Package ‘spm’

September 6, 2021

Title Spatial Predictive Modeling

Version 1.2.1

Date 2021-09-01

Description Introduction to some novel accurate hybrid methods of geostatistical and machine learning methods for spatial predictive modelling. It contains two commonly used geostatistical methods, two machine learning methods, four hybrid methods and two averaging methods. For each method, two functions are provided. One function is for assessing the predictive errors and accuracy of the method based on cross-validation. The other one is for generating spatial predictions using the method. For details please see: Li, J., Potter, A., Huang, Z., Daniell, J. J. and Heap, A. (2010) <https:www.ga.gov.au/metadata-gateway/metadata/record/gcat_71407>
Li, J., Heap, A. D., Potter, A., Huang, Z. and Daniell, J. (2011) <doi:10.1016/j.csr.2011.05.015>
Li, J., Heap, A. D., Potter, A. and Daniell, J. (2011) <doi:10.1016/j.envsoft.2011.07.004>
Li, J., Potter, A., Huang, Z. and Heap, A. (2012) <https:www.ga.gov.au/metadata-gateway/metadata/record/74030>.

Depends R (>= 2.10)

Imports gstat, sp, randomForest, psy, gbm, biomod2, stats, ranger

License GPL (>= 2)

LazyData true

RoxygenNote 7.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Jin Li [aut, cre]

Maintainer Jin Li <`jinli68@gmail.com`>

Repository CRAN

Date/Publication 2021-09-06 09:20:05 UTC

R topics documented:

avi *Averaged variable importance based on random forest*

Description

This function is to derive an averaged variable importance based on random forest

Usage

```
avi(
  trainx,
  trainy,
 mtry = if (!is.null(trainy) && !is.factor(trainy)) max(floor(ncol(trainx)/3), 1) else
    floor(sqrt(ncol(trainx))),
  ntree = 500,
  importance = TRUE,
  maxk = c(4),
  nsim = 100,
  corr.threshold = 0.5,
  ...
)
```

Arguments

`trainx`	a dataframe or matrix contains columns of predictor variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`ntree`	number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. By default, 500 is used.
`importance`	imprtance of predictive variables.
`maxk`	maxk split value. By default, 4 is used.
`nsim`	iteration number. By default, 100 is used.
`corr.threshold`	correlation threshold and the defaults value is 0.5.
`...`	other arguments passed on to randomForest.

Value

A list with the following components: averaged variable importance (avi), column number of importance variable in trainx arranged from the most important to the least important (impvar), names of importance variable arranged from the most important to the least important (impvar2)

Author(s)

Jin Li

References

Smith, S.J., Ellis, N., Pitcher, C.R., 2011. Conditional variable importance in R package extendedForest.

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(petrel)
set.seed(1234)
avi1 <- avi(petrel[, c(1,2, 6:9)], petrel[, 5], nsim = 10)
avi1

avi1 <- avi(petrel[, c(1), drop = FALSE], petrel[, 5], nsim = 10)
avi1

## End(Not run)
```

`cran-comments` *Note on notes*

Description

This is my first submission.

R CMD check results 0 errors | 0 warnings | 0 notes

Author(s)

Jin Li

`gbmcv` *Cross validation, n-fold for generalized boosted regression modeling (gbm)*

Description

This function is a cross validation function for generalized boosted regression modeling.

Usage

```
gbmcv(
  trainx,
  trainy,
  var.monotone = rep(0, ncol(trainx)),
  family = "gaussian",
  n.trees = 3000,
  learning.rate = 0.001,
  interaction.depth = 2,
  bag.fraction = 0.5,
  train.fraction = 1,
  n.minobsinnode = 10,
  cv.fold = 10,
  weights = rep(1, nrow(trainx)),
  keep.data = FALSE,
  verbose = TRUE,
  n.cores = 6,
  predacc = "VEcv",
  ...
)
```

Arguments

`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome. By default, a vector of 0 is used.
`family`	either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed. See gbm for details. By default, "gaussian" is used.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, 3000 is used.
`learning.rate`	a shrinkage parameter applied to each tree in the expansion. Also known as step-size reduction.
`interaction.depth`	the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. By default, 2 is used.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion. By default, 0.5 is used.
`train.fraction`	The first train.fraction * nrows(data) observations are used to fit the gbm and the remainder are used for computing out-of-sample estimates of the loss function.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight. By default, 10 is used.

`cv.fold`	integer; number of folds in the cross-validation. it is also the number of cross-validation folds to perform within gbm. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`weights`	an optional vector of weights to be used in the fitting process. Must be positive but do not need to be normalized. If keep.data = FALSE in the initial call to gbm then it is the user's responsibility to resupply the weights to gbm.more. By default, a vector of 1 is used.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object. Keeping the data and index makes subsequent calls to gbm.more faster at the cost of storing an extra copy of the dataset. By default, 'FALSE' is used.
`verbose`	If TRUE, gbm will print out progress and performance indicators. By default, 'TRUE' is used.
`n.cores`	The number of CPU cores to use. See gbm for details. By default, 6 is used.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to gbm.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv for categorical data: correct classification rate (ccr.cv) and kappa (kappa.cv)

Note

This function is largely based on rf.cv (see Li et al. 2013), rfcv in randomForest and gbm.

Author(s)

Jin Li

References

Li, J., J. Siwabessy, M. Tran, Z. Huang, and A. Heap. 2013. Predicting Seabed Hardness Using Random Forest in R. Pages 299-329 in Y. Zhao and Y. Cen, editors. Data Mining Applications with R. Elsevier.

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Greg Ridgeway with contributions from others (2015). gbm: Generalized Boosted Regression Models. R package version 2.1.1. https://CRAN.R-project.org/package=gbm

Examples

```
## Not run:
data(sponge)

gbmcv1 <- gbmcv(sponge[, -c(3)], sponge[, 3], cv.fold = 10,
family = "poisson", n.cores=2, predacc = "ALL")
gbmcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
gbmcv1 <- gbmcv(sponge[, -c(3)], sponge[, 3], cv.fold = 10,
family = "poisson",  n.cores=2, predacc = "VEcv")
VEcv [i] <- gbmcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for gbm", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

## End(Not run)
```

`gbmidwcv` *Cross validation, n-fold for the hybrid method of generalized boosted regression modeling and inverse distance weighting (gbmidw)*

Description

This function is a cross validation function for the hybrid method of generalized boosted regression modeling and inverse distance weighting.

Usage

```
gbmidwcv(
  longlat,
  trainx,
  trainy,
  var.monotone = rep(0, ncol(trainx)),
  family = "gaussian",
  n.trees = 3000,
  learning.rate = 0.001,
  interaction.depth = 2,
  bag.fraction = 0.5,
  train.fraction = 1,
  n.minobsinnode = 10,
  cv.fold = 10,
  weights = rep(1, nrow(trainx)),
  keep.data = FALSE,
```

```
  verbose = TRUE,
  idp = 2,
  nmax = 12,
  predacc = "VEcv",
  n.cores = 6,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome. By default, a vector of 0 is used.
`family`	either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed. See gbm for details. By default, "gaussian" is used.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, 3000 is used.
`learning.rate`	a shrinkage parameter applied to each tree in the expansion. Also known as step-size reduction.
`interaction.depth`	the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. By default, 2 is used.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion. By default, 0.5 is used.
`train.fraction`	The first train.fraction * nrows(data) observations are used to fit the gbm and the remainder are used for computing out-of-sample estimates of the loss function.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight. By default, 10 is used.
`cv.fold`	integer; number of folds in the cross-validation. it is also the number of cross-validation folds to perform within gbm. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`weights`	an optional vector of weights to be used in the fitting process. Must be positive but do not need to be normalized. If keep.data=FALSE in the initial call to gbm then it is the user's responsibility to resupply the weights to gbm.more. By default, a vector of 1 is used.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object. Keeping the data and index makes subsequent calls to gbm.more faster at the cost of storing an extra copy of the dataset. By default, 'FALSE' is used.

`verbose`	If TRUE, gbm will print out progress and performance indicators. By default, 'TRUE' is used.
`idp`	numeric; specify the inverse distance weighting power.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`n.cores`	The number of CPU cores to use. See gbm for details. By default, 6 is used.
`...`	other arguments passed on to gbm.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv for categorical data: correct classification rate (ccr.cv) and kappa (kappa.cv)

Note

this function is largely based on rf.cv (see Li et al. 2013), rfcvrandomForest and gbm

Author(s)

Jin Li

References

Li, J., J. Siwabessy, M. Tran, Z. Huang, and A. Heap. 2013. Predicting Seabed Hardness Using Random Forest in R. Pages 299-329 in Y. Zhao and Y. Cen, editors. Data Mining Applications with R. Elsevier.

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Greg Ridgeway with contributions from others (2015). gbm: Generalized Boosted Regression Models. R package version 2.1.1. https://CRAN.R-project.org/package=gbm

Examples

```
## Not run:
data(sponge)

gbmidwcv1 <- gbmidwcv(sponge[, c(1,2)], sponge[, -c(3)], sponge[, 3],
cv.fold = 10, family = "poisson", n.cores=2, predacc = "ALL")
gbmidwcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
```

```
gbmidwcv1 <- gbmidwcv(sponge[, c(1,2)], sponge[, -c(3)], sponge[, 3],
cv.fold = 10, family = "poisson", n.cores=2, predacc = "VEcv")
VEcv [i] <- gbmidwcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for gbmidw", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

## End(Not run)
```

`gbmidwpred` *Generate spatial predictions using the hybrid method of generalized boosted regression modeling and inverse distance weighting (gbmidw)*

Description

This function is to make spatial predictions using the hybrid method of generalized boosted regression modeling and inverse distance weighting.

