Package ‘CAST’

October 11, 2018

Type Package

Title 'caret' Applications for Spatial-Temporal Models

Version 0.3.0

Author Hanna Meyer [cre, aut],
Chris Reudenbach [ctb],
Marvin Ludwig [ctb],
Thomas Nauss [ctb]

Maintainer Hanna Meyer <hanna.meyer@geo.uni-marburg.de>

Description Supporting functionality to run 'caret' with spatial or spatial-temporal data. 'caret' is a frequently used package for model training and prediction using machine learning. This package includes functions to improve spatial-temporal modelling tasks using 'caret'. It prepares data for Leave-Location-Out and Leave-Time-Out cross-validation which are target-oriented validation strategies for spatial-temporal models. To decrease overfitting and improve model performances, the package implements a forward feature selection that selects suitable predictor variables in view to their contribution to the target-oriented performance.

License GPL (>= 3) | file LICENSE

URL https://github.com/environmentalinformatics-marburg/CAST

Encoding UTF-8

LazyData true

Depends R (>= 3.1.0)

Imports caret, stats, utils, ggplot2

Suggests doParallel, GSIF, randomForest, lubridate, raster, sp, knitr,
mapview, rmarkdown

RoxygenNote 6.1.0

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2018-10-11 04:30:02 UTC

R topics documented:

---

bss — *Best subset feature selection*

Description

Evaluate all combinations of predictors during model training

Usage

```
bss(predictors, response, method = "rf",
  metric = ifelse(is.factor(response), "Accuracy", "RMSE"),
  maximize = ifelse(metric == "RMSE", FALSE, TRUE),
  trControl = caret::trainControl(), tuneLength = 3, tuneGrid = NULL,
  seed = 100, verbose = TRUE, ...)
```

Arguments

`predictors`	see `train`
`response`	see `train`
`method`	see `train`
`metric`	see `train`
`maximize`	see `train`
`trControl`	see `train`
`tuneLength`	see `train`
`tuneGrid`	see `train`
`seed`	A random number
`verbose`	Logical. Should information about the progress be printed?
`...`	arguments passed to the classification or regression routine (such as randomForest).

Details

bss is an alternative to `ffs` and ideal if the training set is small. Models are iteratively fitted using all different combinations of predictor variables. Hence, 2^X models are calculated. Dont try running bss on very large datasets because the computation time is much higher compared to `ffs`.

The internal cross validation can be run in parallel. See information on parallel processing of carets train functions for details.

Value

A list of class train. Beside of the usual train content the object contains the vector "selectedvars" and "selectedvars_perf" that give the best variables selected as well as their corresponding performance. It also contains "perf_all" that gives the performance of all model runs.

Note

This validation is particulary suitable for spatial leave-location-out cross validations where variable selection MUST be based on the performance of the model on the hold out station. Note that bss is very slow since all combinations of variables are tested. A more time efficient alternative is the forward feature selection (ffs) (ffs).

Author(s)

Hanna Meyer

See Also

train,ffs, trainControl,CreateSpacetimeFolds

Examples

```
## Not run:
data(iris)
bssmodel <- bss(iris[,1:4],iris$Species)
bssmodel$perf_all

## End(Not run)
```

CAST *'caret' Applications for Spatial-Temporal Models*

Description

Supporting functionality to run 'caret' with spatial or spatial-temporal data. 'caret' is a frequently used package for model training and prediction using machine learning. This package includes functions to improve spatial-temporal modelling tasks using 'caret'. It prepares data for Leave-Location-Out and Leave-Time-Out cross-validation which are target-oriented validation strategies for spatial-temporal models. To decrease overfitting and improve model performances, the package implements a forward feature selection that selects suitable predictor variables in view to their contribution to the target-oriented performance.

Details

'caret' Applications for Spatio-Temporal models

Author(s)

Hanna Meyer, Christoph Reudenbach, Marvin Ludwig, Thomas Nauss
Maintainer: Hanna Meyer <hanna.meyer@geo.uni-marburg.de>

CreateSpacetimeFolds *Create Space-time Folds*

Description

Create spatial, temporal or spatio-temporal Folds for cross validation

Usage

```
CreateSpacetimeFolds(x, spacevar = NA, timevar = NA, k = 10,
  seed = sample(1:1000, 1))
```

Arguments

x	data.frame containing spatio-temporal data
spacevar	Character indicating which column of x identifies the spatial units (e.g. ID of weather stations)
timevar	Character indicating which column of x identifies the temporal units (e.g. the day of the year)
k	numeric. Number of folds. If spacevar or timevar is NA and a leave one location out or leave one time step out cv should be performed, set k to the number of unique spatial or temporal units.
seed	numeric. See ?seed

Value

A list that contains a list for model training and a list for model validation that can directly be used as "index" and "indexOut" in caret's trainControl function

Note

Standard k-fold cross-validation can lead to considerable misinterpretation in spatial-temporal modelling tasks. This function can be used to prepare a Leave-Location-Out, Leave-Time-Out or Leave-Location-and-Time-Out cross-validation as target-oriented validation strategies for spatial-temporal prediction tasks. See Meyer et al. (2018) for further information.

Author(s)

Hanna Meyer

References

Meyer, H., Reudenbach, C., Hengl, T., Katurji, M., Nauß, T. (2018): Improving performance of spatio-temporal machine learning models using forward feature selection and target-oriented validation. Environmental Modelling & Software 101: 1-9.

