Package 'effectplots'

November 18, 2024

Title Effect Plots

Version 0.1.0

Description High-performance implementation of various effect plots useful for regression and probabilistic classification tasks. The package includes partial dependence plots (Friedman, 2021, <doi:10.1214/aos/1013203451>), accumulated local effect plots and M-plots (both from Apley and Zhu, 2016, <doi:10.1111/rssb.12377>), as well as plots that describe the statistical associations between model response and features. It supports visualizations with either 'ggplot2' or 'plotly', and is compatible with most models, including 'Tidymodels', models wrapped in 'DALEX' explainers, or models with case weights.

License GPL (>= 3)

Depends R (>= 4.1.0)

Encoding UTF-8

RoxygenNote 7.3.2

Imports collapse, ggplot2, grDevices, patchwork, plotly, Rcpp, stats

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/mayer79/effectplots

BugReports https://github.com/mayer79/effectplots/issues

LinkingTo Rcpp

NeedsCompilation yes

Author Michael Mayer [aut, cre]

Maintainer Michael Mayer <mayermichael79@gmail.com>

Repository CRAN

Date/Publication 2024-11-18 13:50:04 UTC

Contents

`.ale` *Barebone Accumulated Local Effects (ALE)*

Description

This is a barebone implementation of Apley's ALE intended for developers. To get more information on ALE, see `ale()`.

Usage

```
.ale(
  object,
  v,
  data,
  breaks,
  right = TRUE,
  pred_fun = stats::predict,
  trafo = NULL,
  which_pred = NULL,
  bin_size = 200L,
  w = NULL,
  g = NULL,
  ...
)
```

Arguments

`object`	Fitted model.
`v`	Variable name in `data` to calculate ALE.
`data`	Matrix or data.frame.
`breaks`	Breaks for ALE calculation.

right	Should bins specified via breaks be right-closed? The default is TRUE.
pred_fun	Prediction function, by default stats::predict. The function takes three arguments (names irrelevant): object, data, and ....
trafo	How should predictions be transformed? A function or NULL (default). Examples are log (to switch to link scale) or exp (to switch from link scale to the original scale).
which_pred	If the predictions are multivariate: which column to pick (integer or column name). By default NULL (picks last column).
bin_size	Maximal number of observations used per bin. If there are more observations in a bin, bin_size indices are randomly sampled. The default is 200.
w	Optional vector with case weights.
g	For internal use. The result of qF(findInterval(...)). By default NULL.
...	Further arguments passed to pred_fun(), e.g., type = "response" in a glm() or (typically) prob = TRUE in classification models.

Value

Vector of ALE values in the same order as breaks[-length(breaks)].

References

Apley, Daniel W., and Jingyu Zhu. 2020. *Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models.* Journal of the Royal Statistical Society Series B: Statistical Methodology, 82 (4): 1059–1086. doi:10.1111/rssb.12377.

See Also

partial_dependence()

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
v <- "Sepal.Width"
.ale(fit, v, data = iris, breaks = seq(2, 4, length.out = 5))
```

.pd *Barebone Partial Dependence*

Description

This is a barebone implementation of Friedman's partial dependence intended for developers. To get more information on partial dependence, see partial_dependence().

Usage

```
.pd(
  object,
  v,
  data,
  grid,
  pred_fun = stats::predict,
  trafo = NULL,
  which_pred = NULL,
  w = NULL,
  ...
)
```

Arguments

object	Fitted model.
v	Variable name in data to calculate partial dependence.
data	Matrix or data.frame.
grid	Vector or factor of values to calculate partial dependence for.
pred_fun	Prediction function, by default stats::predict. The function takes three arguments (names irrelevant): object, data, and ....
trafo	How should predictions be transformed? A function or NULL (default). Examples are log (to switch to link scale) or exp (to switch from link scale to the original scale).
which_pred	If the predictions are multivariate: which column to pick (integer or column name). By default NULL (picks last column).
w	Optional vector with case weights.
...	Further arguments passed to pred_fun(), e.g., type = "response" in a glm() or (typically) prob = TRUE in classification models.

Value

Vector of partial dependence values in the same order as grid.

References

Friedman, Jerome H. 2001, *Greedy Function Approximation: A Gradient Boosting Machine.* Annals of Statistics 29 (5): 1189-1232. doi:10.1214/aos/1013203451.

See Also

partial_dependence()

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
.pd(fit, "Sepal.Width", data = iris, grid = hist(iris$Sepal.Width)$mids)
.pd(fit, "Species", data = iris, grid = levels(iris$Species))
```

ale *Accumulated Local Effects (ALE)*

Description

Calculates ALE for one or multiple `X` variables.