Usage

```
gbmidwpred(
  longlat,
  trainx,
  trainy,
  longlatpredx,
  predx,
  var.monotone = rep(0, ncol(trainx)),
  family = "gaussian",
  n.trees = 3000,
  learning.rate = 0.001,
  interaction.depth = 2,
  bag.fraction = 0.5,
  train.fraction = 1,
  n.minobsinnode = 10,
  cv.fold = 10,
  weights = rep(1, nrow(trainx)),
  keep.data = FALSE,
  verbose = TRUE,
  idp = 2,
  nmax = 12,
  n.cores = 6,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome. By default, a vector of 0 is used.
`family`	either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed. See gbm for details. By default, "gaussian" is used.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, 3000 is used.
`learning.rate`	a shrinkage parameter applied to each tree in the expansion. Also known as step-size reduction.
`interaction.depth`	the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. By default, 2 is used.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion. By default, 0.5 is used.
`train.fraction`	The first train.fraction * nrows(data) observations are used to fit the gbm and the remainder are used for computing out-of-sample estimates of the loss function.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight. By default, 10 is used.
`cv.fold`	integer; number of folds in the cross-validation. it is also the number of cross-validation folds to perform within gbm. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`weights`	an optional vector of weights to be used in the fitting process. Must be positive but do not need to be normalized. If keep.data = FALSE in the initial call to gbm then it is the user's responsibility to resupply the weights to gbm.more. By default, a vector of 1 is used.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object. Keeping the data and index makes subsequent calls to gbm.more faster at the cost of storing an extra copy of the dataset. By default, 'FALSE' is used.
`verbose`	If TRUE, gbm will print out progress and performance indicators. By default, 'TRUE' is used.
`idp`	numeric; specify the inverse distance weighting power.

`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`n.cores`	The number of CPU cores to use. See gbm for details. By default, 6 is used.
`...`	other arguments passed on to gbm.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse and vecv; or vecv for categorical data: correct classification rate (ccr.cv) and kappa (kappa.cv)

Note

This function is largely based on gbm.

Author(s)

Jin Li

References

Li, J., J. Siwabessy, M. Tran, Z. Huang, and A. Heap. 2013. Predicting Seabed Hardness Using Random Forest in R. Pages 299-329 in Y. Zhao and Y. Cen, editors. Data Mining Applications with R. Elsevier.

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Greg Ridgeway with contributions from others (2015). gbm: Generalized Boosted Regression Models. R package version 2.1.1. https://CRAN.R-project.org/package=gbm

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
gbmidwpred1 <- gbmidwpred(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 3],
  petrel.grid[, c(1,2)], petrel.grid, family = "gaussian", n.cores=6,
  nmax = 12)
names(gbmidwpred1)

## End(Not run)
```

`gbmokcv` *Cross validation, n-fold for the hybrid method of generalized boosted regression modeling and ordinary kriging (gbmok)*

Description

This function is a cross validation function for the hybrid method of generalized boosted regression modeling and ordinary kriging.

Usage

```
gbmokcv(
  longlat,
  trainx,
  trainy,
  var.monotone = rep(0, ncol(trainx)),
  family = "gaussian",
  n.trees = 3000,
  learning.rate = 0.001,
  interaction.depth = 2,
  bag.fraction = 0.5,
  train.fraction = 1,
  n.minobsinnode = 10,
  cv.fold = 10,
  weights = rep(1, nrow(trainx)),
  keep.data = FALSE,
  verbose = TRUE,
  nmax = 12,
  vgm.args = ("Sph"),
  block = 0,
  predacc = "VEcv",
  n.cores = 6,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome. By default, a vector of 0 is used.

`family`	either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed. See gbm for details. By default, "gaussian" is used.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, 3000 is used.
`learning.rate`	a shrinkage parameter applied to each tree in the expansion. Also known as step-size reduction.
`interaction.depth`	the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. By default, 2 is used.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion. By default, 0.5 is used.
`train.fraction`	The first train.fraction * nrows(data) observations are used to fit the gbm and the remainder are used for computing out-of-sample estimates of the loss function.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight. By default, 10 is used.
`cv.fold`	integer; number of folds in the cross-validation. it is also the number of cross-validation folds to perform within gbm. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`weights`	an optional vector of weights to be used in the fitting process. Must be positive but do not need to be normalized. If keep.data = FALSE in the initial call to gbm then it is the user's responsibility to resupply the weights to gbm.more. By default, a vector of 1 is used.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object. Keeping the data and index makes subsequent calls to gbm.more faster at the cost of storing an extra copy of the dataset. By default, 'FALSE' is used.
`verbose`	If TRUE, gbm will print out progress and performance indicators. By default, 'TRUE' is used.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`n.cores`	The number of CPU cores to use. See gbm for details. By default, 6 is used.
`...`	other arguments passed on to gbm.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and `e1`; or vecv for categorical data: correct classification rate (ccr.cv) and kappa (kappa.cv)

Note

This function is largely based on rf.cv (see Li et al. 2013), rfcv in randomForest and gbm. When 'A zero or negative range was fitted to variogram' occurs, to allow gstat running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Li, J., J. Siwabessy, M. Tran, Z. Huang, and A. Heap. 2013. Predicting Seabed Hardness Using Random Forest in R. Pages 299-329 in Y. Zhao and Y. Cen, editors. Data Mining Applications with R. Elsevier.

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Greg Ridgeway with contributions from others (2015). gbm: Generalized Boosted Regression Models. R package version 2.1.1. https://CRAN.R-project.org/package=gbm

Examples

```
## Not run:
data(sponge)

gbmokcv1 <- gbmokcv(sponge[, c(1,2)], sponge[,-c(3)], sponge[, 3],
cv.fold = 10, family = "poisson", n.cores=2, predacc = "ALL")
gbmokcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
gbmokcv1 <- gbmokcv(sponge[, c(1,2)], sponge[, -c(3)], sponge[, 3],
cv.fold = 10, family = "poisson", n.cores=2, predacc = "VEcv")
VEcv [i] <- gbmokcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for gbmok", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

## End(Not run)
```

`gbmokgbmidwcv` *Cross validation, n-fold for the average of the hybrid method of generalized boosted regression modeling and ordinary kriging and the hybrid method of generalized boosted regression modeling and inverse distance weighting (gbmokgbmidw)*

Description

This function is a cross validation function for the average of the hybrid method of generalized boosted regression modeling and ordinary kriging and the hybrid method of generalized boosted regression modeling and inverse distance weighting.

Usage

```
gbmokgbmidwcv(
  longlat,
  trainx,
  trainy,
  var.monotone = rep(0, ncol(trainx)),
  family = "gaussian",
  n.trees = 3000,
  learning.rate = 0.001,
  interaction.depth = 2,
  bag.fraction = 0.5,
  train.fraction = 1,
  n.minobsinnode = 10,
  cv.fold = 10,
  weights = rep(1, nrow(trainx)),
  keep.data = FALSE,
  verbose = TRUE,
  idp = 2,
  nmaxidw = 12,
  nmaxok = 12,
  vgm.args = ("Sph"),
  block = 0,
  predacc = "VEcv",
  n.cores = 6,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.

`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome. By default, a vector of 0 is used.
`family`	either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed. See gbm for details. By default, "gaussian" is used.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, 3000 is used.
`learning.rate`	a shrinkage parameter applied to each tree in the expansion. Also known as step-size reduction.
`interaction.depth`	the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. By default, 2 is used.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion. By default, 0.5 is used.
`train.fraction`	The first train.fraction * nrows(data) observations are used to fit the gbm and the remainder are used for computing out-of-sample estimates of the loss function.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight. By default, 10 is used.
`cv.fold`	integer; number of folds in the cross-validation. it is also the number of cross-validation folds to perform within gbm. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`weights`	an optional vector of weights to be used in the fitting process. Must be positive but do not need to be normalized. If keep.data = FALSE in the initial call to gbm then it is the user's responsibility to resupply the weights to gbm.more. By default, a vector of 1 is used.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object. Keeping the data and index makes subsequent calls to gbm.more faster at the cost of storing an extra copy of the dataset. By default, 'FALSE' is used.
`verbose`	If TRUE, gbm will print out progress and performance indicators. By default, 'TRUE' is used.
`idp`	numeric; specify the inverse distance weighting power.
`nmaxidw`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for IDW.
`nmaxok`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for OK.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgmgstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.

`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`n.cores`	The number of CPU cores to use. See gbm for details. By default, 6 is used.
`...`	other arguments passed on to gbm.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv for categorical data: correct classification rate (ccr.cv) and kappa (kappa.cv)

Note

This function is largely based on rf.cv (see Li et al. 2013), rfcv in randomForest and gbm. When 'A zero or negative range was fitted to variogram' occurs, to allow gstat running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Li, J., J. Siwabessy, M. Tran, Z. Huang, and A. Heap. 2013. Predicting Seabed Hardness Using Random Forest in R. Pages 299-329 in Y. Zhao and Y. Cen, editors. Data Mining Applications with R. Elsevier.

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Greg Ridgeway with contributions from others (2015). gbm: Generalized Boosted Regression Models. R package version 2.1.1. https://CRAN.R-project.org/package=gbm

Examples

```
## Not run:
data(sponge)

gbmokgbmidw1 <- gbmokgbmidwcv(sponge[, c(1,2)], sponge[, -c(3)], sponge[, 3],
cv.fold = 10, family = "poisson", n.cores=2, predacc = "ALL")
gbmokgbmidw1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
gbmokgbmidw1 <- gbmokgbmidwcv(sponge[, c(1,2)], sponge[, -c(3)], sponge[, 3],
cv.fold = 10, family = "poisson", n.cores=2, predacc = "VEcv")
VEcv [i] <- gbmokgbmidw1
}
```

```
plot(VEcv ~ c(1:n), xlab = "Iteration for gbmokgbmidw", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

## End(Not run)
```

`gbmokgbmidwpred` *Generate spatial predictions using the average of the hybrid method of generalized boosted regression modeling and ordinary kriging and the hybrid method of generalized boosted regression modeling and inverse distance weighting (gbmokgbmidw)*

Description

This function is to make spatial predictions using the average of the hybrid method of generalized boosted regression modeling and ordinary kriging and the hybrid method of generalized boosted regression modeling and inverse distance weighting.

