time`

Usage

get_survival_expected_time(lnr, node_index, ...)

Arguments

lnr
    The learner to query.
node_index
    The node in the tree to query.
...    Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```
## Not run: iai::get_survival_expected_time(lnr, 1)
```

get_survival_hazard

Return the predicted hazard ratio at a node of a tree

Description

Julia Equivalent: `IAI.get_survival_hazard`

Usage

get_survival_hazard(lnr, node_index, ...)

Arguments

lnr
    The learner to query.
node_index
    The node in the tree to query.
...    Refer to the Julia documentation for available parameters.
get_tree

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

## Not run: iai::get_survival_hazard(lnr, 1)

get_train_errors

Extract the training objective value for each candidate tree in the comparison, where a lower value indicates a better solution

Description

Julia Equivalent: IAI.get_train_errors

Usage

get_train_errors(similarity)

Arguments

similarity The similarity comparison

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: iai::get_train_errors(similarity)

get_tree

Return a copy of the learner that uses a specific tree rather than the tree with the best training objective.

Description

Julia Equivalent: IAI.get_tree

Usage

get_tree(lnr, index)
get_upper_child

Arguments

    lnr            The original learner
    index         The index of the tree to use

IAI Compatibility

    Requires IAI version 2.2 or higher.

Examples

    ## Not run: iai::get_tree(lnr, index)

get_upper_child  Get the index of the upper child at a split node of a tree

Description

    Julia Equivalent: IAI.get_upper_child

Usage

    get_upper_child(lnr, node_index)

Arguments

    lnr            The learner to query.
    node_index     The node in the tree to query.

Examples

    ## Not run: iai::get_upper_child(lnr, 1)
glmnetcv_classifier
Learner for training GLMNet models for classification problems with cross-validation

Description
Julia Equivalent: IAI.GLMNetCVClassifier

Usage
glmnetcv_classifier(...)

Arguments
... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility
Requires IAI version 3.0 or higher.

Examples
## Not run: lnr <- iai::glmnetcv_classifier()

glmnetcv_regressor
Learner for training GLMNet models for regression problems with cross-validation

Description
Julia Equivalent: IAI.GLMNetCVRegressor

Usage
glmnetcv_regressor(...)

Arguments
... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility
Requires IAI version 2.1 or higher.
glmnetcv_survival_learner

Learner for training GLMNet models for survival problems with cross-validation

Description

Julia Equivalent: `IAI.GLMNetCVSurvivalLearner`

Usage

`glmnetcv_survival_learner(...)`

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.0 or higher.

Examples

```r
## Not run: lnr <- iai::glmnetcv_survivor()
```

grid_search

Controls grid search over parameter combinations

Description

Julia Equivalent: `IAI.GridSearch`

Usage

`grid_search(lnr, ...)`

Arguments

`lnr` The learner to use when validating.

... The parameters to validate over.
**Examples**

```r
## Not run:
grid <- iai::grid_search(
  iai::optimal_tree_classifier(
    random_seed = 1,
  ),
  max_depth = 1:5,
)

## End(Not run)
```

---

**iai_setup**

*Initialize Julia and the IAI package.*

**Description**

This function is called automatically with default parameters the first time any `iai` function is used in an R session. If custom parameters for Julia setup are required, this function must be called in every R session before calling other `iai` functions.

**Usage**

`iai_setup(...)`

**Arguments**

`...` All parameters are passed through to `JuliaCall::julia_setup`

**Examples**

```r
## Not run: iai::iai_setup()
```

---

**imputation_learner**

*Generic learner for imputing missing values*

**Description**

Julia Equivalent: `IAI.ImputationLearner`

**Usage**

`imputation_learner(method = "opt_knn", ...)`
impute

Arguments

`method` (optional) Specifies the imputation method to use.

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: lnr <- iai::imputation_learner(method = "opt_tree")
```

---

### Description

Julia Equivalent: `IAI.impute`

#### Usage

`impute(X, ...)`

#### Arguments

- `X` The dataframe in which to impute missing values.

---

### Usage

```r
impute(X, ...)
```

#### Arguments

- `X` The dataframe in which to impute missing values.

---

### Details

This function was deprecated in iai 1.7.0. This is for consistency with the IAI v3.0.0 Julia release.

#### Examples

```r
## Not run:
X <- iris
X[1, 1] <- NA
iai::impute(X)
## End(Not run)
```
impute_cv

Impute missing values using cross validation

Description

Julia Equivalent: IAI.impute_cv

Usage

impute_cv(X, ...)

Arguments

X

The dataframe in which to impute missing values.

...

Refer to the Julia documentation for available parameters.

Details

This function was deprecated in iai 1.7.0. This is for consistency with the IAI v3.0.0 Julia release.

Examples

## Not run:
X <- iris
X[1, 1] <- NA
iai::impute_cv(X, list(method = c("opt_knn", "opt_tree")))

## End(Not run)

install_julia

Download and install Julia automatically.

Description

Download and install Julia automatically.

Usage

install_julia(version = "latest", prefix = julia_default_install_dir())

Arguments

version

The version of Julia to install (e.g. "1.6.3"). Defaults to "latest", which will install the most recent stable release.

prefix

The directory where Julia will be installed. Defaults to a location determined by rappdirs::user_data_dir.
install_system_image  

Download and install the IAI system image automatically.

Description

Download and install the IAI system image automatically.

Usage

install_system_image(
  version = "latest",
  replace_default = FALSE,
  prefix = sysimage_default_install_dir(),
  accept_license = FALSE
)

Arguments

version  The version of the IAI system image to install (e.g. "2.1.0"). Defaults to "latest", which will install the most recent release.

replace_default  Whether to replace the default Julia system image with the downloaded IAI system image. Defaults to FALSE.

prefix  The directory where the IAI system image will be installed. Defaults to a location determined by rappdirs::user_data_dir.

accept_license  Set to TRUE to confirm that you agree to the End User License Agreement and skip the interactive confirmation dialog.

Examples

## Not run: iai::install_julia()
is_categoric_split  Check if a node of a tree applies a categoric split

Description
Julia Equivalent: `IAI.is_categoric_split`

Usage
```
is_categoric_split(lnr, node_index)
```

Arguments
- `lnr`: The learner to query.
- `node_index`: The node in the tree to query.

Examples
```
## Not run: iai::is_categoric_split(lnr, 1)
```

is_hyperplane_split  Check if a node of a tree applies a hyperplane split

Description
Julia Equivalent: `IAI.is_hyperplane_split`

Usage
```
is_hyperplane_split(lnr, node_index)
```

Arguments
- `lnr`: The learner to query.
- `node_index`: The node in the tree to query.

Examples
```
## Not run: iai::is_hyperplane_split(lnr, 1)
```
is_leaf

Check if a node of a tree is a leaf

Description

Julia Equivalent: IAI.is_leaf

Usage

is_leaf(lnr, node_index)

Arguments

lnr The learner to query.
node_index The node in the tree to query.

Examples

## Not run: iai::is_leaf(lnr, 1)

is_mixed_ordinal_split

Check if a node of a tree applies a mixed ordinal/categoric split

Description

Julia Equivalent: IAI.is_mixed_ordinal_split

Usage

is_mixed_ordinal_split(lnr, node_index)

Arguments

lnr The learner to query.
node_index The node in the tree to query.

Examples

## Not run: iai::is_mixed_ordinal_split(lnr, 1)
is_mixed_parallel_split

Check if a node of a tree applies a mixed parallel/categoric split

Description

Julia Equivalent: IAI.is_mixed_parallel_split

Usage

is_mixed_parallel_split(lnr, node_index)

Arguments

lnr The learner to query.
node_index The node in the tree to query.

Examples

## Not run: iai::is_mixed_parallel_split(lnr, 1)

is_ordinal_split

Check if a node of a tree applies a ordinal split

Description

Julia Equivalent: IAI.is_ordinal_split

Usage

is_ordinal_split(lnr, node_index)

Arguments

lnr The learner to query.
node_index The node in the tree to query.

Examples

## Not run: iai::is_ordinal_split(lnr, 1)
is_parallel_split

Check if a node of a tree applies a parallel split

Description

Julia Equivalent: `IAI.is_parallel_split`

Usage

```julia
is_parallel_split(lnr, node_index)
```

Arguments

- `lnr`: The learner to query.
- `node_index`: The node in the tree to query.

Examples

```julia
## Not run: iai::is_parallel_split(lnr, 1)
```

load_graphviz

Loads the Julia Graphviz library to permit certain visualizations.

Description

The library will be installed if not already present.