The concept of ALE was introduced in Apley et al. (2020) as an alternative to partial dependence (PD). The Ceteris Paribus clause behind PD is a blessing and a curse at the same time:

- Blessing: The interpretation is easy and similar to what we know from linear regression (just averaging out interaction effects).
- Curse: The model is applied to very unlikely or even impossible feature combinations, especially with strongly dependent features.

ALE fixes the curse as follows: Partial dependence is calculated for the lower and upper endpoint of a bin, using all (or a sample) of observations falling into this bin. Its slope provides the *local effect* over the bin. This is repeated for all bins, and the values are *accumulated*. Since the resulting sum starts at 0, one typically shifts the result vertically, e.g., to the average prediction. This is not done by `ale()`, however.

The function is a convenience wrapper around `feature_effects()`, which calls the barebone implementation `.ale()` to calculate ALE. The ALE values calculated by `feature_effects()` are vertically shifted to the same (weighted) average than the partial dependence curve, for optimal comparability.

Usage

```
ale(object, ...)

## Default S3 method:
ale(
  object,
  v,
  data,
  pred_fun = stats::predict,
  trafo = NULL,
  which_pred = NULL,
  w = NULL,
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  ale_n = 50000L,
  ale_bin_size = 200L,
  seed = NULL,
  ...
)
```

```
## S3 method for class 'ranger'
ale(
  object,
  v,
  data,
  pred_fun = NULL,
  trafo = NULL,
  which_pred = NULL,
  w = NULL,
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  ale_n = 50000L,
  ale_bin_size = 200L,
  seed = NULL,
  ...
)

## S3 method for class 'explainer'
ale(
  object,
  v = colnames(data),
  data = object$data,
  pred_fun = object$predict_function,
  trafo = NULL,
  which_pred = NULL,
  w = object$weights,
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  ale_n = 50000L,
  ale_bin_size = 200L,
  seed = NULL,
  ...
)
```

Arguments

object	Fitted model.
...	Further arguments passed to pred_fun(), e.g., type = "response" in a glm() or (typically) prob = TRUE in classification models.
v	Vector of variable names to calculate statistics.
data	Matrix or data.frame.
pred_fun	Prediction function, by default stats::predict. The function takes three arguments (names irrelevant): object, data, and ....

trafo	How should predictions be transformed? A function or NULL (default). Examples are log (to switch to link scale) or exp (to switch from link scale to the original scale).
which_pred	If the predictions are multivariate: which column to pick (integer or column name). By default NULL (picks last column).
w	Optional vector with case weights. Can also be a column name in data.
breaks	An integer, vector, string or function specifying the bins of the numeric X variables as in graphics::hist(). The default is "Sturges". To allow varying values of breaks across variables, it can be a list of the same length as v, or a *named* list with breaks for certain variables.
right	Should bins be right-closed? The default is TRUE. Vectorized over v. Only relevant for numeric X.
discrete_m	Numeric X variables with up to this number of unique values should not be binned and treated as a factor (after calculating partial dependence) The default is 5. Vectorized over v.
outlier_iqr	Outliers of a numeric X are capped via the boxplot rule, i.e., outside outlier_iqr * IQR from the quartiles. The default is 2 is more conservative than the usual rule to account for right-skewed distributions. Set to 0 or Inf for no capping. Note that at most 10k observations are sampled to calculate quartiles. Vectorized over v.
ale_n	Size of the data used for calculating ALE. The default is 50000. For larger data (and w), ale_n rows are randomly sampled. Each variable specified by v uses the same subsample. Set to 0 to omit.
ale_bin_size	Maximal number of observations used per bin for ALE calculations. If there are more observations in a bin, ale_bin_size indices are randomly sampled. The default is 200. Applied after subsampling regarding ale_n.
seed	Optional random seed (an integer) used for: • ALE: select background data if n > ale_n and for bins > ale_bin_size. • Capping X: quartiles are selected based on 10k observations.