Usage

```
gbmokgbmidwpred(
  longlat,
  trainx,
  trainy,
  longlatpredx,
  predx,
  var.monotone = rep(0, ncol(trainx)),
  family = "gaussian",
  n.trees = 3000,
  learning.rate = 0.001,
  interaction.depth = 2,
  bag.fraction = 0.5,
  train.fraction = 1,
  n.minobsinnode = 10,
  cv.fold = 0,
  weights = rep(1, nrow(trainx)),
  keep.data = FALSE,
  verbose = TRUE,
  idp = 2,
  nmaxidw = 12,
  nmaxok = 12,
  vgm.args = ("Sph"),
  block = 0,
  n.cores = 6,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome. By default, a vector of 0 is used.
`family`	either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed. See gbm for details. By default, "gaussian" is used.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, 3000 is used.
`learning.rate`	a shrinkage parameter applied to each tree in the expansion. Also known as step-size reduction.
`interaction.depth`	the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. By default, 2 is used.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion. By default, 0.5 is used.
`train.fraction`	The first train.fraction * nrows(data) observations are used to fit the gbm and the remainder are used for computing out-of-sample estimates of the loss function.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight. By default, 10 is used.
`cv.fold`	integer; number of cross-validation folds to perform within gbm.
`weights`	an optional vector of weights to be used in the fitting process. Must be positive but do not need to be normalized. If keep.data = FALSE in the initial call to gbm then it is the user's responsibility to resupply the weights to gbm.more. By default, a vector of 1 is used.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object. Keeping the data and index makes subsequent calls to gbm.more faster at the cost of storing an extra copy of the dataset. By default, 'FALSE' is used.
`verbose`	If TRUE, gbm will print out progress and performance indicators. By default, 'TRUE' is used.
`idp`	numeric; specify the inverse distance weighting power.
`nmaxidw`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for IDW.

`nmaxok`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for OK.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`n.cores`	The number of CPU cores to use. See gbm for details. By default, 6 is used.
`...`	other arguments passed on to gbm.

Value

A dataframe of longitude, latitude, predictions and variances. The variances are the same as the variances of gbmokpred.

Note

This function is largely based on gbm. When 'A zero or negative range was fitted to variogram' occurs, to allow OK running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Greg Ridgeway with contributions from others (2015). gbm: Generalized Boosted Regression Models. R package version 2.1.1. https://CRAN.R-project.org/package=gbm

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
gbmokgbmidwpred1 <- gbmokgbmidwpred(petrel[, c(1,2)], petrel[, c(1,2, 6:9)],
petrel[, 3], petrel.grid[, c(1,2)], petrel.grid, family = "gaussian",
n.cores=6, nmaxidw = 12, nmaxok = 12, vgm.args = ("Sph"))
names(gbmokgbmidwpred1)

## End(Not run)
```

---

gbmokpred *Generate spatial predictions using the hybrid method of generalized boosted regression modeling and ordinary kriging (gbmok)*

Description

This function is to make spatial predictions using the hybrid method of generalized boosted regression modeling and ordinary kriging.

Usage

```
gbmokpred(
  longlat,
  trainx,
  trainy,
  longlatpredx,
  predx,
  var.monotone = rep(0, ncol(trainx)),
  family = "gaussian",
  n.trees = 3000,
  learning.rate = 0.001,
  interaction.depth = 2,
  bag.fraction = 0.5,
  train.fraction = 1,
  n.minobsinnode = 10,
  cv.fold = 10,
  weights = rep(1, nrow(trainx)),
  keep.data = FALSE,
  verbose = TRUE,
  nmax = 12,
  vgm.args = ("Sph"),
  block = 0,
  n.cores = 6,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.

`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome. By default, a vector of 0 is used.
`family`	either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed. See gbm for details. By default, "gaussian" is used.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, 3000 is used.
`learning.rate`	a shrinkage parameter applied to each tree in the expansion. Also known as step-size reduction.
`interaction.depth`	the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. By default, 2 is used.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion. By default, 0.5 is used.
`train.fraction`	The first train.fraction * nrows(data) observations are used to fit the gbm and the remainder are used for computing out-of-sample estimates of the loss function.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight. By default, 10 is used.
`cv.fold`	integer; number of cross-validation folds to perform within gbm. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`weights`	an optional vector of weights to be used in the fitting process. Must be positive but do not need to be normalized. If keep.data = FALSE in the initial call to gbm then it is the user's responsibility to resupply the weights to gbm.more. By default, a vector of 1 is used.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object. Keeping the data and index makes subsequent calls to gbm.more faster at the cost of storing an extra copy of the dataset. By default, 'FALSE' is used.
`verbose`	If TRUE, gbm will print out progress and performance indicators. By default, 'TRUE' is used.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`n.cores`	The number of CPU cores to use. See gbm for details. By default, 6 is used.
`...`	other arguments passed on to gbm.

Value

A dataframe of longitude, latitude, predictions and variances. The variances are produced by OK based on the residuals of gbm.

Note

This function is largely based on gbm. When 'A zero or negative range was fitted to variogram' occurs, to allow OK running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Greg Ridgeway with contributions from others (2015). gbm: Generalized Boosted Regression Models. R package version 2.1.1. https://CRAN.R-project.org/package=gbm

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
gbmokpred1 <- gbmokpred(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 3],
  petrel.grid[, c(1,2)], petrel.grid, family = "gaussian", n.cores=6,
  nmax = 12, vgm.args = ("Sph"))
names(gbmokpred1)

## End(Not run)
```

---

gbmpred *Generate spatial predictions using generalized boosted regression modeling ('gbm')*

Description

This function is to make spatial predictions using generalized boosted regression modeling.

Usage

```
gbmpred(
  trainx,
  trainy,
  longlatpredx,
  predx,
  var.monotone = rep(0, ncol(trainx)),
  family = "gaussian",
  n.trees = 3000,
  learning.rate = 0.001,
  interaction.depth = 2,
```

```
  bag.fraction = 0.5,
  train.fraction = 1,
  n.minobsinnode = 10,
  cv.fold = 10,
  weights = rep(1, nrow(trainx)),
  keep.data = FALSE,
  verbose = TRUE,
  n.cores = 6,
  ...
)
```

Arguments

`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome. By default, a vector of 0 is used.
`family`	either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed. See ‘gbm‘ for details. By default, "gaussian" is used.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, 3000 is used.
`learning.rate`	a shrinkage parameter applied to each tree in the expansion. Also known as step-size reduction.
`interaction.depth`	the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. By default, 2 is used.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion. By default, 0.5 is used.
`train.fraction`	The first ‘train.fraction * nrows(data)‘ observations are used to fit the ‘gbm‘ and the remainder are used for computing out-of-sample estimates of the loss function.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight. By default, 10 is used.
`cv.fold`	integer; number of cross-validation folds to perform within ‘gbm‘. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`weights`	an optional vector of weights to be used in the fitting process. Must be positive but do not need to be normalized. If keep.data = FALSE in the initial call to

	‘gbm‘ then it is the user’s responsibility to resupply the weights to ‘gbm.more‘. By default, a vector of 1 is used.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object. Keeping the data and index makes subsequent calls to ‘gbm.more‘ faster at the cost of storing an extra copy of the dataset. By default, ’FALSE’ is used.
`verbose`	If TRUE, ‘gbm‘ will print out progress and performance indicators. By default, ’TRUE’ is used.
`n.cores`	The number of CPU cores to use. See ‘gbm‘ for details. By default, 6 is used.
`...`	other arguments passed on to ‘gbm‘.

Value

A dataframe of longitude, latitude and predictions.

Note

This function is largely based on ‘gbm‘.

Author(s)

Jin Li

References

Greg Ridgeway with contributions from others (2015). gbm: Generalized Boosted Regression Models. R package version 2.1.1. https://CRAN.R-project.org/package=gbm

Examples

```
## Not run:
data(sponge)
data(sponge.grid)
gbmpred1 <- gbmpred(sponge[, -c(3)], sponge[, 3], sponge.grid[, c(1:2)],
sponge.grid, family = "poisson", n.cores=2)
names(gbmpred1)

## End(Not run)
```

`hard` *A dataset of seabed hardness in the eastern Joseph Bonaparte Golf, northern Australia marine margin*

Description

This dataset contains 137 samples of 17 variables including area surveyed (Area), easting, northing, prock, bathymetry (bathy), backscatter (bs), local Moran I (bathy.moran), plannar curvature (planar.curv), profile curvature (profile.curv), topographic relief (relief), slope (slope), surface area (surface), topographic position index (tpi), homogeneity of backscatter (homogeneity), local Moran I of backscatter (bs.moran), variance of backscatter (variance) and seabed hardness (hardness).

Usage

```
data("hard")
```

Format

A data frame with 137 observations on the following 17 variables.

`Area` a catergorical vector, no unit

`easting` a numeric vector, m

`northing` a numeric vector, m

`prock` a numeric vector, no unit

`bathy` a numeric vector, meter

`bs` a numeric vector, dB

`bathy.moran` a numeric vector, no unit

`planar.curv` a numeric vector, no unit

`profile.curv` a numeric vector, no unit

`relief` a numeric vector, meter

`slope` a numeric vector, no unit

`surface` a numeric vector, no unit

`tpi` a numeric vector, no unit

`homogeneity` a numeric vector, no unit

`bs.moran` a numeric vector, no unit

`variance` a numeric vector, dB^2

`hardness` a catergorical vector, no unit

Details

For details, please see the source. This dataset was modified by removing 3 samples with missing values from Appendix AA of the book chapter listed in the source.

