

Package ‘biosensors.usc’

December 5, 2021

Encoding UTF-8

Type Package

Title biosensors.usc package

Version 1.0

Date 2020-05-08

Author Juan C. Vidal,
Marcos Matabuena

Maintainer Juan C. Vidal <juan.vidal@usc.es>

Description Biosensor.usc aims to provide a unified and user-friendly framework for using new distributional representations of biosensors data in different statistical modeling tasks: regression models, hypothesis testing, cluster analysis, visualization, and descriptive analysis. Distributional representations are a functional extension of compositional time-range metrics and we have used them successfully so far in modeling glucose profiles and accelerometer data. However, these functional representations can be used to represent any biosensor data such as ECG or medical imaging such as fMRI.

Imports Rcpp, graphics, stats, methods, utils, energy, fda.usc, parallelDist, osqp, truncnorm

Depends R (>= 2.15)

LinkingTo Rcpp, RcppArmadillo

LazyLoad Yes

License GPL-2

NeedsCompilation Yes

RoxygenNote 7.1.1

R topics documented:

biosensors.usc	2
clustering	2
clustering_prediction	3
cpp_confidence_band	3
cpp_wasserstein_regression	4
generate_data	5
hypothesis_testing	6
load_data	6
nadayara_prediction	7
nadayara_regression	8

regmod_prediction	9
regmod_regression	9
ridge_regression	10
wasserstein_prediction	11
wasserstein_regression	12

Index	13
--------------	-----------

biosensors.usc	<i>biosensors.usc Package</i>
----------------	-------------------------------

Description

Biosensor data have the potential to improve disease control and detection. However, the analysis of these data under free-living conditions is not feasible with current statistical techniques. To address this challenge, we introduce a new functional representation of biosensor data, termed the glucodensity, together with a data analysis framework based on distances between them. The new data analysis procedure is illustrated through an application in diabetes with continuous-time glucose monitoring (CGM) data. In this domain, we show marked improvement with respect to state-of-the-art analysis methods. In particular, our findings demonstrate that (i) the glucodensity possesses an extraordinary clinical sensitivity to capture the typical biomarkers used in the standard clinical practice in diabetes; (ii) previous biomarkers cannot accurately predict glucodensity, so that the latter is a richer source of information and; (iii) the glucodensity is a natural generalization of the time in range metric, this being the gold standard in the handling of CGM data. Furthermore, the new method overcomes many of the drawbacks of time in range metrics and provides more indepth insight into assessing glucose metabolism.

Author(s)

Juan C. Vidal <juan.vidal@usc.es>
Marcos Matabuena <marcos.matabuena@usc.es>

clustering	<i>clustering</i>
------------	-------------------

Description

Performs energy clustering with Wasserstein distance using quantile distributional representations as covariates.

Usage

```
clustering(data, clusters=3, iter_max=10, restarts=1)
```

Arguments

data	A biosensor object.
clusters	Number of clusters.
iter_max	Maximum number of iterations.
restarts	Number of restarts.

Value

An object of class `bclustering`: `data` A data frame with biosensor raw data. `result` A `kgroups` object (see `energy` library).

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS
# biology 16(7), 2018.
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")
data = load_data(file1, file2)
clus = clustering(data, clusters=3)
```

`clustering_prediction` *clustering_prediction*

Description

Predicts the cluster of each element of the objects list

Usage

```
clustering_prediction(clustering, objects)
```

Arguments

<code>clustering</code>	A <code>gl.clustering</code> object.
<code>objects</code>	Matrix of objects to cluster.

Value

The clusters to which these objects are assigned.

<code>cpp_confidence_band</code>	<i>This function computes intrinsic confidence bands for Wasserstein regression.</i>
----------------------------------	--

Description

This function computes intrinsic confidence bands for Wasserstein regression.

Usage

```
cpp_confidence_band(xfit, xpred, Q_obs, q_obs, t_vec, alpha)
```

Arguments

xfit	A nxp matrix of predictor values for fitting (do not include a column for the intercept).
xpred	A kxp matrix of input values for regressors for prediction.
Q_obs	A nxm matrix of quantile functions. Q_obs(i, :) is a 1xm vector of quantile function values on grid t_vec.
q_obs	A nxm matrix of quantile density functions. q_obs(i, :) is a 1xm vector of quantile density function values on grid t_vec.
t_vec	A 1xm vector - common grid for all quantile density functions in Q_obs, q_obs, q_prime_obs.
alpha	The significant level is $100 \times (1 - \alpha)$.

Value

An object containing the components: Q_lx Lower bound of confidence bands in terms of density functions. Q_ux Upper bound of confidence bands in terms of density functions. Qpred Fitted density function at xpred.

cpp_wasserstein_regression

This function perform Frechet regression with the Wasserstein distance.

Description

This function perform Frechet regression with the Wasserstein distance.

Usage