Value

A list (of class "EffectData") with a data.frame of statistics per feature. Use single bracket subsetting to select part of the output.

Methods (by class)

- ale(default): Default method.
- ale(ranger): Default method.
- ale(explainer): Default method.

References

Apley, Daniel W., and Jingyu Zhu. 2020. *Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models.* Journal of the Royal Statistical Society Series B: Statistical Methodology, 82 (4): 1059–1086. doi:10.1111/rssb.12377.

See Also

feature_effects(), .ale()

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
M <- ale(fit, v = "Petal.Length", data = iris)
M |> plot()

M2 <- ale(fit, v = colnames(iris)[-1], data = iris, breaks = 5)
plot(M2, share_y = "all")  # Only numeric variables shown
```

---

average_observed *Average Observed*

Description

Calculates average observed y values over the values of one or multiple X variables. This describes the statistical association between y and potential model features.

Usage

```
average_observed(
  X,
  y,
  w = NULL,
  x_name = "x",
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  seed = NULL,
  ...
)
```

Arguments

X	A vector, matrix, or data.frame with variable(s) to be shown on the x axis.
y	A numeric vector of observed responses.
w	An optional numeric vector of weights.
x_name	If X is a vector: what is the name of the variable? By default "x".
breaks	An integer, vector, string or function specifying the bins of the numeric X variables as in graphics::hist(). The default is "Sturges". To allow varying values of breaks across variables, it can be a list of the same length as v, or a *named* list with breaks for certain variables.

right	Should bins be right-closed? The default is TRUE. Vectorized over v. Only relevant for numeric X.
discrete_m	Numeric X variables with up to this number of unique values should not be binned and treated as a factor (after calculating partial dependence) The default is 5. Vectorized over v.
outlier_iqr	Outliers of a numeric X are capped via the boxplot rule, i.e., outside outlier_iqr * IQR from the quartiles. The default is 2 is more conservative than the usual rule to account for right-skewed distributions. Set to 0 or Inf for no capping. Note that at most 10k observations are sampled to calculate quartiles. Vectorized over v.
seed	Optional random seed (an integer) used for capping X based on quantiles calculated from a subsample of 10k observations.
...	Currently unused.

Details

The function is a convenience wrapper around feature_effects().

Value

A list (of class "EffectData") with a data.frame of statistics per feature. Use single bracket subsetting to select part of the output.

See Also

feature_effects()

Examples

```
M <- average_observed(iris$Species, y = iris$Sepal.Length)
M
M |> plot()

# Or multiple potential features X
average_observed(iris[2:5], y = iris[, 1], breaks = 5) |>
  plot()
```

---

average_predicted *Average Predictions*

Description

Calculates average predictions over the values of one or multiple X variables. Shows the combined effect of a feature and other (correlated) features.

Usage

```
average_predicted(
  X,
  pred,
  w = NULL,
  x_name = "x",
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  seed = NULL,
  ...
)
```

Arguments

X	A vector, matrix, or data.frame with variable(s) to be shown on the x axis.
pred	A numeric vector of predictions.
w	An optional numeric vector of weights.
x_name	If X is a vector: what is the name of the variable? By default "x".
breaks	An integer, vector, string or function specifying the bins of the numeric X variables as in graphics::hist(). The default is "Sturges". To allow varying values of breaks across variables, it can be a list of the same length as v, or a *named* list with breaks for certain variables.
right	Should bins be right-closed? The default is TRUE. Vectorized over v. Only relevant for numeric X.
discrete_m	Numeric X variables with up to this number of unique values should not be binned and treated as a factor (after calculating partial dependence) The default is 5. Vectorized over v.
outlier_iqr	Outliers of a numeric X are capped via the boxplot rule, i.e., outside outlier_iqr * IQR from the quartiles. The default is 2 is more conservative than the usual rule to account for right-skewed distributions. Set to 0 or Inf for no capping. Note that at most 10k observations are sampled to calculate quartiles. Vectorized over v.
seed	Optional random seed (an integer) used for capping X based on quantiles calculated from a subsample of 10k observations.
...	Currently unused.

Details

The function is a convenience wrapper around feature_effects().

Value

A list (of class "EffectData") with a data.frame of statistics per feature. Use single bracket subsetting to select part of the output.

References

Apley, Daniel W., and Jingyu Zhu. 2016. *Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models.* Journal of the Royal Statistical Society Series B: Statistical Methodology, 82 (4): 1059–1086. doi:10.1111/rssb.12377.

See Also

`feature_effects()`

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
M <- average_predicted(iris[2:5], pred = predict(fit, iris), breaks = 5)
M
M |> plot()
```