Source

Li, J., J. Siwabessy, M. Tran, Z. Huang, and A. Heap. 2013. Predicting Seabed Hardness Using Random Forest in R. Pages 299-329 in Y. Zhao and Y. Cen, editors. Data Mining Applications with R. Elsevier.

`idwcv` *Cross validation, n-fold for inverse distance weighting (IDW)*

Description

This function is a cross validation function for inverse distance weighting.

Usage

```
idwcv(longlat, trainy, cv.fold = 10, nmax = 12, idp = 2, predacc = "VEcv", ...)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples.
`trainy`	a vector of response, must have length equal to the number of rows in longlat.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`nmax`	for a local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`idp`	numeric; specify the inverse distance weighting power.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to gstat.

Value

A list with the following components: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv only.

Note

This function is largely based on rfcv in randomForest and some functions in library(gstat).

Author(s)

Jin Li

References

Li, J., 2013. Predictive Modelling Using Random Forest and Its Hybrid Methods with Geostatistical Techniques in Marine Environmental Geosciences, In: Christen, P., Kennedy, P., Liu, L., Ong, K.-L., Stranieri, A., Zhao, Y. (Eds.), The proceedings of the Eleventh Australasian Data Mining Conference (AusDM 2013), Canberra, Australia, 13-15 November 2013. Conferences in Research and Practice in Information Technology, Vol. 146.

A. Liaw and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Pebesma, E.J., 2004. Multivariable geostatistics in S: the gstat package. Computers & Geosciences, 30: 683-691.

Examples

```
## Not run:
library(sp)
data(swmud)
data(petrel)

idwcv1 <- idwcv(swmud[, c(1,2)], swmud[, 3], nmax = 12, idp = 2)
idwcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
idwcv1 <- idwcv(petrel[, c(1,2)], petrel[, 3], nmax = 12, predacc = "VEcv")
VEcv [i] <- idwcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for IDW", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd=2)

n <- 20 # number of iterations, 60 to 100 is recommended.
measures <- NULL
for (i in 1:n) {
idwcv1 <- idwcv(swmud[, c(1,2)], swmud[, 3], predacc = "ALL")
measures <- rbind(measures, idwcv1$vecv)
}
plot(measures ~ c(1:n), xlab = "Iteration for IDW", ylab="VEcv (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`idwpred` *Generate spatial predictions using inverse distance weighting (IDW)*

Description

This function is to make spatial predictions using inverse distance weighting.

Usage

```
idwpred(longlat, trainy, longlat2, nmax = 12, idp = 2, ...)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples.
`trainy`	a vector of response, must have length equal to the number of rows in longlat.
`longlat2`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`nmax`	for a local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`idp`	numeric; specify the inverse distance weighting power.
`...`	other arguments passed on to gstat.

Value

A dataframe of longitude, latitude and predictions.

Note

This function is largely based on library(gstat).

Author(s)

Jin Li

References

Pebesma, E.J., 2004. Multivariable geostatistics in S: the gstat package. Computers & Geosciences, 30: 683-691.

Examples

```
## Not run:
library(sp)
data(swmud)
data(sw)
idwpred1 <- idwpred(swmud[, c(1,2)], swmud[, 3], sw, nmax = 12, idp = 2)
names(idwpred1)

## End(Not run)
```

`okcv` *Cross validation, n-fold for ordinary kriging (OK)*

Description

This function is a cross validation function for ordinary kriging.

Usage

```
okcv(
  longlat,
  trainy,
  cv.fold = 10,
  nmax = 12,
  transformation = "none",
  delta = 1,
  vgm.args = ("Sph"),
  anis = c(0, 1),
  alpha = 0,
  block = 0,
  predacc = "VEcv",
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples.
`trainy`	a vector of response, must have length equal to the number of rows in longlat.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`nmax`	for local kriging: the number of nearest observations that should be used for a kriging prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`transformation`	transform the response variable to normalise the data; can be "sqrt" for square root, "arcsine" for arcsine, "log" or "none" for non transformation. By default, "none" is used.
`delta`	numeric; to avoid log(0) in the log transformation.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`anis`	anisotropy parameters: see notes vgm in gstat for details.
`alpha`	direction in plane (x,y). see variogram in gstat for details.
`block`	block size. see krige in gstat for details.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to gstat.

Value

A list with the following components: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv only

Note

This function is largely based on rfcv in randomForest and some functions in library(gstat). When 'A zero or negative range was fitted to variogram' occurs, to allow gstat running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method. If it still occur for okpred function, different method should be used.

Author(s)

Jin Li

References

Li, J., 2013. Predictive Modelling Using Random Forest and Its Hybrid Methods with Geostatistical Techniques in Marine Environmental Geosciences, In: Christen, P., Kennedy, P., Liu, L., Ong, K.-L., Stranieri, A., Zhao, Y. (Eds.), The proceedings of the Eleventh Australasian Data Mining Conference (AusDM 2013), Canberra, Australia, 13-15 November 2013. Conferences in Research and Practice in Information Technology, Vol. 146.

A. Liaw and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Pebesma, E.J., 2004. Multivariable geostatistics in S: the gstat package. Computers & Geosciences, 30: 683-691.

Examples

```
## Not run:
library(sp)
data(swmud)
data(petrel)

okcv1 <- okcv(swmud[, c(1,2)], swmud[, 3], nmax = 7, transformation =
"arcsine", vgm.args = ("Sph"), predacc = "VEcv")
okcv1

n <- 20 # number of iterations,60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
okcv1 <- okcv(petrel[, c(1,2)], petrel[, 5], nmax = 12,
transformation = "arcsine", predacc = "VEcv")
VEcv [i] <- okcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for OK", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

n <- 20 # number of iterations, 60 to 100 is recommended.
```

```
measures <- NULL
for (i in 1:n) {
okcv1 <- okcv(petrel[, c(1,2)], petrel[, 3], nmax = 12, transformation =
"arcsine", predacc = "ALL")
measures <- rbind(measures, okcv1$vecv)
}
plot(measures ~ c(1:n), xlab = "Iteration for OK", ylab = "VEcv (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`okpred` *Generate spatial predictions using ordinary kriging (OK)*

Description

This function is to make spatial predictions using ordinary kriging.

Usage

```
okpred(
  longlat,
  trainy,
  longlat2,
  nmax = 12,
  transformation = "none",
  delta = 1,
  vgm.args = ("Sph"),
  anis = c(0, 1),
  alpha = 0,
  block = 0,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples.
`trainy`	a vector of response, must have length equal to the number of rows in longlat.
`longlat2`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`nmax`	for local kriging: the number of nearest observations that should be used for a kriging prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`transformation`	transform the response variable to normalise the data; can be "sqrt" for square root, "arcsine" for arcsine, "log" or "none" for non transformation. By default, "none" is used.

`delta`	numeric; to avoid log(0) in the log transformation.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`anis`	anisotropy parameters: see notes vgm in gstat for details.
`alpha`	direction in plane (x,y). see variogram in gstat for details.
`block`	block size. see krige in gstat for details.
`...`	other arguments passed on to gstat.

Value

A dataframe of longitude, latitude, predictions and variances.

Author(s)

Jin Li

References

Pebesma, E.J., 2004. Multivariable geostatistics in S: the gstat package. Computers & Geosciences, 30: 683-691.

Examples

```
## Not run:
library(sp)
data(swmud)
data(sw)
okpred1 <- okpred(swmud[, c(1,2)], swmud[, 3], sw, nmax = 7, transformation =
"arcsine", vgm.args = ("Sph"))
names(okpred1)

## End(Not run)
```

`petrel` *A dataset of seabed sediments in the Petrel sub-basin in Australia Exclusive Economic Zone*

Description

This dataset contains 237 samples of 9 variables including longitude (long), latitude (lat), mud content (mud), sand content (sand), gravel content (gravel), bathymetry (bathy), disttance to coast (dist), seabe relief (relief), seabed slope (slope).

Usage

```
data("petrel")
```

Format

A data frame with 237 observations on the following 9 variables.

`long` a numeric vector, decimal degree

`lat` a numeric vector, decimal degree

`mud` a numeric vector, percentage

`sand` a numeric vector, percentage

`gravel` a numeric vector, percentage

`bathy` a numeric vector, meter bellow sea level

`dist` a numeric vector, degree

`relief` a numeric vector, meter

`slope` a numeric vector, no unit

Details

For details, please check the reference.

Source

Li, J., 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods, The International Congress on Modelling and Simulation (MODSIM) 2013: Adelaide, pp. 394-400.

`petrel.grid` *A dataset of grids for producing spatial predictions of seabed sediment content in the Petrel sub-basin in Australia Exclusive Economic Zone*

Description

This dataset contains 248675 rows of 6 variables including longitude (long), latitude (lat), bathymetry (bathy), disttance to coast (dist), seabe relief (relief), seabed slope (slope).

Usage

```
data("petrel")
```

Format

A data frame with 248675 observations on the following 6 variables.

`long` a numeric vector, decimal degree

`lat` a numeric vector, decimal degree

`bathy` a numeric vector, meter bellow sea level

`dist` a numeric vector, degree

`relief` a numeric vector, meter

`slope` a numeric vector, no unit

Details

For details, please check the reference.

Source

Li, J., 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods, The International Congress on Modelling and Simulation (MODSIM) 2013: Adelaide, pp. 394-400.