Usage

```julia
load_graphviz()
```

Examples

```julia
## Not run: iai::load_graphviz()
```
### mean_imputation_learner

*Learner for conducting mean imputation*

**Description**

Julia Equivalent: `IAI.MeanImputationLearner`

**Usage**

```r
mean_imputation_learner(...)```

**Arguments**

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

**Examples**

```r
## Not run: lnr <- iai::mean_imputation_learner()
```

---

### missing_goes_lower

*Check if points with missing values go to the lower child at a split node of a tree*

**Description**

Julia Equivalent: `IAI.missing_goes_lower`

**Usage**

```r
missing_goes_lower(lnr, node_index)
```

**Arguments**

- `lnr` The learner to query.
- `node_index` The node in the tree to query.

**Examples**

```r
## Not run: iai::missing_goes_lower(lnr, 1)
```
**multi_questionnaire**  
*Generic function for constructing an interactive questionnaire with multiple learners*

---

**Description**  
Generic function for constructing an interactive questionnaire with multiple learners

**Usage**  
`multi_questionnaire(obj, ...)`

**Arguments**  
- `obj`  
  The object controlling which method is used
- `...`  
  Arguments depending on the specific method used

---

**multi_questionnaire.default**  
*Construct an interactive questionnaire from multiple specified learners*

---

**Description**  
Refer to the documentation on advanced tree visualization for more information.

**Usage**  
```r  
## Default S3 method:  
multi_questionnaire(obj, ...)  
```

**Arguments**  
- `obj`  
  The questions to visualize. Refer to the Julia documentation on multi-learner visualizations for more information.
- `...`  
  Additional arguments (unused)

**Details**  
Julia Equivalent: `IAI.MultiQuestionnaire`

**IAI Compatibility**  
Requires IAI version 1.1 or higher.
### Description

Julia Equivalent: **IAI.MultiQuestionnaire**

### Usage

```r
## S3 method for class 'grid_search'
multi_questionnaire(obj, ...)```

### Arguments

- **obj**  
The grid to visualize
- **...**  
Additional arguments (unused)

### IAI Compatibility

Requires IAI version 2.0 or higher.

### Examples

```r
## Not run: iai::multi_questionnaire(grid)```
**multi_tree_plot**

Generic function for constructing an interactive tree visualization of multiple tree learners

**Description**

Generic function for constructing an interactive tree visualization of multiple tree learners

**Usage**

```
multi_tree_plot(obj, ...)
```

**Arguments**

- `obj`  The object controlling which method is used
- `...`  Arguments depending on the specific method used

---

**multi_tree_plot.default**

Construct an interactive tree visualization of multiple tree learners as specified by questions

**Description**

Refer to the documentation on advanced tree visualization for more information.

**Usage**

```
## Default S3 method:
multi_tree_plot(obj, ...)
```

**Arguments**

- `obj`  The questions to visualize. Refer to the Julia documentation on multi-learner visualizations for more information.
- `...`  Additional arguments (unused)

**Details**

Julia Equivalent: `IAI.MultiTreePlot`

**IAI Compatibility**

Requires IAI version 1.1 or higher.
multi_tree_plot.grid_search

Construct an interactive tree visualization of multiple tree learners from the results of a grid search

Description

Julia Equivalent: IAI.MultiTreePlot

Usage

## S3 method for class 'grid_search'
multi_tree_plot(obj, ...)

Arguments

obj The grid to visualize

... Additional arguments (unused)

IAI Compatibility

Requires IAI version 2.0 or higher.

Examples

## Not run: iai::multi_tree_plot(grid)
numeric_classification_reward_estimator

Learner for conducting reward estimation with numeric treatments and classification outcomes

Description

Julia Equivalent: IAI.NumericClassificationRewardEstimator

Usage

numeric_classification_reward_estimator(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: lnr <- iai::numeric_classification_reward_estimator()

---

numeric_regression_reward_estimator

Learner for conducting reward estimation with numeric treatments and regression outcomes

Description

Julia Equivalent: IAI.NumericRegressionRewardEstimator

Usage

numeric_regression_reward_estimator(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.
IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

```r
## Not run: lnr <- iai::numeric_regression_reward_estimator()
```

---

`numeric_reward_estimator`  
*Learner for conducting reward estimation with numeric treatments*

Description

This function was deprecated in iai 1.6.0, and `numeric_classification_reward_estimator()` or `numeric_classification_reward_estimator()` should be used instead.

Usage

`numeric_reward_estimator(...)`

Arguments

`...`  
Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Details

This deprecation is no longer supported as of the IAI v3 release.

IAI Compatibility

Requires IAI version 2.1 or 2.2.

Examples

```r
## Not run: lnr <- iai::numeric_reward_estimator()
```
numeric_survival_reward_estimator

Learner for conducting reward estimation with numeric treatments and survival outcomes

Description

Julia Equivalent: IAI.NumericSurvivalRewardEstimator

Usage

numeric_survival_reward_estimator(...)  

Arguments

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: lnr <- iai::numeric_survival_reward_estimator()

optimal_feature_selection_classifier

Learner for conducting Optimal Feature Selection on classification problems

Description

Julia Equivalent: IAI.OptimalFeatureSelectionClassifier

Usage

optimal_feature_selection_classifier(...)  

Arguments

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.
optimal_feature_selection_regressor

Learner for conducting Optimal Feature Selection on regression problems

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

```r
## Not run: lnr <- iai::optimal_feature_selection_regressor()
```

Description

Julia Equivalent: `IAI.OptimalFeatureSelectionRegressor`

Usage

`optimal_feature_selection_regressor(...)`

Arguments

`...` Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

```r
## Not run: lnr <- iai::optimal_feature_selection_regressor()
```
optimal_tree_classifier

Learner for training Optimal Classification Trees

Description
Julia Equivalent: \texttt{IAI.OptimalTreeClassifier}

Usage
\begin{verbatim}
optimal_tree_classifier(...)\end{verbatim}

Arguments
\begin{verbatim}
... \end{verbatim}
Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples
\begin{verbatim}
## Not run: lnr <- iai::optimal_tree_classifier()\end{verbatim}

optimal_tree_multi_classifier

Learner for training multi-task Optimal Classification Trees

Description
Julia Equivalent: \texttt{IAI.OptimalTreeMultiClassifier}

Usage
\begin{verbatim}
optimal_tree_multi_classifier(...)\end{verbatim}

Arguments
\begin{verbatim}
... \end{verbatim}
Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility
Requires IAI version 3.2 or higher.

Examples
\begin{verbatim}
## Not run: lnr <- iai::optimal_tree_multi_classifier()\end{verbatim}
optimal_tree_multi_regressor

Learner for training multi-task Optimal Regression Trees

Description

Julia Equivalent: IAI.OptimalTreeMultiRegressor

Usage

optimal_tree_multi_regressor(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

## Not run: lnr <- iai::optimal_tree_multi_regressor()

optimal_tree_policy_maximizer

Learner for training Optimal Policy Trees where the policy should aim to maximize outcomes

Description

Julia Equivalent: IAI.OptimalTreePolicyMaximizer

Usage

optimal_tree_policy_maximizer(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.0 or higher.
optimal_tree_policy_minimizer

Learner for training Optimal Policy Trees where the policy should aim to minimize outcomes

Description

Julia Equivalent: IAI.OptimalTreePolicyMinimizer

Usage

optimal_tree_policy_minimizer(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.0 or higher.

Examples

## Not run: lnr <- iai::optimal_tree_policy_maximizer()

---

optimal_tree_prescription_maximizer

Learner for training Optimal Prescriptive Trees where the prescriptions should aim to maximize outcomes

Description

Julia Equivalent: IAI.OptimalTreePrescriptionMaximizer

Usage

optimal_tree_prescription_maximizer(...)
optimal_tree_regressor

Arguments

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: lnr <- iai::optimal_tree_prescription_maximizer()
```

optimal_tree_prescription_minimizer

Learner for training Optimal Prescriptive Trees where the prescriptions should aim to minimize outcomes

Description

Julia Equivalent: `IAI.OptimalTreePrescriptionMinimizer`

Usage

`optimal_tree_prescription_minimizer(...)`

Arguments

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: lnr <- iai::optimal_tree_prescription_minimizer()
```

optimal_tree_regressor

Learner for training Optimal Regression Trees

Description

Julia Equivalent: `IAI.OptimalTreeRegressor`

Usage

`optimal_tree_regressor(...)`
optimal_tree_survival_learner

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples

## Not run: lnr <- iai::optimal_tree_regressor()

optimal_tree_survivor

Learner for training Optimal Survival Trees

Description

This function was deprecated and renamed to \texttt{optimal_tree_survival_learner()} in iai 1.3.0. This is for consistency with the IAI v2.0.0 Julia release.

Usage

optimal_tree_survivor(...)
Examples

```r
## Not run: lnr <- iai::optimal_tree_survivor()
```

---

`opt_knn_imputation_learner`

*Learner for conducting optimal k-NN imputation*

Description

Julia Equivalent: `IAI.OptKNNImputationLearner`

Usage

```r
opt_knn_imputation_learner(...)  
```

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: lnr <- iai::opt_knn_imputation_learner()
```

---

`opt_svm_imputation_learner`

*Learner for conducting optimal SVM imputation*

Description

Julia Equivalent: `IAI.OptSVMImputationLearner`

Usage

```r
opt_svm_imputation_learner(...)  
```

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: lnr <- iai::opt_svm_imputation_learner()
```
opt_tree_imputation_learner

Learner for conducting optimal tree-based imputation

Description

Julia Equivalent: IAI.OptTreeImputationLearner

Usage

opt_tree_imputation_learner(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples

## Not run: lnr <- iai::opt_tree_imputation_learner()

plot.grid_search

Plot a grid search results for Optimal Feature Selection learners

Description

Plot a grid search results for Optimal Feature Selection learners

Usage

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

Arguments

x The grid search to plot

... Additional arguments (passed to autoplot.grid_search)

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: plot(grid)
plot.roc_curve  

*Plot an ROC curve*

**Description**

Plot an ROC curve

**Usage**

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

**Arguments**

- `x` The ROC curve to plot
- `...` Additional arguments (unused)

**IAI Compatibility**

Requires IAI version 2.1 or higher.

**Examples**