---

bias *Bias / Average Residuals*

Description

Calculates average residuals (= bias) over the values of one or multiple `X` variables.

Usage

```
bias(
  X,
  resid,
  w = NULL,
  x_name = "x",
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  seed = NULL,
  ...
)
```

Arguments

`X`	A vector, matrix, or data.frame with variable(s) to be shown on the x axis.
`resid`	A numeric vector of residuals, i.e., y - pred.
`w`	An optional numeric vector of weights.
`x_name`	If `X` is a vector: what is the name of the variable? By default "x".

breaks	An integer, vector, string or function specifying the bins of the numeric X variables as in graphics::hist(). The default is "Sturges". To allow varying values of breaks across variables, it can be a list of the same length as v, or a *named* list with breaks for certain variables.
right	Should bins be right-closed? The default is TRUE. Vectorized over v. Only relevant for numeric X.
discrete_m	Numeric X variables with up to this number of unique values should not be binned and treated as a factor (after calculating partial dependence) The default is 5. Vectorized over v.
outlier_iqr	Outliers of a numeric X are capped via the boxplot rule, i.e., outside outlier_iqr * IQR from the quartiles. The default is 2 is more conservative than the usual rule to account for right-skewed distributions. Set to 0 or Inf for no capping. Note that at most 10k observations are sampled to calculate quartiles. Vectorized over v.
seed	Optional random seed (an integer) used for capping X based on quantiles calculated from a subsample of 10k observations.
...	Currently unused.

Details

The function is a convenience wrapper around feature_effects().

Value

A list (of class "EffectData") with a data.frame of statistics per feature. Use single bracket subsetting to select part of the output.

See Also

feature_effects()

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
M <- bias(iris[2:5], resid = fit$residuals, breaks = 5)
M |> update(sort_by = "resid_mean") |> plot(share_y = "all")
```

---

effect_importance *Variable Importance*

Description

Extracts from an "EffectData" object a simple variable importance measure, namely the (bin size weighted) variance of the partial dependence values, or of any other calculated statistic (e.g., "pred_mean" or "y_mean"). It can be used via update.EffectData(, sort_by = "pd") to sort the variables in decreasing importance. Note that this measure captures only the main effect strength. If the importance is calculated with respect to "pd", it is closely related to the suggestion of Greenwell et al. (2018).

Usage

```
effect_importance(x, by = NULL)
```

Arguments

x	Object of class "EffectData".
by	The statistic used to calculate the variance for. One of 'pd', 'pred_mean', 'y_mean', 'resid_mean', or 'ale' (if available). The default is NULL, which picks the first available statistic from above list.

Value

A named vector of importance values of the same length as x.

References

Greenwell, Brandon M., Bradley C. Boehmke, and Andrew J. McCarthy. 2018. *A Simple and Effective Model-Based Variable Importance Measure.* arXiv preprint. https://arxiv.org/abs/1805.04755.

See Also

update.EffectData()

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
M <- feature_effects(fit, v = colnames(iris)[-1], data = iris)
effect_importance(M)
```

---

feature_effects *Feature Effects*

Description

This is the main function of the package. By default, it calculates the following statistics per feature X over values/bins:

- "y_mean": Average observed y values. Used to assess descriptive associations between response and features.
- "pred_mean": Average predictions. Corresponds to "M Plots" (from "marginal") in Apley (2020). Shows the combined effect of X and other (correlated) features. The difference to average observed y values shows model bias.
- "resid_mean": Average residuals. Calculated when both y and predictions are available. Useful to study model bias.
- "pd": Partial dependence (Friedman, 2001): See partial_dependence(). Evaluated at bin averages, not at bin midpoints.

