Package 'first'

May 8, 2026

Title Factor Importance Ranking and Selection using Total Indices

Version 2.1

Imports stats, parallel, FNN, twinning

Description
A model-independent factor importance ranking and selection procedure based on total Sobol' indices. Please see Huang and Joseph (2025) <doi:10.1080/00401706.2025.2483531>. This research is supported by U.S. National Science Foundation grants DMS-2310637 and DMREF-1921873.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.1.2

NeedsCompilation no

Author Chaofan Huang [aut, cre],
V. Roshan Joseph [aut]

Maintainer Chaofan Huang <chaofan.huang@gatech.edu>

Repository CRAN

Date/Publication 2025-09-01 08:00:24 UTC

Contents

`first` *Factor Importance Ranking and Selection using Total (Sobol') indices*

Description

`first` implements the **model-independent** factor importance and selection procedure proposed in Huang and Joseph (2025). The importance measure is based on Sobol' indices from global sensitivity analysis. Factor importance computation and selection are performed directly from the noisy data. Parallel computations are available to accelerate the estimation. For categorical data inputs, please convert them to factor type before calling the function.

Usage

```
first(
  X,
  y,
  n.knn = NULL,
  n.mc = nrow(X),
  twin.mc = FALSE,
  rescale = TRUE,
  n.forward = 2,
  importance = "total",
  parl = NULL,
  verbose = FALSE
)
```

Arguments

`X`	a matrix or data frame for the factors / predictors.
`y`	a vector for the responses.
`n.knn`	number of nearest-neighbors for the inner loop conditional variance estimation. `n.knn=2` is recommended for regression, and `n.knn=3` for binary classification.
`n.mc`	number of Monte Carlo samples for the outer loop expectation estimation.
`twin.mc`	a logical indicating whether to use twinning subsamples, otherwise random subsamples are used. It is supported when the reduction ratio is at least 2.
`rescale`	a logical logical indicating whether to standardize the factors / predictors.
`n.forward`	number of times to run the forward selection phase to tradeoff between efficiency and accuracy. `n_forward=2` is recommended. To run the complete forward selection, please set `n_forward` to the number of factors / predictors.
`importance`	the options for the importance of `FIRST`. Default is `total`, which return a numeric vector for total Sobol' effect, and the other option is `shapley`, which return a numeric vector for Shapley Sobol' effect. See Huang and Joseph (2025) for details.
`parl`	number of cores on which to parallelize the computation. If `NULL`, then no parallelization is done.
`verbose`	a logical indicating whether to display intermediate results.

Details

`first` provides factor importance ranking and selection directly from scattered data without any model fitting. It belongs to the class of forward-backward selection with early dropping algorithm (Borboudakis and Tsamardinos, 2019). In forward selection, each time we find the candidate that maximizes the output variance that can be explained. For candidates that cannot improve the variance explained conditional on the selected factors, they are removed from the candidate set. This forward selection step is run `n_forward` times to tradeoff between accuracy and efficiency. `n_forward=2` is recommended in Yu et al. (2020). To run the complete forward selection, please set `n_forward` to the number of factors / predictors. In backward elimination, we again remove one factor at a time, starting with the factor that can improve the explained variance most, till no factor can further improve.

`n.knn=2` nearest-neighbors is recommended for integer/numeric output, and `n.knn=3` is suggested for binary output. For numeric inputs, it is recommended to standardize them via setting the argument `rescale=TRUE`. Categorical inputs are transformed via one-hot encoding for the nearest-neighbor search. To speed up the nearest-neighbor search, k-d tree from the **FNN** package is used. Also, parallel computation is also supported via the **parallel** package.

For large datasets, we support the use of subsamples for further acceleration. Use argument `n.mc` to specify the number of subsamples. Two options are available for finding the subsamples: random and twinning (Vakayil and Joseph, 2022). Twinning is able to find subsamples that better represent the big data, i.e., providing a more accurate estimation, but at a slightly higher computational cost. For more details, please see the **twinning** package.

Value

A numeric vector of the factor importance, with zero indicating that the factor is not important to the prediction of the response.

Author(s)

Chaofan Huang <chaofan.huang@gatech.edu> and V. Roshan Joseph <roshan@gatech.edu>

References

Huang, C., & Joseph, V. R. (2025). Factor Importance Ranking and Selection using Total Indices. Technometrics.

Sobol', I. M. (2001). Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Mathematics and computers in simulation, 55(1-3), 271-280.