```r
## Not run: plot(roc)
```

---

plot.similarity_comparison  

*Plot a similarity comparison*

**Description**

Plot a similarity comparison

**Usage**

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

**Arguments**

- `x` The similarity comparison to plot
- `...` Additional arguments (unused)
IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

```r
## Not run: plot(similarity)
```

Description

Plot a stability analysis

Usage

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

Arguments

- `x` The stability analysis to plot
- `...` Additional arguments (unused)

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

```r
## Not run: plot(stability)
```
**predict**

Generic function for returning the predictions of a model

### Description

Generic function for returning the predictions of a model

### Usage

```r
predict(obj, ...)
```

### Arguments

- `obj` The object controlling which method is used
- `...` Arguments depending on the specific method used

---

**predict.categorical_reward_estimator**

Return counterfactual rewards estimated by a categorical reward estimator for each observation in the supplied data

### Description

Julia Equivalent: `IAI.predict`

### Usage

```r
## S3 method for class 'categorical_reward_estimator'
predict(obj, X, ...)
```

### Arguments

- `obj` The learner or grid to use for estimation
- `X` The features of the data.
- `...` Additional arguments depending on the treatment and outcome types. Refer to the Julia documentation for more information.

### IAI Compatibility

Requires IAI version 2.0 or higher.

### Examples

```r
## Not run: iai::predict(lnr, X, treatments, outcomes)
```
predict.glmnetcv_learner

Return the predictions made by a GLMNet learner for each point in the features

Description

Julia Equivalent: IAI.predict

Usage

```r
## S3 method for class 'glmnetcv_learner'
predict(obj, X, fit_index = NULL, ...)
```

Arguments

- `obj`: The learner or grid to use for prediction.
- `X`: The features of the data.
- `fit_index`: The index of the fit in the path to use for prediction, defaulting to the best fit if not supplied.
- `...`: Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: iai::predict(lnr, X)
```

predict.numeric_reward_estimator

Return counterfactual rewards estimated by a numeric reward estimator for each observation in the supplied data

Description

Julia Equivalent: IAI.predict

Usage

```r
## S3 method for class 'numeric_reward_estimator'
predict(obj, X, ...)
```

- `obj`: The learner or grid to use for prediction.
- `X`: The features of the data.
- `...`: Refer to the Julia documentation for available parameters.
### predict.optimal_feature_selection_learner

**Description**

Julia Equivalent: `IAI.predict`

**Usage**

```r
## S3 method for class 'optimal_feature_selection_learner'
predict(obj, X, fit_index = NULL, ...)
```

**Arguments**

- **obj**: The learner or grid to use for prediction.
- **X**: The features of the data.
- **fit_index**: The index of the cluster to use for prediction, if the `coordinated_sparsity` parameter on the learner is `TRUE`.
- **...**: Additional arguments depending on the treatment and outcome types. Refer to the Julia documentation for available parameters.

**IAI Compatibility**

Requires IAI version 1.1 or higher.

**Examples**

```r
## Not run: iai::predict(lnr, X)
```
**predict.supervised_learner**

*Return the predictions made by a supervised learner for each point in the features*

---

**Description**

Julia Equivalent: `IAI.predict`

**Usage**

```r
## S3 method for class 'supervised_learner'
predict(obj, X, ...)
```

**Arguments**

- `obj` The learner or grid to use for prediction.
- `X` The features of the data.
- `...` Refer to the Julia documentation for available parameters.

**Examples**

```r
## Not run: iai::predict(lnr, X)
```

---

**predict.supervised_multi_learner**

*Return the predictions made by a multi-task supervised learner for each point in the features*

---

**Description**

Julia Equivalent: `IAI.predict` and `IAI.predict`

**Usage**

```r
## S3 method for class 'supervised_multi_learner'
predict(obj, X, ...)
```

**Arguments**

- `obj` The learner or grid to use for prediction.
- `X` The features of the data.
- `...` Refer to the Julia documentation for available parameters.
**IAI Compatibility**

Requires IAI version 3.2 or higher.

**Examples**

```r
## Not run: iai::predict(lnr, X)
```

---

**predict.survival_learner**

*Return the predictions made by a survival learner for each point in the features*

---

**Description**

Julia Equivalent: `IAI.predict`

**Usage**

```r
## S3 method for class 'survival_learner'
predict(obj, X, t = NULL, ...)
```

**Arguments**

- `obj` The learner or grid to use for prediction.
- `X` The features of the data.
- `t` The time for which to predict survival probability, defaulting to returning the entire survival curve if not supplied
- `...` Additional arguments (unused)

**Examples**

```r
## Not run: iai::predict(lnr, X, t = 10)
```
predict_expected_survival_time

Generic function for returning the expected survival time predicted by a model

Description

Generic function for returning the expected survival time predicted by a model

Usage

predict_expected_survival_time(obj, ...)

Arguments

obj The object controlling which method is used
...

Arguments depending on the specific method used

predict_expected_survival_time.glmnetcv_survival_learner

Return the expected survival time estimate made by a
glmnetcv_survival_learner for each point in the features.

Description

Julia Equivalent: IAI.predict_expected_survival_time

Usage

## S3 method for class 'glmnetcv_survival_learner'
predict_expected_survival_time(obj, X, fit_index = NULL, ...)

Arguments

obj The learner or grid to use for prediction.
X The features of the data.
fit_index The index of the fit in the path to use for prediction, defaulting to the best fit if not supplied.
...

Additional arguments (unused)

IAI Compatibility

Requires IAI version 3.0 or higher.
Examples

```r
## Not run: iai::predict_expected_survival_time(lnr, X)
```

#### predict_expected_survival_time.survival_curve

Return the expected survival time estimate made by a survival curve (as returned by `predict.survival_learner` or `get_survival_curve`).

**Description**

Julia Equivalent: `IAI.predict_expected_survival_time`

**Usage**

```r
## S3 method for class 'survival_curve'
predict_expected_survival_time(obj, ...)
```

**Arguments**

- `obj`: The survival curve to use for prediction.
- `...`: Additional arguments (unused)

**IAI Compatibility**

Requires IAI version 2.2 or higher.

**Examples**

```r
## Not run: iai::predict_expected_survival_time(curve)
```

#### predict_expected_survival_time.survival_learner

Return the expected survival time estimate made by a survival learner for each point in the features.

**Description**

Julia Equivalent: `IAI.predict_expected_survival_time`

**Usage**

```r
## S3 method for class 'survival_learner'
predict_expected_survival_time(obj, X, ...)
```

**Arguments**

- `obj`: The survival learner to use for prediction.
- `X`: Additional arguments (unused)

**IAI Compatibility**

Requires IAI version 2.2 or higher.

**Examples**

```r
## Not run: iai::predict_expected_survival_time(lnr, X)
```
predict_hazard

Arguments

obj The learner or grid to use for prediction.
X The features of the data.
... Additional arguments (unused)

IAI Compatibility

Requires IAI version 2.0 or higher.

Examples

```r
## Not run: iai::predict_expected_survival_time(lnr, X)
```

---

predict_hazard

Generic function for returning the hazard coefficient predicted by a model

Usage

`predict_hazard(obj, ...)`

Arguments

obj The object controlling which method is used
...
Arguments depending on the specific method used

predict_hazard.glmnetcv_survival_learner

Return the fitted hazard coefficient estimate made by a `glmnetcv_survival_learner` for each point in the features.

Description

A higher hazard coefficient estimate corresponds to a smaller predicted survival time.

Usage

```r
## S3 method for class 'glmnetcv_survival_learner'
predict_hazard(obj, X, fit_index = NULL, ...)
```
predict_hazard.survival_learner

Arguments

- obj: The learner or grid to use for prediction.
- X: The features of the data.
- fit_index: The index of the fit in the path to use for prediction, defaulting to the best fit if not supplied.
- ...: Additional arguments (unused)

Details

Julia Equivalent: IAI.predict_hazard

IAI Compatibility

Requires IAI version 1.2 or higher.

Examples

```r
## Not run: iai::predict_hazard(lnr, X)
```

predict_hazard.survival_learner

Return the fitted hazard coefficient estimate made by a survival learner for each point in the features.

Description

A higher hazard coefficient estimate corresponds to a smaller predicted survival time.

Usage

```r
## S3 method for class 'survival_learner'
predict_hazard(obj, X, ...)
```

Arguments

- obj: The learner or grid to use for prediction.
- X: The features of the data.
- ...: Additional arguments (unused)

Details

Julia Equivalent: IAI.predict_hazard

IAI Compatibility

Requires IAI version 1.2 or higher.
predict_outcomes

Examples

## Not run: iai::predict_hazard(lnr, X)

predict_outcomes  
Generic function for returning the outcomes predicted by a model under each treatment

Description

Generic function for returning the outcomes predicted by a model under each treatment

Usage

predict_outcomes(obj, ...)

