Package ‘LinearizedSVR’

February 19, 2015

Type Package

Title Linearized Support Vector Regression

Version 1.3

Author Sriharsha Veeramachaneni and Ken Williams

Maintainer Ken Williams <ken.williams@windlogics.com>

Description Train and predict using fast prototype-based Linearized Support-Vector Regression methods.

License BSD_3_clause + file LICENSE

Copyright WindLogics Inc., 2012-2014

Imports kernlab, LiblineaR, expectreg

Suggests testthat

NeedsCompilation no

Repository CRAN

Date/Publication 2014-08-01 17:36:40

R topics documented:

LinearizedSVR-package ......................................................... 2
LinearizedSVRTrain .......................................................... 3
predict.LinearizedSVR ....................................................... 4
sigma.est ................................................................. 5

Index 6
Description

Train and predict using prototype-based Linearized Support-Vector Regression methods.

Details

Linearized Support Vector Regression is a kernel regression method where the basis is chosen a priori (instead of by the training algorithm as is done by the traditional support vector regression method). This allows the training method to take advantage of fast linear methods like (LiblineaR, lm) etc.

The choice of the basis involves picking the prototypes, which can be done randomly or by k-means, and the kernel. The complexity of the learned model can be controlled by the number of prototypes and the choice of the kernel. See [1] for some theoretical justification for the approach.

In order to take advantage of LiblineaR, a fast linear classifier whose training scales linearly with the number of examples, we reduce regression to classification using the insight proposed in [2]. Given a training dataset \( \{x_i, y_i\}_{i=1}^N \) where we need to build a regression model to predict \( y \) from \( x \) we construct a \( \{0, 1\} \) classification problem with data \( \{(x_i, y_i + \epsilon), 1\}_{i=1}^N \cup \{(x_i, y_i - \epsilon), 0\}_{i=1}^N \). That is, we move the data "up" and "down" by epsilon and then attempt to find the boundary between the two sets. The classification boundary then determines the regression surface. At predict time, in order to obtain the regression value for a test \( x \) we find the \( y \) that would lie on the boundary.

After transforming the data into the chosen basis, it is trivial to use any other linear methods (e.g., quantreg, rlm, expect.reg) to obtain the corresponding non-linear version. We provide expectile regression as an example.

Choice of prototypes: We provide two ways to pick the prototypes: random and Kmeans. When clusterY is TRUE, the Kmeans method also uses the target variable (Y). This presumably provides better prototype selection for regression. The parameter nump specifies the number of prototypes to be used.

The kernel and kpar parameters can be any from the kernlab package. The epsilon.up and epsilon.down parameters allows the epsilon insensitivity band for the regression to be asymmetric.

References


See Also

LinearizedSVRTrain
LinearizedSVRTrain

Description
Train a prototype-based Linearized Support-Vector Regression model

Usage
LinearizedSVRTrain(x, y, C = 1, epsilon = 0.01, nump = floor(sqrt(N)),
ktype = rbf dot, kpar, prototypes = c("kmeans", "random"),
clusterY = FALSE, epsilon.up = epsilon, epsilon.down = epsilon,
expectile = NULL, scale = TRUE, sigest = sigma.est)

Arguments
X matrix of examples, one example per row.
Y vector of target values. Must be the same length as the number of rows in X.
C cost of constraints violation
epsilon tolerance of termination criterion for optimization
nump number of prototypes by which to represent each example in X
ktype kernel-generating function, typically from the kernlab package
kpar a list of any parameters necessary for ktype. See Details.
prototypes the method by which prototypes will be chosen
clusterY whether to cluster X and Y jointly when using prototypes="kmeans". Otherwise X is clustered without influence from Y.
epsilon.up allows you to use a different setting for epsilon in the positive direction.
epsilon.down allows you to use a different setting for epsilon in the negative direction.
expectile if non-null, do expectile regression using the given expectile value. Currently uses the expectreg package.
scale a boolean value indicating whether X and Y should be normalized (to zero-mean and unit-variance) before learning.
sigest if the kernel expects a sigma parameter and none is provided in kpar, this parameter specifies a function to use to compute it.

Details
This function trains a new LinearizedSVR model based on X and Y. See LinearizedSVR-package for an explanation of how such models are defined.

Value
a model object that can later be used as the first argument for the predict() method.
See Also

LinearizedSVR-package

Examples

```r
dat <- rbind(data.frame(y=2, x1=rnorm(500, 1), x2=rnorm(500, 1)),
              data.frame(y=1, x1=rnorm(500,-1), x2=rnorm(500,-1)))
mod <- LinearizedSVRTrain(X=as.matrix(dat[-1]), Y=dat$y, numP=6)
res <- predict(mod, newdata=as.matrix(dat[-1]))
plot(x2 ~ x1, dat, col=c("red","green")[1+(res>1.5)], pch=c(3,20)[dat$y])
```

Description

Predict method for LinearizedSVR models

Usage

```r
## S3 method for class 'LinearizedSVR'
predict(object, newdata, ...)
```

Arguments

- **object**
  a model previously trained using `LinearizedSVRTrain()`
- **newdata**
  a matrix of new data to run predictions on, with the same columns as `X` had during training
- **...**
  further arguments passed to or from other methods

Details

This method produces predicted value, obtained by evaluating the trained model object on the given data set `newdata`. The columns of `newdata` must correspond exactly to the columns of `X` when the model object was created.

Value

A vector of predicted regression values, with length equal to the number of rows in `newdata`. 
sigma.est

Quickly estimates a reasonable default value for the 'sigma' parameter to several of kernlab's kernel functions.

Description

Quickly estimates a reasonable default value for the 'sigma' parameter to several of kernlab's kernel functions.

Usage

sigma.est(X, method = c("meddist", "invvar"), ...)

Arguments

X a numeric matrix
method a which procedure to use for estimation - 'meddist' means
... currently ignored
Index

LinearizedSVR-package, 2, 3
LinearizedSVRTrain, 3
predict.LinearizedSVR, 4
sigma.est, 5