- "ale": Accumulated local effects (Apley, 2020): See `ale()`. Only for numeric X.

Additionally, corresponding counts/weights are calculated, and standard deviations of observed y and residuals.

Numeric X with more than `discrete_m = 5` disjoint values are binned as in `graphics::hist()` via `breaks`. Before calculating bins, outliers are capped at +-2 IQR from the quartiles.

All averages and standard deviation are weighted by optional weights `w`.

If you need only one specific statistic, you can use the simplified APIs of

- `average_observed()`,
- `average_predicted()`,
- `bias()`,
- `partial_dependence()`, and
- `ale()`.

Usage

```
feature_effects(object, ...)

## Default S3 method:
feature_effects(
  object,
  v,
  data,
  y = NULL,
  pred = NULL,
  pred_fun = stats::predict,
  trafo = NULL,
  which_pred = NULL,
  w = NULL,
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  calc_pred = TRUE,
  pd_n = 500L,
  ale_n = 50000L,
  ale_bin_size = 200L,
  seed = NULL,
  ...
)

## S3 method for class 'ranger'
feature_effects(
  object,
  v,
  data,
```

```
  y = NULL,
  pred = NULL,
  pred_fun = NULL,
  trafo = NULL,
  which_pred = NULL,
  w = NULL,
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  calc_pred = TRUE,
  pd_n = 500L,
  ale_n = 50000L,
  ale_bin_size = 200L,
  ...
)

## S3 method for class 'explainer'
feature_effects(
  object,
  v = colnames(data),
  data = object$data,
  y = object$y,
  pred = NULL,
  pred_fun = object$predict_function,
  trafo = NULL,
  which_pred = NULL,
  w = object$weights,
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  calc_pred = TRUE,
  pd_n = 500L,
  ale_n = 50000L,
  ale_bin_size = 200L,
  ...
)
```

Arguments

object	Fitted model.
...	Further arguments passed to pred_fun(), e.g., type = "response" in a glm() or (typically) prob = TRUE in classification models.
v	Vector of variable names to calculate statistics.
data	Matrix or data.frame.
y	Numeric vector with observed values of the response. Can also be a column name in data. Omitted if NULL (default).

pred — Numeric vector with predictions. If NULL, it is calculated as pred_fun(object, data, ...). Used to save time if d() is to be called multiple times.

pred_fun — Prediction function, by default stats::predict. The function takes three arguments (names irrelevant): object, data, and ....

trafo — How should predictions be transformed? A function or NULL (default). Examples are log (to switch to link scale) or exp (to switch from link scale to the original scale).

which_pred — If the predictions are multivariate: which column to pick (integer or column name). By default NULL (picks last column).

w — Optional vector with case weights. Can also be a column name in data.

breaks — An integer, vector, string or function specifying the bins of the numeric X variables as in graphics::hist(). The default is "Sturges". To allow varying values of breaks across variables, it can be a list of the same length as v, or a *named* list with breaks for certain variables.

right — Should bins be right-closed? The default is TRUE. Vectorized over v. Only relevant for numeric X.

discrete_m — Numeric X variables with up to this number of unique values should not be binned and treated as a factor (after calculating partial dependence) The default is 5. Vectorized over v.

outlier_iqr — Outliers of a numeric X are capped via the boxplot rule, i.e., outside outlier_iqr * IQR from the quartiles. The default is 2 is more conservative than the usual rule to account for right-skewed distributions. Set to 0 or Inf for no capping. Note that at most 10k observations are sampled to calculate quartiles. Vectorized over v.

calc_pred — Should predictions be calculated? Default is TRUE. Only relevant if pred = NULL.

pd_n — Size of the data used for calculating partial dependence. The default is 500. For larger data (and w), pd_n rows are randomly sampled. Each variable specified by v uses the same subsample. Set to 0 to omit.

ale_n — Size of the data used for calculating ALE. The default is 50000. For larger data (and w), ale_n rows are randomly sampled. Each variable specified by v uses the same subsample. Set to 0 to omit.

ale_bin_size — Maximal number of observations used per bin for ALE calculations. If there are more observations in a bin, ale_bin_size indices are randomly sampled. The default is 200. Applied after subsampling regarding ale_n.

seed — Optional random seed (an integer) used for:

- Partial dependence: select background data if n > pd_n.
- ALE: select background data if n > ale_n and for bins > ale_bin_size.
- Capping X: quartiles are selected based on 10k observations.

