Package 'rkaf'

April 28, 2026

Title Kolmogorov-Arnold Fourier Networks in R

Version 0.1.0

Description Provides an R implementation of Kolmogorov-Arnold Fourier Networks using the torch backend. The package supports regression, binary classification, multiclass classification, formula and matrix interfaces, mini-batch training, validation splits, early stopping, standardization, best-model restoration, and KAF-specific diagnostics.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports stats, graphics, torch

Suggests testthat (>= 3.0.0), knitr, rmarkdown, pkgdown

Config/testthat/edition 3

URL https://github.com/gsidoine/rkaf, https://gsidoine.github.io/rkaf/

BugReports https://github.com/gsidoine/rkaf/issues

VignetteBuilder knitr

NeedsCompilation no

Author Guillaume Sidoine [aut, cre]

Maintainer Guillaume Sidoine <gjh.sidoine@gmail.com>

Repository CRAN

Date/Publication 2026-04-28 19:40:02 UTC

Contents

`extract_fourier_params` *Extract KAF Fourier parameters*

Description

Extracts the trainable Random Fourier Feature weights and biases from a KAF model.

Usage

```
extract_fourier_params(object, layer = NULL)
```

Arguments

`object`	A fitted object returned by `kaf_fit()`, or a raw `nn_kaf` torch module.
`layer`	Optional integer. If supplied, only extract parameters from this layer.

Value

A data frame with Fourier weights and biases.

`extract_kaf_scales` *Extract KAF branch scales*

Description

Extracts the learned GELU/base and Fourier branch scales from each KAF layer.

Usage

```
extract_kaf_scales(object)
```

Arguments

`object`	A fitted object returned by `kaf_fit()`, or a raw `nn_kaf` torch module.

Value

A data frame with one row per layer-feature pair.

`kaf` *Create a Kolmogorov-Arnold Fourier Network*

Description

Convenience wrapper around `nn_kaf()` using a more user-facing API.

Usage

```
kaf(
  input_dim,
  output_dim = 1,
  hidden = c(64, 64),
  num_grids = 16,
  dropout = 0,
  use_layernorm = TRUE,
  activation_expectation = 1.64,
  fourier_init_scale = 0.01
)
```

Arguments

`input_dim`	Integer. Number of input features.
`output_dim`	Integer. Number of output dimensions.
`hidden`	Integer vector. Hidden layer sizes.
`num_grids`	Integer. Number of Fourier frequencies per KAF layer.
`dropout`	Numeric. Dropout probability.
`use_layernorm`	Logical. Whether to apply layer normalization.
`activation_expectation`	Numeric. Scaling constant for Fourier initialization.
`fourier_init_scale`	Numeric. Initial scale of the Fourier branch.

Value

A torch KAF network.

kaf_fit *Fit a Kolmogorov-Arnold Fourier Network*

Description

Fits a KAF model for regression using mean squared error loss.

Usage

```
kaf_fit(
  x,
  y,
  task = c("auto", "regression", "binary", "multiclass"),
  hidden = c(64, 64),
  num_grids = 16,
  dropout = 0,
  use_layernorm = TRUE,
  fourier_init_scale = 0.01,
  epochs = 1000,
  lr = 0.001,
  batch_size = NULL,
  shuffle = TRUE,
  validation_split = 0,
  x_val = NULL,
  y_val = NULL,
  weight_decay = 0,
  standardize_x = TRUE,
  standardize_y = NULL,
  patience = NULL,
  verbose = TRUE,
  print_every = 100,
  seed = NULL,
  restore_best = TRUE,
  min_delta = 0
)
```

Arguments

x	Matrix, data frame, vector, or 2D torch tensor of predictors.
y	Vector, matrix, data frame, or torch tensor of targets.
task	Character. One of "auto", "regression", "binary", or "multiclass". With "auto", factor, character, and logical targets are treated as classification; numeric targets are treated as regression.
hidden	Integer vector. Hidden layer sizes.
num_grids	Integer. Number of Fourier frequencies per KAF layer.

dropout	Numeric. Dropout probability.
use_layernorm	Logical. Whether to apply layer normalization.
fourier_init_scale	Numeric. Initial scale of the Fourier branch.
epochs	Integer. Maximum number of training epochs.
lr	Numeric. Learning rate.
batch_size	Optional integer. Mini-batch size. If NULL, full-batch training is used.
shuffle	Logical. Whether to shuffle training rows each epoch.
validation_split	Numeric in [0, 1). Fraction of rows to reserve for validation. Ignored if x_val and y_val are supplied.
x_val	Optional validation predictors.
y_val	Optional validation targets.
weight_decay	Numeric. Adam weight decay.
standardize_x	Logical. Whether to standardize predictors using the training-set mean and standard deviation.
standardize_y	Logical or NULL. Whether to standardize regression targets using the training-set mean and standard deviation. If NULL, targets are standardized for regression and not standardized for classification. Predictions are automatically transformed back to the original target scale.
patience	Optional integer. Number of epochs without improvement before early stopping.
verbose	Logical. Whether to print progress.
print_every	Integer. Print frequency.
seed	Optional integer random seed.
restore_best	Logical. Whether to restore the best observed model state after training.
min_delta	Numeric. Minimum loss improvement required to update the best model state.

Value

An object of class "kaf_fit".