`pred.acc` *Predictive error and accuracy measures for predictive models based on cross-validation*

Description

This function is used to calculate the mean error (me), mean absolute error (mae), mean squared error (mse), relative me (rme), relative mae (rmae), root mse (rmse), relative rmse (rrmse), variance explained by predictive models based on cross-validation (vecv), and Legates and McCabe's E1 (e1) for numerical data; and it also calculates correct classification rate (ccr), kappa (kappa), sensitivity (sens), specificity (spec), and true skill statistic (tss) for categorical data with the observed (obs) data specified as factor. They are based on the differences between the predicted values for and the observed values of validation samples for cross-validation. For 0 and 1 data, the observed values need to be specified as factor in order to use accuracy measures for categorical data. Moreover, sens, spec, tss and rmse are for categorical data with two levels (e.g. presence and absence data).

Usage

```
pred.acc(obs, pred)
```

Arguments

`obs`	a vector of observation values of validation samples.
`pred`	a vector of prediction values of predictive models for validation samples.

Value

A list with the following components: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1 for numerical data; ccr, kappa, sens, spec and tss for categorical data with two levels; and ccr, kappa for categorical data with more than two levels.

Author(s)

Jin Li

References

Li, J., 2016. Assessing spatial predictive models in the environmental sciences: accuracy measures, data variation and variance explained. Environmental Modelling & Software 80 1-8.

Li, J., 2017. Assessing the accuracy of predictive models for numerical data: Not r nor r2, why not? Then what? PLOS ONE 12 (8): e0183250.

Allouche, O., Tsoar, A., Kadmon, R., 2006. Assessing the accuracy of species distribution models: prevalence, kappa and true skill statistic (TSS). Journal of Applied Ecology 43 1223-1232.

Examples

```
set.seed(1234)
x <- sample(1:30, 30)
e <- rnorm(30, 1)
y <- x + e
pred.acc(x, y)

y <- 0.8 * x + e
pred.acc(x, y)
```

`RFcv` *Cross validation, n-fold for random forest (RF)*

Description

This function is a cross validation function for random forest.

Usage

```
RFcv(
  trainx,
  trainy,
  cv.fold = 10,
 mtry = if (!is.null(trainy) && !is.factor(trainy)) max(floor(ncol(trainx)/3), 1) else
    floor(sqrt(ncol(trainx))),
  ntree = 500,
  predacc = "ALL",
  ...
)
```

Arguments

`trainx`	a dataframe or matrix contains columns of predictor variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.

`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`ntree`	number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. By default, 500 is used.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to randomForest.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv. for categorical data: correct classification rate (ccr), kappa (kappa), sensitivity (sens), specificity (spec) and true skill statistic (tss)

Note

This function is largely based on rf.cv (see Li et al. 2013) and rfcv in randomForest.

Author(s)

Jin Li

References

Li, J., J. Siwabessy, M. Tran, Z. Huang, and A. Heap. 2013. Predicting Seabed Hardness Using Random Forest in R. Pages 299-329 in Y. Zhao and Y. Cen, editors. Data Mining Applications with R. Elsevier.

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(hard)
data(petrel)

rfcv1 <- RFcv(petrel[, c(1,2, 6:9)], petrel[, 5], predacc = "ALL")
rfcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
rfcv1 <- RFcv(petrel[, c(1,2,6:9)], petrel[, 5], predacc = "VEcv")
VEcv [i] <- rfcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for RF", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
```

```
abline(h = mean(VEcv), col = 'blue', lwd = 2)

n <- 20 # number of iterations, 60 to 100 is recommended.
measures <- NULL
for (i in 1:n) {
rfcv1 <- RFcv(hard[, c(4:6)], hard[, 17])
measures <- rbind(measures, rfcv1$ccr) # for kappa, replace ccr with kappa
}
plot(measures ~ c(1:n), xlab = "Iteration for RF", ylab = "Correct
classification rate  (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`rfidwcv` *Cross validation, n-fold for the hybrid method of random forest and inverse distance weighting (RFIDW)*

Description

This function is a cross validation function for the hybrid method of random forest and inverse distance weighting (RFIDW).

Usage

```
rfidwcv(
  longlat,
  trainx,
  trainy,
  cv.fold = 10,
  mtry = function(p) max(1, floor(sqrt(p))),
  ntree = 500,
  idp = 2,
  nmax = 12,
  predacc = "VEcv",
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.

`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`ntree`	number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. By default, 500 is used.
`idp`	numeric; specify the inverse distance weighting power.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to randomForest or gstat.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv.

Note

This function is largely based on rf.cv (see Li et al. 2013) and rfcv in randomForest.

Author(s)

Jin Li

References

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(petrel)

rfidwcv1 <- rfidwcv(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 5],
predacc = "ALL")
rfidwcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
rfidwcv1 <- rfidwcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "VEcv")
VEcv [i] <- rfidwcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for RFIDW", ylab = "VEcv (%)")
```

```
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

n <- 20 # number of iterations, 60 to 100 is recommended.
measures <- NULL
for (i in 1:n) {
rfidwcv1 <- rfidwcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "ALL")
measures <- rbind(measures, rfidwcv1$vecv)
}
plot(measures ~ c(1:n), xlab = "Iteration for RFIDW", ylab = "VEcv (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`rfidwpred` *Generate spatial predictions using the hybrid method of random forest and inverse distance weighting (RFIDW)*

Description

This function is to make spatial predictions using the hybrid method of random forest and inverse distance weighting (RFIDW).

Usage

```
rfidwpred(
  longlat,
  trainx,
  trainy,
  longlatpredx,
  predx,
  mtry = function(p) max(1, floor(sqrt(p))),
  ntree = 500,
  idp = 2,
  nmax = 12,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.

`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`ntree`	number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. By default, 500 is used.
`idp`	numeric; specify the inverse distance weighting power.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`...`	other arguments passed on to randomForest or gstat.

Value

A dataframe of longitude, latitude and predictions.

Author(s)

Jin Li

References

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
rfidwpred1 <- rfidwpred(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 3],
petrel.grid[, c(1,2)], petrel.grid, ntree = 500, idp = 2, nmax = 12)
names(rfidwpred1)

## End(Not run)
```

`rfokcv` *Cross validation, n-fold for the hybrid method of random forest and ordinary kriging (RFOK)*

Description

This function is a cross validation function for the hybrid method of random forest and ordinary kriging (RFOK).

Usage

```
rfokcv(
  longlat,
  trainx,
  trainy,
  cv.fold = 10,
  mtry = function(p) max(1, floor(sqrt(p))),
  ntree = 500,
  nmax = 12,
  vgm.args = ("Sph"),
  block = 0,
  predacc = "VEcv",
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`ntree`	number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. By default, 500 is used.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to randomForest or gstat.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv.

Note

This function is largely based on rf.cv (see Li et al. 2013) and rfcv in randomForest. When 'A zero or negative range was fitted to variogram' occurs, to allow gstat running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(petrel)

rfokcv1 <- rfokcv(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 5],
predacc = "ALL")
rfokcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
rfokcv1 <- rfokcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "VEcv")
VEcv [i] <- rfokcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for RFOK", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

n <- 20 # number of iterations, 60 to 100 is recommended.
measures <- NULL
for (i in 1:n) {
rfokcv1 <- rfokcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "ALL")
measures <- rbind(measures, rfokcv1$vecv)
}
plot(measures ~ c(1:n), xlab = "Iteration for RFOK", ylab = "VEcv (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`rfokpred` *Generate spatial predictions using the hybrid method of random forest and ordinary kriging (RFOK)*

Description

This function is to make spatial predictions using the hybrid method of random forest and ordinary kriging (RFOK).

Usage

```
rfokpred(
  longlat,
  trainx,
  trainy,
  longlatpredx,
  predx,
  mtry = function(p) max(1, floor(sqrt(p))),
  ntree = 500,
  nmax = 12,
  vgm.args = ("Sph"),
  block = 0,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`ntree`	number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. By default, 500 is used.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`...`	other arguments passed on to randomForest or gstat.

Value

A dataframe of longitude, latitude, predictions and variances. The variances are produced by OK based on the residuals of rf.

Note

This function is largely based rfcv in randomForest. When 'A zero or negative range was fitted to variogram' occurs, to allow OK running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
rfokpred1 <- rfokpred(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 3],
petrel.grid[, c(1,2)], petrel.grid, ntree = 500, nmax = 12, vgm.args =
("Sph"))
names(rfokpred1)

## End(Not run)
```

`rfokrfidwcv` *Cross validation, n-fold for the average of the hybrid method of random forest and ordinary kriging and the hybrid method of random forest and inverse distance weighting (RFOKRFIDW)*

Description

This function is a cross validation function for the average of the hybrid method of random forest and ordinary kriging and the hybrid method of random forest and inverse distance weighting (RFOKRFIDW).