Arguments

obj  The object controlling which method is used
...
Arguments depending on the specific method used

predict_outcomes.policy_learner

Return the predicted outcome for each treatment made by a policy learner for each point in the features

Description

Julia Equivalent: IAI.predict_outcomes

Usage

## S3 method for class 'policy_learner'
predict_outcomes(obj, X, rewards, ...)

Arguments

obj  The learner or grid to use for prediction.
X  The features of the data.
rewards  The estimated reward matrix for the data.
...
Additional arguments (unused)

IAI Compatibility

Requires IAI version 2.0 or higher
predict_outcomes

Return the predicted outcome for each treatment made by a prescription learner for each point in the features

Description

Julia Equivalent: IAI.predict_outcomes

Usage

## S3 method for class 'prescription_learner'
predict_outcomes(obj, X, ...)

Arguments

obj       The learner or grid to use for prediction.
X         The features of the data.
...       Additional arguments (unused)

Examples

## Not run: iai::predict_outcomes(lnr, X)

predict_proba

Generic function for returning the probabilities of class membership predicted by a model

Description

Generic function for returning the probabilities of class membership predicted by a model

Usage

predict_proba(obj, ...)

Arguments

obj       The object controlling which method is used
...       Arguments depending on the specific method used
**predict_proba.classification_learner**

Return the probabilities of class membership predicted by a classification learner for each point in the features

---

**Description**

Julia Equivalent: `IAI.predict_proba`

**Usage**

```r
## S3 method for class 'classification_learner'
predict_proba(obj, X, ...)
```

**Arguments**

- `obj`: The learner or grid to use for prediction.
- `X`: The features of the data.
- `...`: Additional arguments (unused)

**Examples**

```r
## Not run: iai::predict_proba(lnr, X)
```

---

**predict_proba.classification_multi_learner**

Return the probabilities of class membership predicted by a multi-task classification learner for each point in the features

---

**Description**

Julia Equivalent: `IAI.predict_proba` and `IAI.predict_proba`

**Usage**

```r
## S3 method for class 'classification_multi_learner'
predict_proba(obj, X, ...)
```

**Arguments**

- `obj`: The learner or grid to use for prediction.
- `X`: The features of the data.
- `...`: Additional arguments (unused)
IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

```r
## Not run: iai::predict_proba(lnr, X)
```

predict_proba.glmnetcv_classifier

*Return the probabilities of class membership predicted by a glmnetcv_classifier learner for each point in the features*

Description

Julia Equivalent: *IAI.predict_proba*

Usage

```r
## S3 method for class 'glmnetcv_classifier'
predict_proba(obj, X, fit_index = NULL, ...)
```

Arguments

- `obj` The learner or grid to use for prediction.
- `X` The features of the data.
- `fit_index` The index of the fit in the path to use for prediction, defaulting to the best fit if not supplied.
- `...` Additional arguments (unused)

IAI Compatibility

Requires IAI version 3.0 or higher.

Examples

```r
## Not run: iai::predict_proba(lnr, X)
```
predict_reward

Generic function for returning the counterfactual rewards estimated by a model under each treatment

Description

Generic function for returning the counterfactual rewards estimated by a model under each treatment

Usage

predict_reward(obj, ...)

Arguments

obj The object controlling which method is used
...

predict_reward.categorical_reward_estimator

Return counterfactual rewards estimated by a categorical reward estimator for each observation in the supplied data and predictions

Description

Julia Equivalent: IAI.predict_reward

Usage

## S3 method for class 'categorical_reward_estimator'
predict_reward(obj, X, ...)

Arguments

obj The learner or grid to use for estimation
X The features of the data.
...

IAI Compatibility

Requires IAI version 3.0 or higher.

Examples

## Not run: iai::predict_reward(lnr, X, treatments, outcomes, predictions)
### predict_shap

Calculate SHAP values for all points in the features using the learner

**Description**

Julia Equivalent: `IAI.predict_shap`

**Usage**

```r
predict_shap(lnr, X)
```

**Arguments**

- `lnr` The XGBoost learner or grid to use for prediction.
- `X` The features of the data.

---

### predict_reward

Return counterfactual rewards estimated by a numeric reward estimator for each observation in the supplied data and predictions

**Description**

Julia Equivalent: `IAI.predict_reward`

**Usage**

```r
## S3 method for class 'numeric_reward_estimator'
predict_reward(obj, X, ...)
```

**Arguments**

- `obj` The learner or grid to use for estimation
- `X` The features of the data.
- `...` Additional arguments depending on the treatment and outcome types. Refer to the Julia documentation for more information.

**IAI Compatibility**

Requires IAI version 3.0 or higher.

**Examples**

```r
## Not run: iai::predict_reward(lnr, X, treatments, outcomes, predictions)
```
predict_treatment_outcome

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

```r
## Not run: iai::predict_shap(lnr, X)
```

Description

Julia Equivalent: `IAI.predict_treatment_outcome`

Usage

```r
predict_treatment_outcome(lnr, X)
```

Arguments

- `lnr`: The learner or grid to use for prediction.
- `X`: The features of the data.

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: iai::predict_treatment_outcome(lnr, X)
```
predict_treatment_outcome_standard_error

*Return the standard error for the estimated quality of each treatment in the trained model of the learner for each point in the features*

**Description**

Julia Equivalent: IAI.predict_treatment_outcome_standard_error

**Usage**

predict_treatment_outcome_standard_error(lnr, X)

**Arguments**

- **lnr**
  - The learner or grid to use for prediction.
- **X**
  - The features of the data.

**IAI Compatibility**

Requires IAI version 3.2 or higher.

**Examples**

```r
## Not run: iai::predict_treatment_outcome_standard_error(lnr, X)
```

predict_treatment_rank

*Return the treatments in ranked order of effectiveness for each point in the features*

**Description**

Julia Equivalent: IAI.predict_treatment_rank

**Usage**

predict_treatment_rank(lnr, X)

**Arguments**

- **lnr**
  - The learner or grid to use for prediction.
- **X**
  - The features of the data.
IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```julia
## Not run: iai::predict_treatment_rank(lnr, X)
```

---

**print_path**

*Print the decision path through the learner for each sample in the features*

### Description

Julia Equivalent: `IAI.print_path`

### Usage

```julia
print_path(lnr, X, ...)
```

### Arguments

- **lnr**: The learner or grid to query.
- **X**: The features of the data.
- **...**: Refer to the Julia documentation for available parameters.

### Examples

```julia
## Not run:
iai::print_path(lnr, X)
iai::print_path(lnr, X, 1)

## End(Not run)
```
prune_trees

Use the trained trees in a learner along with the supplied validation data to determine the best value for the ‘cp’ parameter and then prune the trees according to this value.

Description

Julia Equivalent: `IAI.prune_trees!`

Usage

prune_trees(lnr, ...)

Arguments

lnr The learner to prune
...

Refer to the Julia documentation for available parameters

IAI Compatibility

Requires IAI version 3.0 or higher.

Examples

## Not run: iai::prune_trees(lnr, ...)

questionnaire

Generic function for constructing an interactive questionnaire

Description

Julia Equivalent: `IAI.Questionnaire`

Usage

questionnaire(obj, ...)

Arguments

obj The object controlling which method is used
...

Arguments depending on the specific method used
**Description**

Julia Equivalent: `IAI.Questionnaire`

**Usage**

```r
def S3 method for class 'optimal_feature_selection_learner'
questionnaire(obj, ...)
```

**Arguments**

- **obj** The learner to visualize.
- **...** Refer to the Julia documentation for available parameters.

**IAI Compatibility**

Requires IAI version 2.1 or higher.

**Examples**

```r
## Not run: iai::questionnaire(lnr)
```

---

**Description**

Julia Equivalent: `IAI.Questionnaire`

**Usage**

```r
def S3 method for class 'tree_learner'
questionnaire(obj, ...)
```

**Arguments**

- **obj** The learner to visualize.
- **...** Refer to the Julia documentation for available parameters.
**random_forest_regressor**

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

```
## Not run: iai::questionnaire(lnr)
```

---

**random_forest_classifier**

Learner for training random forests for classification problems

Description

Julia Equivalent: `IAI.RandomForestClassifier`

Usage

```
random_forest_classifier(...)  
```

Arguments

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```
## Not run: lnr <- iai::random_forest_classifier()
```

---

**random_forest_regressor**

Learner for training random forests for regression problems

Description

Julia Equivalent: `IAI.RandomForestRegressor`

Usage