See Also

trainControl,ffs

Examples

```
library(GSIF)
data(cookfarm)
### Prepare for 10-fold Leave-Location-and-Time-Out cross validation
indices <- CreateSpacetimeFolds(cookfarm$readings,"SOURCEID","Date")
str(indices)
### Prepare for 10-fold Leave-Location-Out cross validation
indices <- CreateSpacetimeFolds(cookfarm$readings,spacevar="SOURCEID")
str(indices)
### Prepare for leave-One-Location-Out cross validation
indices <- CreateSpacetimeFolds(cookfarm$readings,spacevar="SOURCEID",
k=length(unique(cookfarm$readings$SOURCEID)))
str(indices)
```

---

ffs *Forward feature selection*

Description

A simple forward feature selection algorithm

Usage

```
ffs(predictors, response, method = "rf",
  metric = ifelse(is.factor(response), "Accuracy", "RMSE"),
  maximize = ifelse(metric == "RMSE", FALSE, TRUE), withinSE = FALSE,
  trControl = caret::trainControl(), tuneLength = 3, tuneGrid = NULL,
  seed = sample(1:1000, 1), verbose = TRUE, ...)
```

Arguments

predictors	see train
response	see train
method	see train
metric	see train
maximize	see train

`withinSE`	Logical Models are only selected if they are better than the currently best models Standard error
`trControl`	see `train`
`tuneLength`	see `train`
`tuneGrid`	see `train`
`seed`	A random number used for model training
`verbose`	Logical. Should information about the progress be printed?
`...`	arguments passed to the classification or regression routine (such as randomForest).

Details

Models with two predictors are first trained using all possible pairs of predictor variables. The best model of these initial models is kept. On the basis of this best model the predictor variables are iteratively increased and each of the remaining variables is tested for its improvement of the currently best model. The process stops if none of the remaining variables increases the model performance when added to the current best model.

The internal cross validation can be run in parallel. See information on parallel processing of carets train functions for details.

Using withinSE will favour models with less variables and probably shorten the calculation time

Value

A list of class train. Beside of the usual train content the object contains the vector "selectedvars" and "selectedvars_perf" that give the order of the best variables selected as well as their corresponding performance (starting from the first two variables). It also contains "perf_all" that gives the performance of all model runs.

Note

This validation is particulary suitable for spatial leave-location-out cross validations where variable selection MUST be based on the performance of the model on the hold out station. See Meyer et al. (2018) for further details.

Author(s)

Hanna Meyer

References

- Gasch, C.K., Hengl, T., Gräler, B., Meyer, H., Magney, T., Brown, D.J. (2015): Spatio-temporal interpolation of soil water, temperature, and electrical conductivity in 3D+T: the Cook Agronomy Farm data set. Spatial Statistics 14: 70-90.
- Meyer, H., Reudenbach, C., Hengl, T., Katurji, M., Nauß, T. (2018): Improving performance of spatio-temporal machine learning models using forward feature selection and target-oriented validation. Environmental Modelling & Software 101: 1-9.

See Also

train,bss, trainControl,CreateSpacetimeFolds

Examples

```
## Not run:
data(iris)
ffsmodel <- ffs(iris[,1:4],iris$Species)
ffsmodel$selectedvars
ffsmodel$selectedvars_perf

## End(Not run)

# or perform model with target-oriented validation (LLO CV)
#the example is taken from the GSIF package and is described
#in Gasch et al. (2015). The ffs approach for this dataset is described in
#Meyer et al. (2018). Due to high computation time needed, only a small and thus not robust example
#is shown here.

## Not run:
#run the model on three cores:
library(doParallel)
cl <- makeCluster(3)
registerDoParallel(cl)

#load and prepare dataset:
dat <- get(load(system.file("extdata","Cookfarm.RData",package="CAST")))
trainDat <- dat[dat$altitude==-0.3&year(dat$Date)==2012&week(dat$Date)%in%c(13:14),]

#visualize dataset:
ggplot(data = trainDat, aes(x=Date, y=VW)) + geom_line(aes(colour=SOURCEID))

#create folds for Leave Location Out Cross Validation:
set.seed(10)
indices <- CreateSpacetimeFolds(trainDat,spacevar = "SOURCEID",k=3)
ctrl <- trainControl(method="cv",index = indices$index)

#define potential predictors:
predictors <- c("DEM","TWI","BLD","Precip_cum","cday","MaxT_wrcc",
"Precip_wrcc","NDRE.M","Bt","MinT_wrcc","Northing","Easting")

#run ffs model with Leave Location out CV
set.seed(10)
ffsmodel <- ffs(trainDat[,predictors],trainDat$VW,method="rf",
tuneLength=1,trControl=ctrl)
ffsmodel

#compare to model without ffs:
model <- ffs(trainDat[,predictors],trainDat$VW,method="rf",
tuneLength=1, trControl=ctrl)
model
stopCluster(cl)
```

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

---

plot_ffs *Plot results of a Forward feature selection or best subset selection*

Description

A plotting function for a forward feature selection result. Each point is the mean performance of a model run. Error bars represent the standard errors from cross validation. Marked points show the best model from each number of variables until a further variable could not improve the results.

Usage

```
plot_ffs(ffs_model, palette = rainbow, reverse = FALSE,
  marker = "black", size = 1.5, lwd = 0.5, pch = 21)
```

Arguments

ffs_model	Result of a forward feature selection see ffs
palette	A color palette
reverse	Character. Should the palette be reversed?
marker	Character. Color to mark the best models
size	Numeric. Size of the points
lwd	Numeric. Width of the error bars
pch	Numeric. Type of point marking the best models

Author(s)

Marvin Ludwig and Hanna Meyer

See Also

ffs, bss

Examples

```
## Not run:
data(iris)
ffsmodel <- ffs(iris[,1:4],iris$Species)
plot_ffs(ffsmodel)

## End(Not run)
```

Index