Usage

```
rfokrfidwcv(
  longlat,
  trainx,
  trainy,
  cv.fold = 10,
  mtry = function(p) max(1, floor(sqrt(p))),
  ntree = 500,
```

```
  idp = 2,
  nmaxok = 12,
  nmaxidw = 12,
  vgm.args = ("Sph"),
  block = 0,
  predacc = "VEcv",
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`ntree`	number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. By default, 500 is used.
`idp`	numeric; specify the inverse distance weighting power.
`nmaxok`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for OK.
`nmaxidw`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for IDW.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to randomForest or gstat.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv.

Note

This function is largely based on rf.cv (see Li et al. 2013) and rfcv in randomForest. When 'A zero or negative range was fitted to variogram' occurs, to allow gstat running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(petrel)

rfokrfidwcv1 <- rfokrfidwcv(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 5],
predacc = "ALL")
rfokrfidwcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
rfokrfidwcv1 <- rfokrfidwcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "VEcv")
VEcv [i] <- rfokrfidwcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for RFOKRFIDW", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

n <- 20 # number of iterations, 60 to 100 is recommended.
measures <- NULL
for (i in 1:n) {
rfokrfidwcv1 <- rfokrfidwcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "ALL")
measures <- rbind(measures, rfokrfidwcv1$vecv)
}
plot(measures ~ c(1:n), xlab = "Iteration for RFOKRFIDW", ylab = "VEcv (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`rfokrfidwpred` *Generate spatial predictions using the average of the hybrid method of random forest and ordinary kriging and the hybrid method of random forest and inverse distance weighting (RFOKRFIDW)*

Description

This function is to make spatial predictions using the average of the hybrid method of random forest and ordinary kriging and the hybrid method of random forest and inverse distance weighting (RFOKRFIDW).

Usage

```
rfokrfidwpred(
  longlat,
  trainx,
  trainy,
  longlatpredx,
  predx,
  mtry = function(p) max(1, floor(sqrt(p))),
  ntree = 500,
  idp = 2,
  nmaxok = 12,
  nmaxidw = 12,
  vgm.args = (”Sph”),
  block = 0,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`ntree`	number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. By default, 500 is used.
`idp`	numeric; specify the inverse distance weighting power.
`nmaxok`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for OK.
`nmaxidw`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for IDW.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.

`block`	block size. see krige in gstat for details.
`...`	other arguments passed on to randomForest or gstat.

Value

A dataframe of longitude, latitude, predictions and variances. The variances are the same as the variances of rfokpred.

Note

This function is largely based rfcv in randomForest. When 'A zero or negative range was fitted to variogram' occurs, to allow OK running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
rfokrfidwpred1 <- rfokrfidwpred(petrel[, c(1,2)], petrel[, c(1,2, 6:9)],
petrel[, 3], petrel.grid[, c(1,2)], petrel.grid, ntree = 500, idp = 2,
nmaxok = 12, nmaxidw = 12)
names(rfokrfidwpred1)

## End(Not run)
```

`rfpred` *Generate spatial predictions using random forest (RF)*

Description

This function is to make spatial predictions using random forest.

Usage

```
rfpred(
  trainx,
  trainy,
  longlatpredx,
  predx,
  mtry = if (!is.null(trainy) && !is.factor(trainy)) max(floor(ncol(trainx)/3), 1) else
    floor(sqrt(ncol(trainx))),
  ntree = 500,
  ...
)
```

Arguments

`trainx`	a dataframe or matrix contains columns of predictor variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`ntree`	number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. By default, 500 is used.
`...`	other arguments passed on to randomForest.

Value

A dataframe of longitude, latitude and predictions.

Author(s)

Jin Li

References

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
rfpred1 <- rfpred(petrel[, c(1,2, 6:9)], petrel[, 5], petrel.grid[, c(1,2)],
petrel.grid, ntree = 500)
names(rfpred1)
```

```
## End(Not run)
```

---

rgcv *Cross validation, n-fold for random forest in ranger (RG)*

Description

This function is a cross validation function for random forest in ranger.

Usage

```
rgcv(
  trainx,
  trainy,
  cv.fold = 10,
 mtry = if (!is.null(trainy) && !is.factor(trainy)) max(floor(ncol(trainx)/3), 1) else
    floor(sqrt(ncol(trainx))),
  num.trees = 500,
  min.node.size = NULL,
  num.threads = NULL,
  verbose = FALSE,
  predacc = "ALL",
  ...
)
```

Arguments

`trainx`	a dataframe or matrix contains columns of predictor variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`mtry`	Number of variables to possibly split at in each node. Default is the (rounded down) square root of the number variables.
`num.trees`	number of trees. By default, 500 is used.
`min.node.size`	Default 1 for classification, 5 for regression.
`num.threads`	number of threads. Default is number of CPUs available.
`verbose`	Show computation status and estimated runtime.Default is FALSE.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to randomForest.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv. for categorical data: correct classification rate (ccr), kappa (kappa), sensitivity (sens), specificity (spec) and true skill statistic (tss)

Note

This function is largely based on RFcv.

Author(s)

Jin Li

References

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Wright, M. N. & Ziegler, A. (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. J Stat Softw 77:1-17. http://dx.doi.org/10.18637/jss.v077.i01.

Examples

```
## Not run:
data(hard)
data(petrel)

rgcv1 <- rgcv(petrel[, c(1,2, 6:9)], petrel[, 5], predacc = "ALL")
rgcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
rgcv1 <- rgcv(petrel[, c(1,2,6:9)], petrel[, 5], predacc = "VEcv")
VEcv [i] <- rgcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for RF", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

n <- 20 # number of iterations, 60 to 100 is recommended.
measures <- NULL
for (i in 1:n) {
rgcv1 <- rgcv(hard[, c(4:6)], hard[, 17])
measures <- rbind(measures, rgcv1$ccr) # for kappa, replace ccr with kappa
}
plot(measures ~ c(1:n), xlab = "Iteration for RF", ylab = "Correct
classification rate  (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`rgidwcv` *Cross validation, n-fold for the hybrid method of random forest in ranger and inverse distance weighting (RGIDW)*

Description

This function is a cross validation function for the hybrid method of random forest in ranger and inverse distance weighting (RGIDW).

Usage

```
rgidwcv(
  longlat,
  trainx,
  trainy,
  cv.fold = 10,
  mtry = function(p) max(1, floor(sqrt(p))),
  num.trees = 500,
  min.node.size = NULL,
  num.threads = NULL,
  verbose = FALSE,
  idp = 2,
  nmax = 12,
  predacc = "VEcv",
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`num.trees`	number of trees. By default, 500 is used.
`min.node.size`	Default 1 for classification, 5 for regression.
`num.threads`	number of threads. Default is number of CPUs available.
`verbose`	Show computation status and estimated runtime.Default is FALSE.
`idp`	numeric; specify the inverse distance weighting power.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.

`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to randomForest or gstat.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv.

Note

This function is largely based on rfidwcv.

Author(s)

Jin Li

References

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Wright, M. N. & Ziegler, A. (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. J Stat Softw 77:1-17. http://dx.doi.org/10.18637/jss.v077.i01.

Examples

```
## Not run:
data(petrel)

rgidwcv1 <- rgidwcv(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 5],
predacc = "ALL")
rgidwcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
rgidwcv1 <- rgidwcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "VEcv")
VEcv [i] <- rgidwcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for RFIDW", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

n <- 20 # number of iterations, 60 to 100 is recommended.
measures <- NULL
for (i in 1:n) {
rgidwcv1 <- rgidwcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "ALL")
measures <- rbind(measures, rgidwcv1$vecv)
}
```

```
plot(measures ~ c(1:n), xlab = "Iteration for RFIDW", ylab = "VEcv (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`rgidwpred` *Generate spatial predictions using the hybrid method of random forest in ranger and inverse distance weighting (RGIDW)*

Description

This function is to make spatial predictions using the hybrid method of random forest in ranger and inverse distance weighting (RGIDW).

Usage

```
rgidwpred(
  longlat,
  trainx,
  trainy,
  longlatpredx,
  predx,
  mtry = function(p) max(1, floor(sqrt(p))),
  num.trees = 500,
  min.node.size = NULL,
  type = "response",
  num.threads = NULL,
  verbose = FALSE,
  idp = 2,
  nmax = 12,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.

`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`num.trees`	number of trees. By default, 500 is used.
`min.node.size`	Default 1 for classification, 5 for regression.
`type`	Type of prediction. One of 'response', 'se', 'terminalNodes' with default 'response'. See ranger::predict.ranger for details.
`num.threads`	number of threads. Default is number of CPUs available.
`verbose`	Show computation status and estimated runtime.Default is FALSE.
`idp`	numeric; specify the inverse distance weighting power.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used.
`...`	other arguments passed on to randomForest or gstat.

Value

A dataframe of longitude, latitude and predictions.

Note

This function is largely based on rfidwpred.

Author(s)

Jin Li

References

Wright, M. N. & Ziegler, A. (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. J Stat Softw 77:1-17. http://dx.doi.org/10.18637/jss.v077.i01.

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
rgidwpred1 <- rgidwpred(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 3],
petrel.grid[, c(1,2)], petrel.grid, num.trees = 500, idp = 2, nmax = 12)
names(rgidwpred1)

## End(Not run)
```

`rgokcv` *Cross validation, n-fold for the hybrid method of random forest in ranger and ordinary kriging (RGFOK)*

Description

This function is a cross validation function for the hybrid method of random forest in ranger and ordinary kriging (RFOK).

Usage

```
rgokcv(
  longlat,
  trainx,
  trainy,
  cv.fold = 10,
  mtry = function(p) max(1, floor(sqrt(p))),
  num.trees = 500,
  min.node.size = NULL,
  num.threads = NULL,
  verbose = FALSE,
  nmax = 12,
  vgm.args = ("Sph"),
  block = 0,
  predacc = "VEcv",
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`num.trees`	number of trees. By default, 500 is used.
`min.node.size`	Default 1 for classification, 5 for regression.
`num.threads`	number of threads. Default is number of CPUs available.
`verbose`	Show computation status and estimated runtime.Default is FALSE.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12.

`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to randomForest or gstat.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and e1; or vecv.

Note

This function is largely based on rfokcv. When 'A zero or negative range was fitted to variogram' occurs, to allow gstat running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Wright, M. N. & Ziegler, A. (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. J Stat Softw 77:1-17. http://dx.doi.org/10.18637/jss.v077.i01.