```
random_forest_regressor(...)  
```

random_forest_survival_learner

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

## Not run: lnr <- iai::random_forest_regressor()

-------------------
random_forest_survival_learner

   Learner for training random forests for survival problems
-------------------

Description

Julia Equivalent: IAI.RandomForestSurvivalLearner

Usage

random_forest_survival_learner(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: lnr <- iai::random_forest_survival_learner()
rand_imputation_learner

Learner for conducting random imputation

Description

Julia Equivalent: IAI.RandImputationLearner

Usage

rand_imputation_learner(...)

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples

## Not run: lnr <- iai::rand_imputation_learner()

---

read_json

Read in a learner or grid saved in JSON format

Description

Julia Equivalent: IAI.read_json

Usage

read_json(filename)

Arguments

filename The location of the JSON file.

Examples

## Not run: obj <- iai::read_json("out.json")
refit_leaves

Refit the models in the leaves of a trained learner using the supplied data.

Description

Julia Equivalent: IAI.refit_leaves!

Usage

refit_leaves(lnr, ...)

Arguments

lnr The learner to refit
...

Refer to the Julia documentation for available parameters

IAI Compatibility

Requires IAI version 3.0 or higher.

Examples

## Not run: iai::refit_leaves(lnr, ...)

release_license

Release any IAI license held by the current session.

Description

Julia Equivalent: IAI.release_license

Usage

release_license()

IAI Compatibility

Requires IAI version 3.1 or higher.

Examples

## Not run: iai::release_license()
reset_display_label  
/reset the predicted probability displayed to be that of the predicted label when visualizing a learner

Description
Julia Equivalent: `IAI.reset_display_label!`

Usage
`reset_display_label(lnr)`

Arguments
- `lnr`  
The learner to modify.

Examples
```
## Not run: iai::reset_display_label(lnr)
```

resume_from_checkpoint  
Resume training from a checkpoint file

Description
Julia Equivalent: `IAI.resume_from_checkpoint`

Usage
`resume_from_checkpoint(checkpoint_file)`

Arguments
- `checkpoint_file`  
The location of the checkpoint file.

IAI Compatibility
Requires IAI version 3.1 or higher.

Examples
```
## Not run: obj <- iai::resume_from_checkpoint("checkpoint.json")
```
**reward_estimator**  

Learner for conducting reward estimation with categorical treatments

**Description**

This function was deprecated and renamed to `categorical_reward_estimator()` in iai 1.4.0. This is for consistency with the IAI v2.1.0 Julia release.

**Usage**

```r
reward_estimator(...)  
```

**Arguments**

...  

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

**Details**

This deprecation is no longer supported as of the IAI v3 release.

**IAI Compatibility**

Requires IAI version 2.2 or lower.

**Examples**

```r
## Not run: lnr <- iai::reward_estimator()
```

---

**roc_curve**  

Generic function for constructing an ROC curve

**Description**

Julia Equivalent: IAI.ROCCurve

**Usage**

```r
roc_curve(obj, ...)
```

**Arguments**

`obj`  
The object controlling which method is used

...  
Arguments depending on the specific method used
**roc_curve.classification_learner**

*Construct an ROC curve using a trained classification learner on the given data*

### Description

Julia Equivalent: `IAI.ROCCurve`

### Usage

```r
## S3 method for class 'classification_learner'
roc_curve(obj, X, y, ...)
```

### Arguments

- `obj`: The learner or grid to use for prediction.
- `X`: The features of the data.
- `y`: The labels of the data.
- `...`: Refer to the Julia documentation for available parameters.

### Examples

```r
## Not run: iai::roc_curve(lnr, X, y)
```

---

**roc_curve.classification_multi_learner**

*Construct an ROC curve using a trained multi-task classification learner on the given data*

### Description

Julia Equivalent: `IAI.ROCCurve` and `IAI.ROCCurve`

### Usage

```r
## S3 method for class 'classification_multi_learner'
roc_curve(obj, X, y, ...)
```

### Arguments

- `obj`: The learner or grid to use for prediction.
- `X`: The features of the data.
- `y`: The labels of the data.
- `...`: Refer to the Julia documentation for available parameters.

### Examples

```r
## Not run: iai::roc_curve(lnr, X, y)
```
IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

```r
## Not run: iai::roc_curve(lnr, X, y)
```

Description

Julia Equivalent: ```IAI.ROCCurve```

Usage

```r
## Default S3 method:
roc_curve(obj, y, positive_label = stop("positive_label is required"), ...)
```

Arguments

- `obj` The predicted probabilities for each point in the data.
- `y` The true labels of the data.
- `positive_label` The label for which probability is being predicted.
- `...` Additional arguments (unused)

IAI Compatibility

Requires IAI version 2.0 or higher.

Examples

```r
## Not run: iai::roc_curve(probs, y, positive_label=positive_label)
```
`roc_curve(glmnetcv_classifier)`

*Construct an ROC curve using a trained glmnetcv_classifier on the given data*

**Description**

Julia Equivalent: `IAI.ROCCurve`

**Usage**

```r
## S3 method for class ‘glmnetcv_classifier’
roc_curve(obj, X, y, fit_index = NULL, ...)
```

**Arguments**

- `obj`: The learner or grid to use for prediction.
- `X`: The features of the data.
- `y`: The labels of the data.
- `fit_index`: The index of the fit in the path to use for prediction, defaulting to the best fit if not supplied.
- `...`: Refer to the Julia documentation for available parameters.

**IAI Compatibility**

Requires IAI version 3.0 or higher.

**Examples**

```r
## Not run: iai::roc_curve(lnr, X, y)
```

---

`score`  
*Generic function for calculating scores*

**Description**

Generic function for calculating scores

**Usage**

```r
score(obj, ...)
```

**Arguments**

- `obj`: The object controlling which method is used
- `...`: Arguments depending on the specific method used
score.categorical_reward_estimator

*Calculate the scores for a categorical reward estimator on the given data*

**Description**

Julia Equivalent: `IAI.score`

**Usage**

```r
## S3 method for class 'categorical_reward_estimator'
score(obj, X, ...)
```

**Arguments**

- `obj`: The learner or grid to evaluate.
- `X`: The features of the data.
- `...`: Other parameters, including zero or more target vectors as required by the problem type. Refer to the Julia documentation for other available parameters.

**IAI Compatibility**

Requires IAI version 2.1 or higher.

**Examples**

```r
## Not run: iai::score(lnr, X, treatments, outcomes)
```

score.default

*Calculate the score for a set of predictions on the given data*

**Description**

Julia Equivalent: `IAI.score`

**Usage**

```r
## Default S3 method:
score(obj, predictions, truths, ...)
```

```r
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>The type of problem.</td>
</tr>
<tr>
<td>predictions</td>
<td>The predictions to evaluate.</td>
</tr>
<tr>
<td>truths</td>
<td>The true target values for these observations.</td>
</tr>
<tr>
<td>...</td>
<td>Other parameters, including the criterion. Refer to the Julia documentation for available parameters.</td>
</tr>
</tbody>
</table>

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: iai::score("regression", y_pred, y_true, criterion="mse")
```

---

score.glmnetcv_learner

*Calculate the score for a GLMNet learner on the given data*

Description

Julia Equivalent: `IAI.score`

Usage

```r
## S3 method for class 'glmnetcv_learner'
score(obj, X, ...)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>The learner or grid to evaluate.</td>
</tr>
<tr>
<td>X</td>
<td>The features of the data.</td>
</tr>
<tr>
<td>...</td>
<td>Other parameters, including zero or more target vectors as required by the problem type. <code>fit_index</code> can be used to specify the index of the fit in the path to use for prediction, defaulting to the best fit if not supplied. Refer to the Julia documentation for other available parameters.</td>
</tr>
</tbody>
</table>

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: iai::score(lnr, X, y, fit_index=1)
```
score.numeric_reward_estimator

Calculate the scores for a numeric reward estimator on the given data

Description

Julia Equivalent: IAI.score

Usage

## S3 method for class 'numeric_reward_estimator'
score(obj, X, ...)

Arguments

obj The learner or grid to evaluate.
X The features of the data.
...

Other parameters, including zero or more target vectors as required by the problem type. Refer to the Julia documentation for other available parameters.

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

## Not run: iai::score(lnr, X, treatments, outcomes)

score.optimal_feature_selection_learner

Calculate the score for an Optimal Feature Selection learner on the given data

Description

Julia Equivalent: IAI.score

Usage

## S3 method for class 'optimal_feature_selection_learner'
score(obj, X, ...)

## Not run: iai::score(lnr, X, treatments, outcomes)
score.supervised_learner

Arguments

- **obj**: The learner or grid to evaluate.
- **X**: The features of the data.
- **...**: Other parameters, including zero or more target vectors as required by the problem type. If the `coordinated_sparsity` parameter on the learner is `TRUE`, then `fit_index` must be used to specify which cluster should be used. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

```r
## Not run: iai::score(lnr, X, y, fit_index=1)
```

---

**Description**

Julia Equivalent: `IAI.score`

**Usage**

```r
## S3 method for class 'supervised_learner'
score(obj, X, ...)
```

Arguments

- **obj**: The learner or grid to evaluate.
- **X**: The features of the data.
- **...**: Other parameters, including zero or more target vectors as required by the problem type. Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: iai::score(lnr, X, y)
```
score.supervised_multi_learner

Calculate the score for a multi-task model on the given data

Description

Julia Equivalent: IAI.score and IAI.score

Usage

## S3 method for class 'supervised_multi_learner'
score(obj, X, ...)

Arguments

- obj: The learner or grid to evaluate.
- X: The features of the data.
- ...: Other parameters, including zero or more target vectors as required by the problem type. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 3.2 or higher.

Examples

## Not run: iai::score(lnr, X, y)

set_display_label

Show the probability of a specified label when visualizing a learner

Description

Julia Equivalent: IAI.set_display_label!