Broto, B., Bachoc, F., & Depecker, M. (2020). Variance reduction for estimation of Shapley effects and adaptation to unknown input distribution. SIAM/ASA Journal on Uncertainty Quantification, 8(2), 693-716.

Borboudakis, G., & Tsamardinos, I. (2019). Forward-backward selection with early dropping. The Journal of Machine Learning Research, 20(1), 276-314.

Yu, K., Guo, X., Liu, L., Li, J., Wang, H., Ling, Z., & Wu, X. (2020). Causality-based feature selection: Methods and evaluations. ACM Computing Surveys (CSUR), 53(5), 1-36.

Vakayil, A., & Joseph, V. R. (2022). Data twinning. Statistical Analysis and Data Mining: The ASA Data Science Journal, 15(5), 598-610.

Examples

```
ishigami <- function(x) {
  x <- -pi + 2*pi*x
  y <- sin(x[1]) + 7*sin(x[2])^2 + 0.1*x[3]^4*sin(x[1])
  return (y)
}
set.seed(123)
n <- 1000
p <- 6
X <- matrix(runif(n*p), ncol=p)
y <- apply(X,1,ishigami) + rnorm(n)
imp <- first(X, y, n.knn=2, rescale=FALSE, verbose=TRUE)
print(round(imp,3)) # Only first 3 factors are important
```

`first.rank` *Factor Ranking via Cumulative Variance that can be explained.*

Description

`first.tank` implements FIRST-RANK algorithm in Huang and Joseph (2025). Parallel computation is available to accelerate the estimation. For categorical inputs, please convert them to factor before calling this function. For large datasets, we support the use of subsample to reduce the computational cost.

Usage

```
first.rank(
  X,
  y,
  noise,
  n.knn = NULL,
  n.mc = nrow(X),
  twin.mc = FALSE,
  rescale = TRUE,
  parl = NULL
)
```

Arguments

`X`	a matrix or data frame for the factors / predictors.
`y`	a vector for the responses.
`noise`	a logical indicating whether the responses are noisy.
`n.knn`	number of nearest-neighbors for the inner loop conditional variance estimation. `n.knn=2` is recommended for regression, and `n.knn=3` for binary classification.
`n.mc`	number of Monte Carlo samples for the outer loop expectation estimation.

twin.mc	a logical indicating whether to use twinning subsamples, otherwise random subsamples are used. It is supported when the reduction ratio is at least 2.
rescale	a logical logical indicating whether to standardize the factors / predictors.
parl	number of cores on which to parallelize the computation. If NULL, then no parallelization is done.

Details

first.rank provdes factor ranking directly from scattered data via cumulative variance that can be explained. Please see Huang and Joseph (2025) for a more detailed discussion and comparison.

For integer/numeric output, n.knn=2 nearest-neighbors is recommended for the noisy data (Huang and Joseph, 2025), and n.knn=3 nearest-neighbors is suggested for the clean/noiseless data (Broto et al., 2020). For numeric inputs, it is recommended to standardize them via setting the argument rescale=TRUE. Categorical inputs are transformed via one-hot encoding for the nearest-neighbor search. To speed up the nearest-neighbor search, k-d tree from the **FNN** package is used. Also, parallel computation is also supported via the **parallel** package.

Last, for large datasets, we support the use of subsamples for further acceleration. Use argument n.mc to specify the number of subsamples. Two options are available for finding the subsamples: random and twinning (Vakayil and Joseph, 2022). Twinning is able to find subsamples that better represent the big data, i.e., providing a more accurate estimation, but at a slightly higher computational cost. For more details, please see the **twinning** package.

Value

ranking	Indices of the factors, ordered from most to least importance
explained.variance	Cumulative variance explained by the factors in the ranking order

Author(s)

Chaofan Huang <chaofan.huang@gatech.edu> and V. Roshan Joseph <roshan@gatech.edu>

References

Huang, C., & Joseph, V. R. (2025). Factor Importance Ranking and Selection using Total Indices. Technometrics.

Sobol', I. M. (2001). Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Mathematics and computers in simulation, 55(1-3), 271-280.

Broto, B., Bachoc, F., & Depecker, M. (2020). Variance reduction for estimation of Shapley effects and adaptation to unknown input distribution. SIAM/ASA Journal on Uncertainty Quantification, 8(2), 693-716.

Vakayil, A., & Joseph, V. R. (2022). Data twinning. Statistical Analysis and Data Mining: The ASA Data Science Journal, 15(5), 598-610.