Value

A list (of class "EffectData") with a data.frame of statistics per feature. Use single bracket subsetting to select part of the output.

Methods (by class)

- `feature_effects(default)`: Default method.
- `feature_effects(ranger)`: Method for "ranger" models.
- `feature_effects(explainer)`: Method for DALEX "explainer".

References

1. Molnar, Christoph. 2019. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. https://christophm.github.io/interpretable-ml-book/.
2. Friedman, Jerome H. 2001, *Greedy Function Approximation: A Gradient Boosting Machine.* Annals of Statistics 29 (5): 1189-1232. doi:10.1214/aos/1013203451.3.
3. Apley, Daniel W., and Jingyu Zhu. 2016. *Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models.* Journal of the Royal Statistical Society Series B: Statistical Methodology, 82 (4): 1059–1086. doi:10.1111/rssb.12377.

See Also

`plot.EffectData()`, `update.EffectData()`, `partial_dependence()`, `ale()`, average_observed, `average_predicted()`, `bias()`

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
xvars <- colnames(iris)[2:5]
M <- feature_effects(fit, v = xvars, data = iris, y = "Sepal.Length", breaks = 5)
M
M |> update(sort = "pd") |> plot(share_y = "all")
```

`partial_dependence` *Partial Dependence*

Description

Calculates PD for one or multiple `X` variables.

PD was introduced by Friedman (2001) to study the (main) effects of a ML model. PD of a model f and variable `X` at a certain value g is derived by replacing the `X` values in a reference `data` by g, and then calculating the average prediction of f over this modified data. This is done for different g to see how the average prediction of f changes in `X`, keeping all other feature values constant (Ceteris Paribus).

This function is a convenience wrapper around `feature_effects()`, which calls the barebone implementation `.pd()` to calculate PD. As grid points, it uses the arithmetic mean of `X` per bin (specified by `breaks`), and eventually weighted by `w`.

Usage

```
partial_dependence(object, ...)

## Default S3 method:
partial_dependence(
  object,
  v,
  data,
  pred_fun = stats::predict,
  trafo = NULL,
  which_pred = NULL,
  w = NULL,
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  pd_n = 500L,
  seed = NULL,
  ...
)

## S3 method for class 'ranger'
partial_dependence(
  object,
  v,
  data,
  pred_fun = NULL,
  trafo = NULL,
  which_pred = NULL,
  w = NULL,
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  pd_n = 500L,
  seed = NULL,
  ...
)

## S3 method for class 'explainer'
partial_dependence(
  object,
  v = colnames(data),
  data = object$data,
  pred_fun = object$predict_function,
  trafo = NULL,
  which_pred = NULL,
  w = object$weights,
```