Usage

set_display_label(lnr, display_label)

Arguments

- lnr: The learner to modify.
- display_label: The label for which to show probabilities.
set_params

Set all supplied parameters on a learner

Description

Julia Equivalent: `IAI.set_params!`

Usage

```r
set_params(lnr, ...)
```

Arguments

- `lnr` The learner to modify.
- `...` The parameters to set on the learner.

Examples

```r
## Not run: iai::set_params(lnr, random_seed = 1)
```

set_julia_seed

Set the random seed in Julia

Description

Julia Equivalent: `Random.seed!`

Usage

```r
set_julia_seed(seed)
```

Arguments

- `seed` The seed to set

Examples

```r
## Not run: iai::set_julia_seed(1)
```

Examples

```r
## Not run: iai::set_display_label(lnr, "A")
```
set_reward_kernel_bandwidth

Save a new reward kernel bandwidth inside a learner, and return new
reward predictions generated using this bandwidth for the original
data used to train the learner.

Description

Julia Equivalent: IAI.set_reward_kernel_bandwidth!

Usage

set_reward_kernel_bandwidth(lnr, ...)

Arguments

lnr The learner to modify
...

Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: iai::set_reward_kernel_bandwidth(lnr, ...)

set_rich_output_param

Sets a global rich output parameter

Description

Julia Equivalent: IAI.set_rich_output_param!

Usage

set_rich_output_param(key, value)

Arguments

key The parameter to set.
value The value to set

Examples

## Not run: iai::set_rich_output_param("simple_layout", TRUE)
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Julia Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>set_threshold</td>
<td>For a binary classification problem, update the predicted labels in the leaves of the learner to predict a label only if the predicted probability is at least the specified threshold.</td>
<td>IAI.set_threshold!</td>
</tr>
</tbody>
</table>

**Description**

Julia Equivalent: `IAI.set_threshold!`

**Usage**

```plaintext
set_threshold(lnr, label, threshold, ...)
```

**Arguments**

- `lnr`: The learner to modify.
- `label`: The referenced label.
- `threshold`: The probability threshold above which `label` will be predicted.
- `...`: Refer to the Julia documentation for available parameters.

**Examples**

```plaintext
## Not run: iai::set_threshold(lnr, "A", 0.4)
```

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
</table>
| show_in_browser   | Generic function for showing interactive visualization in browser                                                                                                                                              |}

**Description**

Generic function for showing interactive visualization in browser

**Usage**

```plaintext
show_in_browser(obj, ...)
```

**Arguments**

- `obj`: The object controlling which method is used
- `...`: Arguments depending on the specific method used
**Description**

Julia Equivalent: `IAI.show_in_browser`

**Usage**

```r
## S3 method for class 'abstract_visualization'
show_in_browser(obj, ...)  
```

**Arguments**

- `obj`: The object to visualize.
- `...`: Refer to the Julia documentation for available parameters.

**Examples**

```r
## Not run: iai::show_in_browser(lnr)
```

---

**Description**

Julia Equivalent: `IAI.show_in_browser`

**Usage**

```r
## S3 method for class 'roc_curve'
show_in_browser(obj, ...)  
```

**Arguments**

- `obj`: The curve to visualize.
- `...`: Refer to the Julia documentation for available parameters.

**IAI Compatibility**

Requires IAI version 1.1 or higher.
show_questionnaire

Examples

## Not run: iai::show_in_browser(curve)

show_in_browser.tree_learner

Show interactive tree visualization of a tree learner in the default browser

Description

Julia Equivalent: IAI.show_in_browser

Usage

## S3 method for class 'tree_learner'
show_in_browser(obj, ...)

Arguments

obj The learner or grid to visualize.
...
Refer to the Julia documentation for available parameters.

IAI Compatibility

Showing a grid search requires IAI version 2.0 or higher.

Examples

## Not run: iai::show_in_browser(lnr)

show_questionnaire Generic function for showing interactive questionnaire in browser

Description

Generic function for showing interactive questionnaire in browser

Usage

show_questionnaire(obj, ...)

Arguments

obj The object controlling which method is used
...
Arguments depending on the specific method used
show_questionnaire.optimal_feature_selection_learner

*Show an interactive questionnaire based on an Optimal Feature Selection learner in default browser*

**Description**

Julia Equivalent: `IAI.show_questionnaire`

**Usage**

```r
## S3 method for class 'optimal_feature_selection_learner'
show_questionnaire(obj, ...)
```

**Arguments**

- `obj`: The learner or grid to visualize.
- `...`: Refer to the Julia documentation for available parameters.

**IAI Compatibility**

Requires IAI version 2.1 or higher.

**Examples**

```r
## Not run: iai::show_questionnaire(lnr)
```

---

show_questionnaire.tree_learner

*Show an interactive questionnaire based on a tree learner in default browser*

**Description**

Julia Equivalent: `IAI.show_questionnaire`

**Usage**

```r
## S3 method for class 'tree_learner'
show_questionnaire(obj, ...)
```

**Arguments**

- `obj`: The learner or grid to visualize.
- `...`: Refer to the Julia documentation for available parameters.
IAI Compatibility

Showing a grid search requires IAI version 2.0 or higher.

Examples

```r
## Not run: iai::show_questionnaire(lnr)
```

---

similarity_comparison  Conduct a similarity comparison between the final tree in a learner and all trees in a new learner to consider the tradeoff between training performance and similarity to the original tree

**Description**

Refer to the documentation on tree stability for more information.

**Usage**

```r
similarity_comparison(lnr, new_lnr, deviations)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>lnr</code></td>
<td>The original learner</td>
</tr>
<tr>
<td><code>new_lnr</code></td>
<td>The new learner</td>
</tr>
<tr>
<td><code>deviations</code></td>
<td>The deviation between the original tree and each tree in the new learner</td>
</tr>
</tbody>
</table>

**Details**

Julia Equivalent: `IAI.SimilarityComparison`

**IAI Compatibility**

Requires IAI version 2.2 or higher.

**Examples**

```r
## Not run: iai::similarity_comparison(lnr, new_lnr, deviations)
```
**single_knn_imputation_learner**

*Learner for conducting heuristic k-NN imputation*

Description

Julia Equivalent: `IAI.SingleKNNImputationLearner`

Usage

`single_knn_imputation_learner(...)`

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: lnr <- iai::single_knn_imputation_learner()
```

---

**split_data**

*Split the data into training and test datasets*

Description

Julia Equivalent: `IAI.split_data`

Usage

`split_data(task, X, ...)`

Arguments

- `task`: The type of problem.
- `X`: The features of the data.
- ... Other parameters, including zero or more target vectors as required by the problem type. Refer to the Julia documentation for available parameters.
Examples

```r
## Not run:
X <- iris[, 1:4]
y <- iris$Species
split <- iai::split_data("classification", X, y, train_proportion = 0.75)
train_X <- split$train$X
train_y <- split$train$y
test_X <- split$test$X
test_y <- split$test$y

## End(Not run)
```

---

### stability_analysis

**Conduct a stability analysis of the trees in a tree learner**

**Description**

Refer to the documentation on tree stability for more information.

**Usage**

```r
stability_analysis(lnr, ...)
```

**Arguments**

- `lnr`: The original learner
- `...`: Additional arguments (refer to Julia documentation)

**Details**

Julia Equivalent: `IAI.StabilityAnalysis`

**IAI Compatibility**

Requires IAI version 2.2 or higher.

**Examples**

```r
## Not run: iai::stability_analysis(lnr, ...)
```
transform

*Impute missing values in a dataframe using a fitted imputation model*

**Description**

Julia Equivalent: `IAI.transform`

**Usage**

```r
transform(lnr, X)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lnr</td>
<td>The learner or grid to use for imputation</td>
</tr>
<tr>
<td>X</td>
<td>The features of the data.</td>
</tr>
</tbody>
</table>

**Examples**

```r
## Not run: iai::transform(lnr, X)
```

---

transform_and_expand

*Transform features with a trained imputation learner and create adaptive indicator features to encode the missing pattern*

**Description**

Julia Equivalent: `IAI.transform_and_expand`

**Usage**

```r
transform_and_expand(lnr, X, ...)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lnr</td>
<td>The learner to use for imputation.</td>
</tr>
<tr>
<td>X</td>
<td>The dataframe in which to impute missing values.</td>
</tr>
<tr>
<td>...</td>
<td>Refer to the Julia documentation for available parameters.</td>
</tr>
</tbody>
</table>

**IAI Compatibility**

Requires IAI version 3.0 or higher.

**Examples**

```r
## Not run: lnr <- iai::transform_and_expand(lnr, X, type = "finite")
```
tree_plot

Specify an interactive tree visualization of a tree learner

Description

Julia Equivalent: `IAI.TreePlot`

Usage

```julia
tree_plot(lnr, ...)
```

Arguments

- `lnr` The learner to visualize.
- `...` Refer to the Julia documentation on advanced tree visualization for available parameters.

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

```julia
## Not run: iai::tree_plot(lnr)
```

tune_reward_kernel_bandwidth

Conduct the reward kernel bandwidth tuning procedure for a range of starting bandwidths and return the final tuned values.