`kaf_fit_formula` *Fit a KAF model using an R formula*

Description

Fits a Kolmogorov-Arnold Fourier Network using a formula and data frame. This is a convenience wrapper around `kaf_fit()` for regression tasks.

Usage

```
kaf_fit_formula(
  formula,
  data,
  include_intercept = FALSE,
  na.action = stats::na.omit,
  ...
)
```

Arguments

formula	A model formula, such as y ~ x1 + x2.
data	A data frame.
include_intercept	Logical. Whether to keep the model-matrix intercept column. Defaults to FALSE.
na.action	Missing-value handling function passed to model.frame().
...	Additional arguments passed to kaf_fit().

Value

An object of class "kaf_fit".

---

nn_kaf *Kolmogorov-Arnold Fourier Network*

Description

Torch module implementing a stacked Kolmogorov-Arnold Fourier Network.

Usage

```
nn_kaf(
  layers,
  num_grids = 8,
  dropout = 0,
  use_layernorm = TRUE,
  activation_expectation = 1.64,
  fourier_init_scale = 0.01
)
```

Arguments

layers	Integer vector. Network architecture, including input and output dimensions. For example, c(10, 64, 64, 1).
num_grids	Integer. Number of Fourier frequencies per KAF layer.
dropout	Numeric. Dropout probability.
use_layernorm	Logical. Whether to apply layer normalization before the Fourier feature block.
activation_expectation	Numeric. Scaling constant for Fourier initialization.
fourier_init_scale	Numeric. Initial scale of the Fourier component.

Value

A torch nn_module.

---

nn_kaf_layer *Kolmogorov-Arnold Fourier Layer*

Description

Torch module implementing one KAF layer using a hybrid GELU and trainable Random Fourier Features activation.

Usage

```
nn_kaf_layer(
  input_dim,
  output_dim,
  num_grids = 8,
  dropout = 0,
  use_layernorm = TRUE,
  activation_expectation = 1.64,
  fourier_init_scale = 0.01
)
```

Arguments

input_dim	Integer. Input dimension.
output_dim	Integer. Output dimension.
num_grids	Integer. Number of Fourier frequencies.
dropout	Numeric. Dropout probability.
use_layernorm	Logical. Whether to apply layer normalization before the Fourier feature block.
activation_expectation	Numeric. Scaling constant for Fourier initialization.
fourier_init_scale	Numeric. Initial scale of the Fourier component.

Details

The layer computes a feature-wise hybrid activation of the form:

$$z = \alpha \odot GELU(x) + \beta \odot RFF(x)$$

followed by a linear projection to the output dimension.

Value

A torch nn_module.

`nn_random_fourier_features` *Random Fourier Features Layer*

Description

Torch module implementing trainable random Fourier features for Kolmogorov-Arnold Fourier Networks.

Usage

```
nn_random_fourier_features(
  input_dim,
  num_grids = 8,
  dropout = 0,
  activation_expectation = 1.64
)
```

Arguments

`input_dim`	Integer. Input dimension.
`num_grids`	Integer. Number of Fourier frequencies.
`dropout`	Numeric. Dropout probability.
`activation_expectation`	Numeric. Scaling constant for initialization.

Value

A torch nn_module.

plot.kaf_fit *Plot a fitted KAF model*

Description

Plot a fitted KAF model

Usage

```
## S3 method for class 'kaf_fit'
plot(x, type = c("loss", "fit"), newdata = NULL, y = NULL, ...)
```

Arguments

x	A fitted object returned by kaf_fit().
type	Character. Either "loss" or "fit".
newdata	Optional predictors used when type = "fit".
y	Optional observed target values used when type = "fit".
...	Additional arguments passed to base plotting functions.

Value

Invisibly returns x.

plot_kaf_scales *Plot KAF branch scales*

Description

Visualizes the learned base/GELU and Fourier branch scales for a selected KAF layer.

Usage

```
plot_kaf_scales(object, layer = 1, type = c("ratio", "branch"), ...)
```

Arguments

object	A fitted object returned by kaf_fit(), or a raw nn_kaf torch module.
layer	Integer. Layer to inspect.
type	Character. Either "ratio" or "branch".
...	Additional arguments passed to base plotting functions.

Value

Invisibly returns the extracted scale data.

predict.kaf_fit *Predict from a fitted KAF model*

Description

Predict from a fitted KAF model

Usage

```
## S3 method for class 'kaf_fit'
predict(
  object,
  newdata,
  type = c("response", "prob", "class", "link"),
  threshold = 0.5,
  as_tensor = FALSE,
  ...
)
```

Arguments

object	A fitted object returned by kaf_fit().
newdata	Matrix, data frame, vector, or 2D torch tensor.
type	Character. Prediction type. "response" returns regression predictions for regression, probabilities for classification. "prob" returns probabilities for classification. "class" returns predicted classes. "link" returns raw model logits/outputs.
threshold	Numeric. Classification threshold used for binary class predictions.
as_tensor	Logical. If TRUE, return a torch tensor where supported.
...	Unused.

Value

Predictions as a vector, matrix, factor, or torch tensor.

print.kaf_fit *Print a fitted KAF model*

Description

Print a fitted KAF model

Usage

```
## S3 method for class 'kaf_fit'
print(x, ...)
```

Arguments

x	A fitted object returned by `kaf_fit()`.
...	Unused.

Value

Invisibly returns `x`.

Index