Examples

```
ishigami <- function(x) {
  x <- -pi + 2*pi*x
  y <- sin(x[1]) + 7*sin(x[2])^2 + 0.1*x[3]^4*sin(x[1])
  return (y)
}

set.seed(123)
n <- 10000
p <- 3
X <- matrix(runif(n*p), ncol=p)
y <- apply(X,1,ishigami) + rnorm(n)
res <- first.rank(X, y, noise=TRUE, n.knn=2, rescale=FALSE)
res
```

---

shapleysobol.knn *Estimating Shapley Sobol' Effect from Data*

Description

shapleysobol.knn implements the estimation of the Shapley Sobol' effect directly from scattered data. Parallel computation is available to accelerate the estimation. For categorical inputs, please convert them to factor before calling this function. For large datasets, we support the use of subsample to reduce the computational cost.

Usage

```
shapleysobol.knn(
  X,
  y,
  noise,
  n.knn = NULL,
  n.mc = nrow(X),
  twin.mc = FALSE,
  rescale = TRUE,
  parl = NULL
)
```

Arguments

X	a matrix or data frame for the factors / predictors.
y	a vector for the responses.
noise	a logical indicating whether the responses are noisy.
n.knn	number of nearest-neighbors for the inner loop conditional variance estimation. n.knn=2 is recommended for regression, and n.knn=3 for binary classification.
n.mc	number of Monte Carlo samples for the outer loop expectation estimation.

twin.mc	a logical indicating whether to use twinning subsamples, otherwise random subsamples are used. It is supported when the reduction ratio is at least 2.
rescale	a logical logical indicating whether to standardize the factors / predictors.
parl	number of cores on which to parallelize the computation. If NULL, then no parallelization is done.

Details

shapleysobol.knn provides consistent estimation of the Shapley Sobol' Effect (Owen, 2014; Song et al., 2016) from scattered data. When the output is clean/noiseless (noise=FALSE), shapleysobol.knn implements the Nearest-Neighbor estimator proposed in Broto et al. (2020). When the output is noisy (noise=TRUE), shapleysobol.knn implements the Noise-Adjusted Nearest-Neighbor (NANNE) estimator (Huang and Joseph, 2025). NANNE estimator can correct the estimation bias of the nearest-neighbor estimator caused by the random noise. Please see Huang and Joseph (2025) for a more detailed discussion and comparison.

For integer/numeric output, n.knn=2 nearest-neighbors is recommended for the noisy data (Huang and Joseph, 2025), and n.knn=3 nearest-neighbors is suggested for the clean/noiseless data (Broto et al., 2020). For numeric inputs, it is recommended to standardize them via setting the argument rescale=TRUE. Categorical inputs are transformed via one-hot encoding for the nearest-neighbor search. To speed up the nearest-neighbor search, k-d tree from the **FNN** package is used. Also, parallel computation is also supported via the **parallel** package.

Last, for large datasets, we support the use of subsamples for further acceleration. Use argument n.mc to specify the number of subsamples. Two options are available for finding the subsamples: random and twinning (Vakayil and Joseph, 2022). Twinning is able to find subsamples that better represent the big data, i.e., providing a more accurate estimation, but at a slightly higher computational cost. For more details, please see the **twinning** package.

Value

A numeric vector of the Shapley Sobol' effect estimation.

Author(s)

Chaofan Huang <chaofan.huang@gatech.edu> and V. Roshan Joseph <roshan@gatech.edu>

References

Huang, C., & Joseph, V. R. (2025). Factor Importance Ranking and Selection using Total Indices. Technometrics.

Owen, A. B. (2014), "Sobol'indices and Shapley value," SIAM/ASA Journal on Uncertainty Quantification, 2, 245–251.

Song, E., Nelson, B. L., & Staum, J. (2016), "Shapley effects for global sensitivity analysis: Theory and computation," SIAM/ASA Journal on Uncertainty Quantification, 4, 1060-1083.

Broto, B., Bachoc, F., & Depecker, M. (2020). Variance reduction for estimation of Shapley effects and adaptation to unknown input distribution. SIAM/ASA Journal on Uncertainty Quantification, 8(2), 693-716.

Vakayil, A., & Joseph, V. R. (2022). Data twinning. Statistical Analysis and Data Mining: The ASA Data Science Journal, 15(5), 598-610.