Description

Julia Equivalent: `IAI.tune_reward_kernel_bandwidth`

Usage

```julia
tune_reward_kernel_bandwidth(lnr, ...)
```

Arguments

- `lnr` The learner to use for tuning the bandwidth
- `...` Refer to the Julia documentation for other parameters
variable_importance

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

```r
## Not run: iai::tune_reward_kernel_bandwidth(lnr, ...)
```

variable_importance  
Generic function for calculating variable importance

Description

Generic function for calculating variable importance

Usage

```
variable_importance(obj, ...)
```

Arguments

- `obj`: The object controlling which method is used
- `...`: Arguments depending on the specific method used

```
variable_importance.learner

Generate a ranking of the variables in a learner according to their importance during training. The results are normalized so that they sum to one.
```

Description

Julia Equivalent: `IAI.variable_importance`

Usage

```
## S3 method for class 'learner'
variable_importance(obj, ...)
```

Arguments

- `obj`: The learner to query.
- `...`: Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: iai::variable_importance(lnr, ...)
```
variable_importance.optimal_feature_selection_learner

Generate a ranking of the variables in an Optimal Feature Selection learner according to their importance during training. The results are normalized so that they sum to one.

Description

Julia Equivalent: IAI.variable_importance

Usage

## S3 method for class 'optimal_feature_selection_learner'
variable_importance(obj, fit_index = NULL, ...)

Arguments

- obj: The learner to query.
- fit_index: The index of the cluster to use for prediction, if the coordinated_sparsity parameter on the learner is TRUE.
- ...: Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

## Not run: iai::variable_importance(lnr, ...)

variable_importance.tree_learner

Generate a ranking of the variables in a tree learner according to their importance during training. The results are normalized so that they sum to one.

Description

Julia Equivalent: IAI.variable_importance

Usage

## S3 method for class 'tree_learner'
variable_importance(obj, ...)

Arguments

- obj: The learner to query.
variable_importance_similarity

Arguments

obj

The learner to query.

... Refer to the Julia documentation for available parameters.

Examples

## Not run: iai::variable_importance(lnr, ...)

variable_importance_similarity

Calculate similarity between the final tree in a tree learner with all
trees in new tree learner using variable importance scores.

Description

Julia Equivalent: IAI.variable_importance_similarity

Usage

variable_importance_similarity(lnr, new_lnr, ...)

Arguments

lnr The original learner

new_lnr The new learner

... Additional arguments (refer to Julia documentation)

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: iai::variable_importance_similarity(lnr, new_lnr)
write_booster

Write the internal booster saved in the learner to file

Description

Julia Equivalent: `IAI.write_booster`

Usage

```r
write_booster(filename, lnr)
```

Arguments

- `filename`: Where to save the output.
- `lnr`: The XGBoost learner with the booster to output.

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: iai::write_booster(file.path(tempdir(), "out.json"), lnr)
```

write_dot

Output a learner in [\textit{R}]\footnote{\url{https://www.graphviz.org/content/dot-language/}} format

Description

Julia Equivalent: `IAI.write_dot`

Usage

```r
write_dot(filename, lnr, ...)
```

Arguments

- `filename`: Where to save the output.
- `lnr`: The learner to output.
- `...`: Refer to the Julia documentation for available parameters.

Examples

```r
## Not run: iai::write_dot(file.path(tempdir(), "tree.dot"), lnr)
```
write_html

Generic function for writing interactive visualization to file

Description

Generic function for writing interactive visualization to file

Usage

write_html(filename, obj, ...)

Arguments

filename Where to save the output.
obj The object controlling which method is used
... Arguments depending on the specific method used

write_html.abstract_visualization

Output an object as an interactive browser visualization in HTML format

Description

Julia Equivalent: IAI.write_html

Usage

## S3 method for class 'abstract_visualization'
write_html(filename, obj, ...)

Arguments

filename Where to save the output.
obj The object to output.
... Refer to the Julia documentation for available parameters.

Examples

## Not run: iai::write_html(file.path(tempdir(), "out.html"), lnr)
write_html.roc_curve  
Output an ROC curve as an interactive browser visualization in HTML format

Description

Julia Equivalent: IAI.write_html

Usage

## S3 method for class 'roc_curve'
write_html(filename, obj, ...)

Arguments

filename  Where to save the output.
obj  The curve to output.
...  Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

## Not run: iai::write_html(file.path(tempdir(), "roc.html"), lnr)

write_html.tree_learner  
Output a tree learner as an interactive browser visualization in HTML format

Description

Julia Equivalent: IAI.write_html

Usage

## S3 method for class 'tree_learner'
write_html(filename, obj, ...)

Arguments

filename  Where to save the output.
obj  The tree learner to output.
...  Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 1.1 or higher.

Examples

## Not run: iai::write_html(file.path(tempdir(), "tree.html"), lnr)
write_json

Arguments

filename  Where to save the output.
obj       The learner or grid to output.
...       Refer to the Julia documentation for available parameters.

IAI Compatibility

Outputting a grid search requires IAI version 2.0 or higher.

Examples

```julia
## Not run: iai::write_html(file.path(tempdir(), "tree.html"), lnr)
```

---

_write_json  Output a learner or grid in JSON format_

Description

Julia Equivalent: IAI.write_json

Usage

write_json(filename, obj, ...)

Arguments

filename  Where to save the output.
obj       The learner or grid to output.
...       Refer to the Julia documentation for available parameters.

Examples

```julia
## Not run: iai::write_json(file.path(tempdir(), "out.json"), obj)
```
write_pdf  
*Output a learner as a PDF image*

**Description**

Before using this function, either run `load_graphviz` or ensure that `Graphviz` is installed and on the system `PATH`.

**Usage**

```plaintext
write_pdf(filename, lnr, ...)
```

**Arguments**

- `filename` : Where to save the output.
- `lnr` : The learner to output.
- `...` : Refer to the Julia documentation for available parameters.

**Details**

Julia Equivalent: `IAI.write_pdf`

**IAI Compatibility**

Requires IAI version 2.1 or higher.

**Examples**

```plaintext
## Not run: iai::write_pdf(file.path(tempdir(), "tree.pdf"), lnr)
```

---

write_png  
*Output a learner as a PNG image*

**Description**

Before using this function, either run `load_graphviz` or ensure that `Graphviz` is installed and on the system `PATH`.

**Usage**

```plaintext
write_png(filename, lnr, ...)
```
arguments

filename Where to save the output.

lnr The learner to output.

... Refer to the Julia documentation for available parameters.

details

Julia Equivalent: IAI.write_png

examples

```r
## Not run: iai::write_png(file.path(tempdir(), "tree.png"), lnr)
```

write_questionnaire Generic function for writing interactive questionnaire to file

description

Generic function for writing interactive questionnaire to file

usage

```r
write_questionnaire(filename, obj, ...)
```

arguments

filename Where to save the output.

obj The object controlling which method is used

... Arguments depending on the specific method used

write_questionnaire.optimal_feature_selection_learner

Output an Optimal Feature Selection learner as an interactive questionnaire in HTML format

description

Julia Equivalent: IAI.write_questionnaire

usage

```r
## S3 method for class 'optimal_feature_selection_learner'
write_questionnaire(filename, obj, ...)
```
Arguments

filename  Where to save the output.
obj  The learner or grid to output.
...

Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```r
## Not run: iai::write_questionnaire(file.path(tempdir(), "questionnaire.html"), lnr)
```

---

**write_questionnaire.tree_learner**

*Output a tree learner as an interactive questionnaire in HTML format*

Description

Julia Equivalent: `IAI.write_questionnaire`

Usage

```r
## S3 method for class 'tree_learner'
write_questionnaire(filename, obj, ...)
```

Arguments

filename  Where to save the output.
obj  The learner or grid to output.
...

Refer to the Julia documentation for available parameters.

IAI Compatibility

Outputting a grid search requires IAI version 2.0 or higher.

Examples

```r
## Not run: iai::write_questionnaire(file.path(tempdir(), "questionnaire.html"), lnr)
```
write_svg

Output a learner as a SVG image

Description

Before using this function, either run load_graphviz or ensure that Graphviz is installed and on the system PATH.

Usage

write_svg(filename, lnr, ...)

Arguments

- filename: Where to save the output.
- lnr: The learner to output.
- ...: Refer to the Julia documentation for available parameters.

Details

Julia Equivalent: IAI.write_svg

IAI Compatibility

Requires IAI version 2.1 or higher.

Examples

```# Not run: iai::write_svg(file.path(tempdir(), "tree.svg"), lnr)```

xgboost_classifier

Learner for training XGBoost models for classification problems

Description

Julia Equivalent: IAI.XGBoostClassifier

Usage

xgboost_classifier(...) 

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.
### xgboost_survival_learner

**Description**

Julia Equivalent: `IAI.XGBoostSurvivalLearner`

**Usage**

```r
xgboost_survival_learner(...)```

**Arguments**

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

### IAI Compatibility

Requires IAI version 2.1 or higher.

**Examples**

```r
## Not run: lsr <- iai::xgboost_survival_learner()
```

---

### xgboost_regressor

**Learner for training XGBoost models for regression problems**

**Description**

Julia Equivalent: `IAI.XGBoostRegressor`

**Usage**

```r
xgboost_regressor(...)```

**Arguments**

Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

### IAI Compatibility

Requires IAI version 2.1 or higher.

