Package: Rstg (via r-universe)

October 10, 2024

Title STG : Feature Selection using STochastic Gates

Version 0.0.1

Description 'STG' is a method for feature selection in neural network. The procedure is based on probabilistic relaxation of the l0 norm of features, or the count of the number of selected features. The framework simultaneously learns either a nonlinear regression or classification function while selecting a small subset of features. Read more: Yamada et al. (2020) <https://proceedings.mlr.press/v119/yamada20a.html>.

Imports reticulate (>= 1.4)

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author Yutaro Yamada [aut, cre]

Maintainer Yutaro Yamada <yutaro.yamada@yale.edu>

Date/Publication 2021-12-13 11:10:05 UTC

Repository https://runopti.r-universe.dev

RemoteUrl https://github.com/cran/Rstg

RemoteRef HEAD

RemoteSha 7633af48d7c5041efa89227eae16f78f013477e0

Contents

pystg_is_available *Check whether STG Python package is available and can be loaded*

Description

This is used to avoid running tests on CRAN

Usage

```
pystg_is_available()
```

Value

No return value, called for side effects

stg *STG: Feature Selection using Stochastic Gates*

Description

STG is a method for feature selection in neural network estimation problems. The new procedure is based on probabilistic relaxation of the l0 norm of features, or the count of the number of selected features. STG simultaneously learns either a nonlinear regression or classification function while selecting a small subset of features, as described in Yamada, et al, ICML 2020.

Usage

```
stg(
  task_type,
  input_dim,
  output_dim,
  hidden_dims,
  activation = "relu",
  sigma = 0.5,
  lam = 0.1,
  optimizer = "Adam",
  learning_rate = 0.001,
  batch_size = 100L,
  freeze_onward = NULL,
  feature_selection = TRUE,
  weight_decay = 0.001,
  random_state = 123L,
  device = "cpu"
)
```

Arguments

task_type	string choose 'regression', 'classification', or 'cox'
input_dim	integer The number of features of your data (input dimension)
output_dim	integer The number of classes for 'classification'. Should be 1 for 'regression' and 'cox'
hidden_dims	vector of integers,optional,default:c(60, 20, 3) architecture vector of the neural network
activation	string the type of activation functions.
sigma	float the noise level for the gaussian distribution
lam	float the regularization parameter
optimizer	string choose 'Adam' or 'SGD'
learning_rate	float
batch_size	int
freeze_onward	integer, default:NULL the network parameters will be frozen after 'freeze_onward' epoch. This is to train the gate parameters.
feature_selection	bool
weight_decay	float
random_state	integer
device	string 'cpu' or 'cuda' (if you have GPU)

Value

a "stg" object is returned.

Examples

```
if (pystg_is_available()){
n_size <- 1000L;
p_size <- 20L;
stg.model <- stg(task_type='regression', input_dim=p_size, output_dim=1L,
hidden_dims = c(500,50, 10), activation='tanh',
optimizer='SGD', learning_rate=0.1, batch_size=n_size,
feature_selection=TRUE, sigma=0.5, lam=0.1, random_state=0.1)
}
```

Index