```
  breaks = "Sturges",
  right = TRUE,
  discrete_m = 5L,
  outlier_iqr = 2,
  pd_n = 500L,
  seed = NULL,
  ...
)
```

Arguments

object	Fitted model.
...	Further arguments passed to pred_fun(), e.g., type = "response" in a glm() or (typically) prob = TRUE in classification models.
v	Vector of variable names to calculate statistics.
data	Matrix or data.frame.
pred_fun	Prediction function, by default stats::predict. The function takes three arguments (names irrelevant): object, data, and ....
trafo	How should predictions be transformed? A function or NULL (default). Examples are log (to switch to link scale) or exp (to switch from link scale to the original scale).
which_pred	If the predictions are multivariate: which column to pick (integer or column name). By default NULL (picks last column).
w	Optional vector with case weights. Can also be a column name in data.
breaks	An integer, vector, string or function specifying the bins of the numeric X variables as in graphics::hist(). The default is "Sturges". To allow varying values of breaks across variables, it can be a list of the same length as v, or a *named* list with breaks for certain variables.
right	Should bins be right-closed? The default is TRUE. Vectorized over v. Only relevant for numeric X.
discrete_m	Numeric X variables with up to this number of unique values should not be binned and treated as a factor (after calculating partial dependence) The default is 5. Vectorized over v.
outlier_iqr	Outliers of a numeric X are capped via the boxplot rule, i.e., outside outlier_iqr * IQR from the quartiles. The default is 2 is more conservative than the usual rule to account for right-skewed distributions. Set to 0 or Inf for no capping. Note that at most 10k observations are sampled to calculate quartiles. Vectorized over v.
pd_n	Size of the data used for calculating partial dependence. The default is 500. For larger data (and w), pd_n rows are randomly sampled. Each variable specified by v uses the same subsample. Set to 0 to omit.
seed	Optional random seed (an integer) used for: • Partial dependence: select background data if n > pd_n. • Capping X: quartiles are selected based on 10k observations.

Value

A list (of class "EffectData") with a data.frame of statistics per feature. Use single bracket subsetting to select part of the output.

Methods (by class)

- `partial_dependence(default)`: Default method.
- `partial_dependence(ranger)`: Default method.
- `partial_dependence(explainer)`: Default method.

References

Friedman, Jerome H. 2001, *Greedy Function Approximation: A Gradient Boosting Machine.* Annals of Statistics 29 (5): 1189-1232. doi:10.1214/aos/1013203451.

See Also

`feature_effects()`, `.pd()`, `ale()`.

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
M <- partial_dependence(fit, v = "Species", data = iris)
M |> plot()

M2 <- partial_dependence(fit, v = colnames(iris)[-1], data = iris)
plot(M2, share_y = "all")
```

---

plot.EffectData *Plots "EffectData" Object*

Description

Versatile plot function for an "EffectData" object. By default, all calculated statistics (except "resid_mean") are shown. To select certain statistics, use the `stats` argument. Set `plotly = TRUE` for interactive plots.

Usage

```
## S3 method for class 'EffectData'
plot(
  x,
  stats = NULL,
  ncol = grDevices::n2mfrow(length(x))[2L],
  byrow = TRUE,
  share_y = c("no", "all", "rows", "cols"),
  ylim = NULL,
```

```
  cat_lines = TRUE,
  num_points = FALSE,
  title = "",
  subplot_titles = TRUE,
  ylab = NULL,
  legend_labels = NULL,
  interval = c("no", "ci", "ciw", "sd"),
  ci_level = 0.95,
  colors = getOption("effectplots.colors"),
  fill = getOption("effectplots.fill"),
  alpha = 1,
  bar_height = 1,
  bar_width = 0.7,
  bar_measure = c("weight", "N"),
  wrap_x = 10,
  rotate_x = 0,
  plotly = getOption("effectplots.plotly"),
  ...
)
```

Arguments

x	An object of class "EffectData".
stats	Vector of statistics to show. The default NULL equals either c("y_mean", "pred_mean", "pd", "ale"), or "resid_mean" (when x results from bias()). Only *available* statistics are shown. Additionally, this argument controls the order used to plot the lines.
ncol	Number of columns of the plot layout, by default grDevices::n2mfrow(length(x))[2L]. Only relevant for multiple plots.
byrow	Should plots be placed by row? Default is TRUE. Only for multiple plots.
share_y	Should y axis be shared across subplots? The default is "no". Other choices are "all", "rows", and "cols". Note that this currently does not take into account error bars/ribbons. Has mo effect if ylim is passed. Only for multiple plots.
ylim	A vector of length 2 with manual y axis limits, or a list thereof.
cat_lines	Show lines for non-numeric features. Default is TRUE.
num_points	Show points for numeric features. Default is FALSE.
title	Overall plot title, by default "" (no title).
subplot_titles	Should variable names be shown as subplot titles? Default is TRUE. Only for multiple plots.
ylab	Label of the y axis. The default NULL automatically derives a reasonable name.
legend_labels	Vector of legend labels in the same order as the statistics plotted, or NULL (default).
interval	What intervals should be shown for observed y and residuals? One of • "no" (default), • "ci": Z confidence intervals using sqrt(N) as standard error of the mean,

- "ciw": Like "ci", but using sqrt(weight) as standard error of the mean, or
- "sd": standard deviations. Ribbons for numeric X, error bars for categorical X.