**Examples**

```r
## Not run: lnr <- iai::xgboost_regressor()
```
zero_imputation_learner

Arguments

... Use keyword arguments to set parameters on the resulting learner. Refer to the Julia documentation for available parameters.

IAI Compatibility

Requires IAI version 2.2 or higher.

Examples

## Not run: lnr <- iai::xgboost_survival_learner()
Index

acquire_license, 7
add_julia_processes, 7
all_treatment_combinations, 8
apply, 8
apply_nodes, 9
as.mixeddata, 9
autoplot.grid_search, 10, 87
autoplot.roc_curve, 11
autoplot.similarity_comparison, 11
autoplot.stability_analysis, 12
categorical_classification_reward_estimator, 13
categorical_regression_reward_estimator, 13
categorical_reward_estimator, 14
categorical_reward_estimator(), 115
categorical_survival_reward_estimator, 15
cleanup_installation, 15
clean, 16
convert_treatments_to_numeric, 16
copy_splits_and_refit_leaves, 17
decision_path, 17
delete_rich_output_param, 18
equal_propensity_estimator, 18

fit, 19
fit.grid_search, 19
fit.imputation_learner, 20, 25
fit.learner, 21
fit.optimal_feature_selection_learner, 21
fit_and_expand, 22
fit_cv, 23
fit_predict, 23
fit_predict.categorical_reward_estimator, 24
fit_predict.numeric_reward_estimator, 24
fit_transform, 25
fit_transform_cv, 26
get_best_params, 27
get_classification_label, 27
get_classification_label.classification_tree_learner, 28
get_classification_label.classification_tree_multi_learner, 28
get_classification_proba, 29
get_classification_proba.classification_tree_learner, 29
get_classification_proba.classification_tree_multi_learner, 30
get_cluster_assignments, 30
get_cluster_details, 31
get_cluster_distances, 32
get_depth, 32
get_estimation_densities, 33
get_features_used, 33
get_grid_result_details, 34
get_grid_result_summary, 35
get_grid_results, 34
get_learner, 35
get_lower_child, 36
get_machine_id, 36
get_num_fits, 37
get_num_fits.glmnetcv_learner, 37
get_num_fits.optimal_feature_selection_learner, 38
get_num_nodes, 38
get_num_samples, 39
get_params, 39
get_parent, 40
get_policy_treatment_outcome, 40
get_policy_treatment_outcome_standard_error, 41
get_policy_treatment_rank, 41

148
get_prediction_constant, 42
get_prediction_constant.glmnetcv_learner, 42
get_prediction_constant.optimal_feature_selection_learner, 43
get_prediction_weights, 44
get_prediction_weights.glmnetcv_learner, 44
get_prediction_weights.optimal_feature_selection_learner, 45
get_prescription_treatment_rank, 45
get_regression_constant, 46
get_regression_constant.classification_tree_learner, 46
get_regression_constant.classification_tree_multi_learner, 47
get_regression_constant.prescription_tree_learner, 48
get_regression_constant.regression_tree_learner, 49
get_regression_constant.regression_tree_multi_learner, 49
get_regression_constant.survival_tree_learner, 49
get_regression_weights, 50
get_regression_weights.classification_tree_learner, 50
get_regression_weights.classification_tree_multi_learner, 51
get_regression_weights.prescription_tree_learner, 52
get_regression_weights.regression_tree_learner, 52
get_regression_weights.regression_tree_multi_learner, 53
get_regression_weights.survival_tree_learner, 53
get_rich_output_params, 54
get_roc_curve_data, 54
get_split_categories, 55
get_split_feature, 56
get_split_threshold, 56
get_split_weights, 57
get_stability_results, 57
get_survival_curve, 58, 58, 96
get_survival_curve_data, 58
get_survival_expected_time, 59
get_survival_hazard, 59
get_train_errors, 60
get_tree, 60
get_upper_child, 61
glmnetcv_classifier, 62, 102, 118
glmnetcv_regressor, 62
glmnetcv_survival_learner, 63, 95, 97
glmnetcv_learner, 62
grid_search, 19, 63
i32, 19
imputation_learner, 64
impute, 65
impute_cv, 66
install_julia, 15, 66
install_system_image, 15, 67
is_categorical_split, 68
is_hyperplane_split, 68
is_leaf, 69
is_mixed_ordinal_split, 69
is_mixed_parallel_split, 70
is_ordinal_split, 70
is_parallel_split, 71
load_graphviz, 71, 142, 145
mean_imputation_learner, 72
missing_goes_lower, 72
multi_questionnaire, 73
multi_questionnaire.default, 73
multi_questionnaire.grid_search, 74
multi_tree_plot, 75
multi_tree_plot.default, 75
multi_tree_plot.grid_search, 76
numeric_classification_reward_estimator, 77
numeric_regression_reward_estimator, 77
numeric_reward_estimator, 78
numeric_survival_reward_estimator, 79
opt_knn_imputation_learner, 86
opt_svm_imputation_learner, 86
opt_tree_imputation_learner, 87
optimal_feature_selection_classifier, 79
optimal_feature_selectionRegressor, 80
optimal_tree_classifier, 81
optimal_tree_multi_classifier, 81
optimal_tree_multi_regressor, 82
optimal_tree_policy_maximizer, 82
optimal_tree_policy_minimizer, 83
optimal_tree_prescription_maximizer, 83
optimal_tree_prescription_minimizer, 84
optimal_tree_regressor, 84
optimal_tree_survival_learner, 85
optimal_tree_survival_learner(), 85
optimal_tree_survivor, 85

plot.grid_search, 87
plot.roc_curve, 88
plot.similarity_comparison, 88
plot.stability_analysis, 89
predict, 90
predict.categorical_reward_estimator, 90
predict.glmmnetcv_learner, 91
predict.numeric_reward_estimator, 91
predict.optimal_feature_selection_learner, 92
predict.supervised_learner, 93
predict.supervised_multi_learner, 93
predict.survival_learner, 58, 94, 96
predict.expected_survival_time, 95
predict.expected_survival_time.glmmnetcv_survival_learner, 95
predict.expected_survival_time.survival_curve, 96
predict.expected_survival_time.survival_learner, 96
predict_hazard, 97
predict_hazard.glmmnetcv_survival_learner, 97
predict_hazard.survival_learner, 98
predict_outcomes, 99
predict_outcomes.policy_learner, 99
predict_outcomes.prescription_learner, 100
predict_proba, 100
predict_proba.classification_learner, 101
predict_proba.classification_multi_learner, 101
predict_proba.glmmnetcv_classifier, 102
predict_reward, 103

predict_reward.categorical_reward_estimator, 103
predict_reward.numeric_reward_estimator, 104
predict_shap, 104
predict_treatment_outcome, 105
predict_treatment_outcome_standard_error, 106
predict_treatment_rank, 106
print_path, 107
prune_trees, 108
questionnaire, 108
questionnaire.optimal_feature_selection_learner, 109
questionnaire.tree_learner, 109
rand_imputation_learner, 112
random_forest_classifier, 110
random_forest_regressor, 110
random_forest_survival_learner, 111
read_json, 112
refit_leaves, 113
release_license, 113
reset_display_label, 114
resume_from_checkpoint, 114
reward_estimator, 115
roc_curve, 115, 127
roc_curve.classification_learner, 54, 116
roc_curve.classification_multi_learner, 116
roc_curve.default, 117
roc_curve.glmmnetcv_classifier, 118
score, 118
score.categorical_reward_estimator, 119
score.default, 119
score.glmmnetcv_learner, 120
score.numeric_reward_estimator, 121
score.optimal_feature_selection_learner, 121
score.supervised_learner, 122
score.supervised_multi_learner, 123
set_display_label, 123
set_julia_seed, 124
set_params, 124
set_reward_kernel_bandwidth, 125
set_rich_output_param, 125
set_threshold, 126
show_in_browser, 126
show_in_browser.abstract_visualization, 127
show_in_browser.roc_curve, 127
show_in_browser.tree_learner, 128
show_questionnaire, 128
show_questionnaire.optimal_feature_selection_learner, 129
show_questionnaire.tree_learner, 129
similarity_comparison, 130
single_knn_imputation_learner, 131
split_data, 131
stability_analysis, 132
transform, 25, 133
transform_and_expand, 133
tree_plot, 134
tune_reward_kernel_bandwidth, 134
variable_importance, 135
variable_importance.learner, 135
variable_importance.optimal_feature_selection_learner, 136
variable_importance.tree_learner, 136
variable_importance_similarity, 137
write_booster, 138
write_dot, 138
write_html, 139
write_html.abstract_visualization, 139
write_html.roc_curve, 140
write_html.tree_learner, 140
write_json, 141
write_pdf, 142
write_png, 142
write_questionnaire, 143
write_questionnaire.optimal_feature_selection_learner, 143
write_questionnaire.tree_learner, 144
write_svg, 145
xgboost_classifier, 145
xgboost_regressor, 146
xgboost_survival_learner, 146
zero_imputation_learner, 147