Examples

```
## Not run:
data(petrel)

rgokcv1 <- rgokcv(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 5],
predacc = "ALL")
rgokcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
rgokcv1 <- rgokcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "VEcv")
VEcv [i] <- rgokcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for RFOK", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)
```

```
n <- 20 # number of iterations, 60 to 100 is recommended.
measures <- NULL
for (i in 1:n) {
rgokcv1 <- rgokcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "ALL")
measures <- rbind(measures, rgokcv1$vecv)
}
plot(measures ~ c(1:n), xlab = "Iteration for RFOK", ylab = "VEcv (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`rgokpred` *Generate spatial predictions using the hybrid method of random forest in ranger and ordinary kriging (RGOK)*

Description

This function is to make spatial predictions using the hybrid method of random forest in ranger and ordinary kriging (RGOK).

Usage

```
rgokpred(
  longlat,
  trainx,
  trainy,
  longlatpredx,
  predx,
  mtry = function(p) max(1, floor(sqrt(p))),
  num.trees = 500,
  min.node.size = NULL,
  type = "response",
  num.threads = NULL,
  verbose = FALSE,
  nmax = 12,
  vgm.args = ("Sph"),
  block = 0,
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.

`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`num.trees`	number of trees. By default, 500 is used.
`min.node.size`	Default 1 for classification, 5 for regression.
`type`	Type of prediction. One of 'response', 'se', 'terminalNodes' with default 'response'. See ranger::predict.ranger for details.
`num.threads`	number of threads. Default is number of CPUs available.
`verbose`	Show computation status and estimated runtime.Default is FALSE.
`nmax`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`...`	other arguments passed on to randomForest or gstat.

Value

A dataframe of longitude, latitude, predictions and variances. The variances are produced by OK based on the residuals of rf.

Note

This function is largely based rfokpred. When 'A zero or negative range was fitted to variogram' occurs, to allow OK running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Wright, M. N. & Ziegler, A. (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. J Stat Softw 77:1-17. http://dx.doi.org/10.18637/jss.v077.i01.

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
rgokpred1 <- rgokpred(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 3],
petrel.grid[, c(1,2)], petrel.grid, num.trees = 500, nmax = 12, vgm.args =
("Sph"))
names(rgokpred1)

## End(Not run)
```

`rgokrgidwcv` *Cross validation, n-fold for the average of the hybrid method of random forest in ranger (RG) and ordinary kriging and the hybrid method of RG and inverse distance weighting (RGOKRGIDW)*

Description

This function is a cross validation function for the average of the hybrid method of random forest in ranger (RG) and ordinary kriging and the hybrid method of RG and inverse distance weighting (RGOKRGIDW).

Usage

```
rgokrgidwcv(
  longlat,
  trainx,
  trainy,
  cv.fold = 10,
  mtry = function(p) max(1, floor(sqrt(p))),
  num.trees = 500,
  min.node.size = NULL,
  num.threads = NULL,
  verbose = FALSE,
  idp = 2,
  nmaxok = 12,
  nmaxidw = 12,
  vgm.args = ("Sph"),
  block = 0,
  predacc = "VEcv",
  ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).

`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`cv.fold`	integer; number of folds in the cross-validation. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`num.trees`	number of trees. By default, 500 is used.
`min.node.size`	Default 1 for classification, 5 for regression.
`num.threads`	number of threads. Default is number of CPUs available.
`verbose`	Show computation status and estimated runtime.Default is FALSE.
`idp`	numeric; specify the inverse distance weighting power.
`nmaxok`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for OK.
`nmaxidw`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for IDW.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`predacc`	can be either "VEcv" for vecv or "ALL" for all measures in function pred.acc.
`...`	other arguments passed on to randomForest or gstat.

Value

A list with the following components: for numerical data: me, rme, mae, rmae, mse, rmse, rrmse, vecv and `e1`; or vecv.

Note

This function is largely based on rfokrfidw. When 'A zero or negative range was fitted to variogram' occurs, to allow gstat running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Li, J. 2013. Predicting the spatial distribution of seabed gravel content using random forest, spatial interpolation methods and their hybrid methods. Pages 394-400 The International Congress on Modelling and Simulation (MODSIM) 2013, Adelaide.

Wright, M. N. & Ziegler, A. (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. J Stat Softw 77:1-17. http://dx.doi.org/10.18637/jss.v077.i01.

Examples

```
## Not run:
data(petrel)

rgokrgidwcv1 <- rgokrgidwcv(petrel[, c(1,2)], petrel[, c(1,2, 6:9)], petrel[, 5],
predacc = "ALL")
rgokrgidwcv1

n <- 20 # number of iterations, 60 to 100 is recommended.
VEcv <- NULL
for (i in 1:n) {
rgokrgidwcv1 <- rgokrgidwcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "VEcv")
VEcv [i] <- rgokrgidwcv1
}
plot(VEcv ~ c(1:n), xlab = "Iteration for RFOKRFIDW", ylab = "VEcv (%)")
points(cumsum(VEcv) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(VEcv), col = 'blue', lwd = 2)

n <- 20 # number of iterations, 60 to 100 is recommended.
measures <- NULL
for (i in 1:n) {
rgokrgidwcv1 <- rgokrgidwcv(petrel[, c(1,2)], petrel[, c(1,2,6:9)], petrel[, 5],
predacc = "ALL")
measures <- rbind(measures, rgokrgidwcv1$vecv)
}
plot(measures ~ c(1:n), xlab = "Iteration for RFOKRFIDW", ylab = "VEcv (%)")
points(cumsum(measures) / c(1:n) ~ c(1:n), col = 2)
abline(h = mean(measures), col = 'blue', lwd = 2)

## End(Not run)
```

`rgokrgidwpred` *Generate spatial predictions using the average of the hybrid method of random forest in ranger (RG) and ordinary kriging and the hybrid method of RG and inverse distance weighting (RGOKRGIDW)*

Description

This function is to make spatial predictions using the average of the hybrid method of random forest in ranger (RG) and ordinary kriging and the hybrid method of RG and inverse distance weighting (RGOKRGIDW).

Usage

```
rgokrgidwpred(
  longlat,
  trainx,
```

```
    trainy,
    longlatpredx,
    predx,
    mtry = function(p) max(1, floor(sqrt(p))),
    num.trees = 500,
    min.node.size = NULL,
    type = "response",
    num.threads = NULL,
    verbose = FALSE,
    idp = 2,
    nmaxok = 12,
    nmaxidw = 12,
    vgm.args = ("Sph"),
    block = 0,
    ...
)
```

Arguments

`longlat`	a dataframe contains longitude and latitude of point samples (i.e., trainx and trainy).
`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`mtry`	a function of number of remaining predictor variables to use as the mtry parameter in the randomForest call.
`num.trees`	number of trees. By default, 500 is used.
`min.node.size`	Default 1 for classification, 5 for regression.
`type`	Type of prediction. One of 'response', 'se', 'terminalNodes' with default 'response'. See ranger::predict.ranger for details.
`num.threads`	number of threads. Default is number of CPUs available.
`verbose`	Show computation status and estimated runtime.Default is FALSE.
`idp`	numeric; specify the inverse distance weighting power.
`nmaxok`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for OK.
`nmaxidw`	for local predicting: the number of nearest observations that should be used for a prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, 12 observations are used for IDW.
`vgm.args`	arguments for vgm, e.g. variogram model of response variable and anisotropy parameters. see notes vgm in gstat for details. By default, "Sph" is used.
`block`	block size. see krige in gstat for details.
`...`	other arguments passed on to randomForest or gstat.

Value

A dataframe of longitude, latitude, predictions and variances. The variances are the same as the variances of rfokpred.

Note

This function is largely based rfokrfidwpred. When 'A zero or negative range was fitted to variogram' occurs, to allow OK running, the range was set to be positive by using min(vgm1$dist). In this case, caution should be taken in applying this method, although sometimes it can still outperform IDW and OK.

Author(s)

Jin Li

References

Liaw, A. and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18-22.

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
rgokrgidwpred1 <- rgokrgidwpred(petrel[, c(1,2)], petrel[, c(1,2, 6:9)],
petrel[, 3], petrel.grid[, c(1,2)], petrel.grid, num.trees = 500, idp = 2,
nmaxok = 12, nmaxidw = 12)
names(rgokrgidwpred1)

## End(Not run)
```

`rgpred` *Generate spatial predictions using random forest in ranger (RG)*

Description

This function is to make spatial predictions using random forest in ranger.

Usage

```
rgpred(
  trainx,
  trainy,
  longlatpredx,
  predx,
 mtry = if (!is.null(trainy) && !is.factor(trainy)) max(floor(ncol(trainx)/3), 1) else
```

```
    floor(sqrt(ncol(trainx))),
  num.trees = 500,
  min.node.size = NULL,
  type = "response",
  num.threads = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

`trainx`	a dataframe or matrix contains columns of predictor variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.
`longlatpredx`	a dataframe contains longitude and latitude of point locations (i.e., the centres of grids) to be predicted.
`predx`	a dataframe or matrix contains columns of predictive variables for the grids to be predicted.
`mtry`	Number of variables to possibly split at in each node. Default is the (rounded down) square root of the number variables.
`num.trees`	number of trees. By default, 500 is used.
`min.node.size`	Default 1 for classification, 5 for regression.
`type`	Type of prediction. One of 'response', 'se', 'terminalNodes' with default 'response'. See ranger::predict.ranger for details.
`num.threads`	number of threads. Default is number of CPUs available.
`verbose`	Show computation status and estimated runtime.Default is FALSE.
`...`	other arguments passed on to randomForest.

Value

A dataframe of longitude, latitude and predictions.

Note

This function is largely based on rfpred.

Author(s)

Jin Li

References

Wright, M. N. & Ziegler, A. (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. J Stat Softw 77:1-17. http://dx.doi.org/10.18637/jss.v077.i01.