`ci_level` The nominal level of the Z confidence intervals (only when `error` equals "ci" or "ciw"). The default is 0.95.

`colors` Vector of line/point colors of sufficient length. By default, a color blind friendly palette from "ggthemes". To change globally, set `options(effectplots.colors = new colors)`.

`fill` Fill color of bars. The default equals "lightgrey". To change globally, set `options(effectplots.fill = new color)`.

`alpha` Alpha transparency of lines and points. Default is 1.

`bar_height` Relative bar height (default 1). Set to 0 for no bars.

`bar_width` Relative bar width of non-numeric features, by default 0.7.

`bar_measure` What should bars represent? Either "weight" (default) or "N".

`wrap_x` Should categorical x axis labels be wrapped after this length? The default is 10. Set to 0 for no wrapping. Vectorized over `x`. Only for "ggplot2" backend.

`rotate_x` Should categorical xaxis labels be rotated by this angle? The default is 0 (no rotation). Vectorized over `x`. Only for "ggplot2" backend.

`plotly` Should 'plotly' be used? The default is `FALSE` ('ggplot2' with 'patchwork'). Use `options(effectplots.plotly = TRUE)` to change globally.

`...` Passed to `patchwork::plot_layout()` or `plotly::subplot()`. Typically not used.

Value

If a single plot, an object of class "ggplot" or "plotly". Otherwise, an object of class "patchwork", or a "plotly" subplot.

See Also

`feature_effects()`, `average_observed()`, `average_predicted()`, `partial_dependence()`, `bias()`, `ale()`

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
xvars <- colnames(iris)[-1]
M <- feature_effects(fit, v = xvars, data = iris, y = "Sepal.Length", breaks = 5)
plot(M, share_y = "all")
plot(M, stats = c("pd", "ale"), legend_labels = c("PD", "ALE"))
plot(M, stats = "resid_mean", share_y = "all", interval = "ci")
```

update.EffectData *Update "EffectData" Object*

Description

Updates an "EffectData" object by

- sorting the variables by their importance, see `effect_importance()`,
- collapsing levels of categorical variables with many levels,
- dropping small bins, or
- dropping bins with missing name.

Except for `sort_by`, all arguments are vectorized, i.e., you can pass a vector or list of the same length as `object`.

Usage

```
## S3 method for class 'EffectData'
update(
  object,
  sort_by = c("no", "pd", "pred_mean", "y_mean", "resid_mean", "ale"),
  collapse_m = 30L,
  collapse_by = c("weight", "N"),
  drop_below_n = 0,
  drop_below_weight = 0,
  na.rm = FALSE,
  ...
)
```

Arguments

`object`	Object of class "EffectData".
`sort_by`	By which statistic ("pd", "pred_mean", "y_mean", "resid_mean", "ale") should the results be sorted? The default is "no" (no sorting). Calculated after all other update steps, e.g., after collapsing or dropping rare levels.
`collapse_m`	If a categorical X has more than `collapse_m` levels, rare levels are collapsed into a new level "Other". Standard deviations are collapsed via root of the weighted average variances. The default is 30. Set to `Inf` for no collapsing.
`collapse_by`	How to determine "rare" levels in `collapse_m`? Either "weight" (default) or "N". Only matters in situations with case weights `w`.
`drop_below_n`	Drop bins with N below this value. Applied after collapsing.
`drop_below_weight`	Drop bins with weight below this value. Applied after collapsing.
`na.rm`	Should missing bin centers be dropped? Default is `FALSE`.
`...`	Currently not used.

Value

An object of class "EffectData".

See Also

feature_effects(), average_observed(), average_predicted(), partial_dependence(), ale(), bias(), effect_importance()

Examples

```
fit <- lm(Sepal.Length ~ ., data = iris)
xvars <- colnames(iris)[-1]
feature_effects(fit, v = xvars, data = iris, y = "Sepal.Length", breaks = 5) |>
  update(sort = "pd", collapse_m = 2) |>
  plot()
```

Index