Examples

```
ishigami <- function(x) {
  x <- -pi + 2*pi*x
  y <- sin(x[1]) + 7*sin(x[2])^2 + 0.1*x[3]^4*sin(x[1])
  return (y)
}

set.seed(123)
n <- 10000
p <- 3
X <- matrix(runif(n*p), ncol=p)
y <- apply(X,1,ishigami) + rnorm(n)
ssi <- shapleysobol.knn(X, y, noise=TRUE, n.knn=2, rescale=FALSE)
print(round(ssi,3))
```

`totalsobol.knn` *Estimating Total Sobol' Indices from Data*

Description

`totalsobol.knn` implements the estimation of the total Sobol' indices directly from scattered data. Parallel computation is available to accelerate the estimation. For categorical inputs, please convert them to factor before calling this function. For large datasets, we support the use of subsample to reduce the computational cost.

Usage

```
totalsobol.knn(
  X,
  y,
  noise,
  n.knn = NULL,
  n.mc = nrow(X),
  twin.mc = FALSE,
  rescale = TRUE,
  parl = NULL
)
```

Arguments

`X`	a matrix or data frame for the factors / predictors.
`y`	a vector for the responses.
`noise`	a logical indicating whether the responses are noisy.
`n.knn`	number of nearest-neighbors for the inner loop conditional variance estimation. `n.knn=2` is recommended for regression, and `n.knn=3` for binary classification.
`n.mc`	number of Monte Carlo samples for the outer loop expectation estimation.

twin.mc	a logical indicating whether to use twinning subsamples, otherwise random subsamples are used. It is supported when the reduction ratio is at least 2.
rescale	a logical logical indicating whether to standardize the factors / predictors.
parl	number of cores on which to parallelize the computation. If NULL, then no parallelization is done.

Details

totalsobol.knn provides consistent estimation of the total Sobol' indices (Sobol, 1993) from scattered data. When the output is clean/noiseless (noise=FALSE), totalsobol.knn implements the Nearest-Neighbor estimator proposed in Broto et al. (2020). See shapleysobol_knn from the **sensitivity** package for another implementation of the nearest-neighbor estimator. When the output is noisy (noise=TRUE), totalsobol.knn implements the Noise-Adjusted Nearest-Neighbor (NANNE) estimator (Huang and Joseph, 2025). NANNE estimator can correct the estimation bias of the nearest-neighbor estimator caused by the random noise. Please see Huang and Joseph (2025) for a more detailed discussion and comparison.

For integer/numeric output, n.knn=2 nearest-neighbors is recommended for the noisy data (Huang and Joseph, 2025), and n.knn=3 nearest-neighbors is suggested for the clean/noiseless data (Broto et al., 2020). For numeric inputs, it is recommended to standardize them via setting the argument rescale=TRUE. Categorical inputs are transformed via one-hot encoding for the nearest-neighbor search. To speed up the nearest-neighbor search, k-d tree from the **FNN** package is used. Also, parallel computation is also supported via the **parallel** package.

Last, for large datasets, we support the use of subsamples for further acceleration. Use argument n.mc to specify the number of subsamples. Two options are available for finding the subsamples: random and twinning (Vakayil and Joseph, 2022). Twinning is able to find subsamples that better represent the big data, i.e., providing a more accurate estimation, but at a slightly higher computational cost. For more details, please see the **twinning** package.

Value

A numeric vector of the total Sobol' indices estimation.

Author(s)

Chaofan Huang <chaofan.huang@gatech.edu> and V. Roshan Joseph <roshan@gatech.edu>

References

Huang, C., & Joseph, V. R. (2025). Factor Importance Ranking and Selection using Total Indices. Technometrics.

Sobol', I. M. (2001). Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Mathematics and computers in simulation, 55(1-3), 271-280.

Broto, B., Bachoc, F., & Depecker, M. (2020). Variance reduction for estimation of Shapley effects and adaptation to unknown input distribution. SIAM/ASA Journal on Uncertainty Quantification, 8(2), 693-716.

Vakayil, A., & Joseph, V. R. (2022). Data twinning. Statistical Analysis and Data Mining: The ASA Data Science Journal, 15(5), 598-610.

Examples

```
ishigami <- function(x) {
  x <- -pi + 2*pi*x
  y <- sin(x[1]) + 7*sin(x[2])^2 + 0.1*x[3]^4*sin(x[1])
  return (y)
}

set.seed(123)
n <- 10000
p <- 3
X <- matrix(runif(n*p), ncol=p)
y <- apply(X,1,ishigami) + rnorm(n)
tsi <- totalsobol.knn(X, y, noise=TRUE, n.knn=2, rescale=FALSE)
print(round(tsi,3)) # Analytical Total Sobol' Indices: 0.558, 0.442, 0.244
```

Index