Examples

```
## Not run:
data(petrel)
data(petrel.grid)
set.seed(1234)
rgpred1 <- rgpred(petrel[, c(1,2, 6:9)], petrel[, 5], petrel.grid[, c(1,2)],
petrel.grid, num.trees = 500)
names(rgpred1)

## End(Not run)
```

`rvi` *Relative variable influence based on generalized boosted regression modeling (gbm)*

Description

This function is to to derive a relative variable influence based on generalized boosted regression modeling.

Usage

```
rvi(
  trainx,
  trainy,
  var.monotone = rep(0, ncol(trainx)),
  family = "gaussian",
  n.trees = 3000,
  learning.rate = 0.001,
  interaction.depth = 2,
  bag.fraction = 0.5,
  train.fraction = 1,
  n.minobsinnode = 10,
  cv.fold = 10,
  weights = rep(1, nrow(trainx)),
  keep.data = FALSE,
  verbose = TRUE,
  n.cores = 6,
  ...
)
```

Arguments

`trainx`	a dataframe or matrix contains columns of predictive variables.
`trainy`	a vector of response, must have length equal to the number of rows in trainx.

`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome. By default, a vector of 0 is used.
`family`	either a character string specifying the name of the distribution to use or a list with a component name specifying the distribution and any additional parameters needed. See gbm for details. By default, "gaussian" is used.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, 3000 is used.
`learning.rate`	a shrinkage parameter applied to each tree in the expansion. Also known as step-size reduction.
`interaction.depth`	the maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc. By default, 2 is used.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion. By default, 0.5 is used.
`train.fraction`	The first train.fraction * nrows(data) observations are used to fit the gbm and the remainder are used for computing out-of-sample estimates of the loss function.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight. By default, 10 is used.
`cv.fold`	integer; number of cross-validation folds to perform within gbm. if > 1, then apply n-fold cross validation; the default is 10, i.e., 10-fold cross validation that is recommended.
`weights`	an optional vector of weights to be used in the fitting process. Must be positive but do not need to be normalized. If keep.data = FALSE in the initial call to gbm then it is the user's responsibility to resupply the weights to gbm.more. By default, a vector of 1 is used.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object. Keeping the data and index makes subsequent calls to gbm.more faster at the cost of storing an extra copy of the dataset. By default, 'FALSE' is used.
`verbose`	If TRUE, gbm will print out progress and performance indicators. By default, 'TRUE' is used.
`n.cores`	The number of CPU cores to use. See gbm for details. By default, 6 is used.
`...`	other arguments passed on to gbm.

Value

A dataframe of variables (var), and relative influence (rel.inf) arranged from the most influential to the least influential

Note

This function is largely based on gbm.

Author(s)

Jin Li

References

Greg Ridgeway with contributions from others (2015). gbm: Generalized Boosted Regression Models. R package version 2.1.1. https://CRAN.R-project.org/package=gbm

Examples

```
## Not run:
data(sponge)
set.seed(1234)
rvi1 <- rvi(sponge[, -c(3)], sponge[, 3], family = "poisson", n.cores=2)
names(rvi1)
# The least influence variable
livar <- as.character(rvi1$gbm.rvi$var[dim(sponge[, -3])[2]])
livar

# The least influence variable to be removed from the training dataset
rmvar <- which(names(sponge[, -c(3)]) == livar)
rmvar

## End(Not run)
```

`sponge` *A dataset of sponge species richness in the Timor Sea region, northern Australia marine margin*

Description

This dataset contains 77 samples of 8 variables including longitude (easting), latitude (northing), sponge, topographic position index (tpi3), variance of backscatter (var7), entropy (entro7), backscatter at incidence angle 11 degree (bs11), and backscatter at incidence angle 34 degree (bs34).

Usage

```
data("sponge")
```

Format

A data frame with 77 observations on the following 8 variables.

`easting` a numeric vector, m

`northing` a numeric vector, m

`sponge` a numeric vector, no unit

`tpi3` a numeric vector, no unit

`var7` a numeric vector, dB^2

`entro7` a numeric vector, no unit

`bs11` a numeric vector, dB

`bs34` a numeric vector, dB

Details

For details, please see the source. This dataset was published as an appendix of the paper listed in the source. Where the long and lat were replaced with easting and northing for prediction purpose.

Source

Li, J., B. Alvarez, J. Siwabessy, M. Tran, Z. Huang, R. Przeslawski, L. Radke, F. Howard, and S. Nichol. 2017. Application of random forest, generalised linear model and their hybrid methods with geostatistical techniques to count data: Predicting sponge species richness. Environmental Modelling & Software, 97: 112-129

`sponge.grid` *A dataset of predictors for generating sponge species richness in a selected region in the Timor Sea region, northern Australia marine margin*

Description

This dataset contains 95530 rows of 7 predictive variables including longitude (easting), latitude (northing), topographic position index (tpi3), variance of backscatter (var7), entropy (entro7), backscatter at incidence angle 11 degree (bs11), and backscatter at incidence angle 34 degree (bs34).

Usage

```
data("sponge.grid")
```

Format

A data frame with 95530 rows on the following 7 variables.

`easting` a numeric vector, m

`northing` a numeric vector, m

`tpi3` a numeric vector, no unit

`var7` a numeric vector, dB^2

`entro7` a numeric vector, no unit

`bs11` a numeric vector, dB

`bs34` a numeric vector, dB

Details

For details, please see the source. This dataset was used to produce the figure of predictions in the paper listed in the source.

Source

Li, J., B. Alvarez, J. Siwabessy, M. Tran, Z. Huang, R. Przeslawski, L. Radke, F. Howard, and S. Nichol. 2017. Application of random forest, generalised linear model and their hybrid methods with geostatistical techniques to count data: Predicting sponge species richness. Environmental Modelling & Software,97: 112-129.

`sw` *A dataset of grids for producing spatial predictions of seabed mud content in the southwest Australia Exclusive Economic Zone*

Description

This dataset contains 500703 rows of 2 variables including longitude (long), latitude (lat).

Usage

```
data("sw")
```

Format

A data frame with 500703 rows on the following 2 variables.

`long`	a numeric vector, decimal degree
`lat`	a numeric vector, decimal degree

Details

For details, please check the source.

Source

Li, J., Potter, A., Huang, Z., Daniell, J.J., Heap, A., 2010. Predicting Seabed Mud Content across the Australian Margin: Comparison of Statistical and Mathematical Techniques Using a Simulation Experiment. Geoscience Australia, 2010/11, 146pp.

`swmud` *A dataset of seabed mud content in the southwest Australia Exclusive Economic Zone*

Description

This dataset contains 177 samples of 3 variables including longitude (long), latitude (lat), mud content (mud).

Usage

```
data("swmud")
```

Format

A data frame with 177 observations on the following 3 variables.

`long`	a numeric vector, decimal degree
`lat`	a numeric vector, decimal degree
`mud`	a numeric vector, percentage

Details

For details, please check the source.

Source

Li, J., Potter, A., Huang, Z., Daniell, J.J., Heap, A., 2010. Predicting Seabed Mud Content across the Australian Margin: Comparison of Statistical and Mathematical Techniques Using a Simulation Experiment. Geoscience Australia, 2010/11, 146pp.

`tovecv` *Convert error measures to vecv*

Description

tovecv can be used to convert existing predictive error measures to vecv. For the definition of vecv, please see function vecv in library (spm). The error measures considered are mean square error (mse), root mse (rmse), relative rmse (rrmse), standardised rmse (srmse) and mean square reduced error (msre).

Usage

```
tovecv(n, mu, s, m, measure = c("mse", "rmse", "rrmse", "srmse", "msre"))
```

Arguments

`n`	sample number of validation samples.
`mu`	mean of validation samples.
`s`	standard deviation of validation samples.
`m`	value of an error measure.
`measure`	a type of error measure (i.e. "mse", "rmse", "rrmse", "srmse" or "msre").

Value

a numeric number.

Author(s)

Jin Li

References

Li, J., 2016. Assessing spatial predictive models in the environmental sciences: accuracy. measures, data variation and variance explained. Environmental Modelling & Software 80 1-8.

Li, J., 2017. Assessing the accuracy of predictive models for numerical data: Not r nor r2, why not? Then what? PLOS ONE 12 (8): e0183250.

Examples

```
n <- 300
mu <- 15.5
sd <- 8.80
mse <- 50.43
rmse <- sqrt(mse)
rrmse <- rmse / mu * 100
srmse <- rmse / sd
msre <- mse / sd ^ 2
tovecv(n=n, mu=mu, s=sd, m=mse, measure="mse")

tovecv(n=n, mu=mu, s=sd, m=rmse, measure="rmse")

tovecv(n=n, mu=mu, s=sd, m=rrmse, measure="rrmse")

tovecv(n=n, mu=mu, s=sd, m=srmse, measure="srmse")

tovecv(n=n, mu=mu, s=sd, m=msre, measure="msre")
```

---

vecv *Variance explained by predictive models based on cross-validation*

Description

vecv is used to calculate the variance explained by predictive models based on cross-validation. The vecv is based on the differences between the predicted values for, and the observed values of, validation samples for cross-validation. It measures the proportion of variation in the validation data explained by the predicted values obtained from predictive models based on cross-validation.

Usage

```
vecv(obs, pred)
```

Arguments

`obs`	observation values of validation samples.
`pred`	prediction values of predictive models for validation samples.

Value

a numeric number.

Author(s)

Jin Li

References

Li, J., 2016. Assessing spatial predictive models in the environmental sciences: accuracy. measures, data variation and variance explained. Environmental Modelling & Software 80 1-8.

Examples

```
set.seed(1234)
x <- sample(1:30, 30)
e <- rnorm(30, 1)
y <- x + e
vecv(x, y)

y <- 0.8 * x + e
vecv(x, y)
```

Index