```
cpp_wasserstein_regression(xfit, q, Q0, xpred, t, qdmin)
```

Arguments

xfit	A nxp matrix of predictor values for fitting (do not include a column for the intercept).
q	A nxm matrix of quantile density functions. q(i, :) is a 1xm vector of quantile density function values on an equispaced grid on [0, 1].
Q0	A 1xn array of quantile function values at 0.
xpred	A kxp matrix of input values for regressors for prediction.
t	A 1xm vector - common grid for all quantile density functions in q. If missing, defaults to linspace(0, 1, m); For best results, should use a finer grid than for quantle estimation, especially near the boundaries.
qdmin	A positive lower bound on the estimated quantile densites. Defaults to 1e-6.

Value

An object containing the components: `xpred` See input of same name. `qpred` A $k \times N$ array. `qpred(l, :)` is the regression prediction of q (the quantile density) given $X = \text{xpred}(l, :)$. `Qpred` A $k \times m$ array. `Qpred(l, :)` is the regression prediction of Q given $X = \text{xpred}(l, :)$. `fpred` A $k \times m$ array. `fpred(l, :)` is the regression prediction of f (the density) given $X = \text{xpred}(l, :)$, evaluated on the grid `Qpred(l, :)`. `xfit` See input of same name. `qfit` A $n \times N$ array. `qfit(l, :)` is the regression prediction of q given $X = \text{xfit}(l, :)$. `Qfit` A $n \times m$ array. `Qfit(l, :)` is the regression prediction of Q given $X = \text{xfit}(l, :)$. `fpred` A $k \times m$ array. `fpred(l, :)` is the regression prediction of f (the density) given $X = \text{xpred}(l, :)$, evaluated on the grid `Qfit(l, :)`. `QP_used` A flag indicating whether OLS fits all satisfied the constraints ($=0$) or if the quadratic program was used in fitting ($=1$).

`generate_data`

generate_data

Description

Generates a quantile regression model $V + V2 * v + \tau * V2 * Q0$ where $Q0$ is a truncated random variable, $v = 2 * X$, $\tau = 2 * X$, $V \sim \text{Unif}(-1, 1)$, $V2 \sim \text{Unif}(-1, -1)$, $V3 \sim \text{Unif}(0.8, 1.2)$, and $E(VIX) = \tau * Q0$;

Usage

```
generate_data(n = 100, Qp = 100, Xp = 5)
```

Arguments

<code>n</code>	Sample size.
<code>Qp</code>	Dimension of the quantile.
<code>Xp</code>	Dimension of covariates where $X_i \sim \text{Unif}(0,1)$.

Value

A biosensor object: `data` NULL. `densities` NULL. `quantiles` A functional data object (`fdata`) with the empirical quantile estimation. `variables` A data frame with `Xp` covariates.

Examples

```
data = generate_data(n=100, Qp=100, Xp=5)
names(data)
head(data$quantiles)
head(data$variables)
plot(data$quantiles, main="Quantile curves")
```

hypothesis_testing	<i>hypothesis_testing</i>
--------------------	---------------------------

Description

Hypothesis testing between two random samples of distributional representations to detect differences in scale and localization (ANOVA test) or distributional differences (Energy distance).

Usage

```
hypothesis_testing(data1, data2, permutations=100)
```

Arguments

data1	A biosensor object. First population.
data2	A biosensor object. Second population.
permutations	Number of permutations used in the energy distance calibration test.

Value

An object of class `biotest`: `p1_mean` Quantile mean of the first population. `p1_variance` Quantile variance of the first population. `p2_mean` Quantile mean of the second population. `p2_variance` Quantile variance of the second population. `energy_pvalue` P-value of the energy distance test. `anova_pvalue` P-value of the ANOVA-Fréchet test.

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS
# biology 16(7), 2018.
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")
data1 = load_data(file1, file2)
file3 = system.file("extdata", "data_2.csv", package = "biosensors.usc")
file4 = system.file("extdata", "variables_2.csv", package = "biosensors.usc")
data2 = load_data(file3, file4)
htest = hypothesis_testing(data1, data2)
```

load_data	<i>load_data</i>
-----------	------------------

Description

R function to read biosensors data from a csv files.

Usage

```
load_data(filename_fdata, filename_variables = NULL)
```

Arguments

`filename_fdata` A csv file with the functional data. The csv file must have long format with, at least, the following three columns: id, time, and value, where the id identifies the individual, the time indicates the moment in which the data was captured, and the value is a monitor measure.

`filename_variables` A csv file with the clinical variables. The csv file contains a row per individual and must have a column id identifying this individual.

Value

A biosensor object: `data` A data frame with biosensor raw data. `densities` A functional data object (fdata) with a non-parametric density estimation. `quantiles` A functional data object (fdata) with the empirical quantile estimation. `variables` A data frame with the covariates.

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS
# biology 16(7), 2018.
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")
data = load_data(file1, file2)
names(data)
head(data$quantiles)
head(data$variables)
plot(data$quantiles, main="Quantile curves")
```

nadayara_prediction	<i>nadayara_prediction</i>
---------------------	----------------------------

Description

Functional non-parametric Nadaraya-Watson prediction with 2-Wasserstein distance.

Usage

```
nadayara_prediction(nadaraya, Qpred, hs=NULL)
```

Arguments

`nadaraya` A Nadaraya regression object.

`Qpred` Quantile curves that will be used in the predictions

`hs` Smoothing parameters for the predictions, by default `hs = seq(0.8, 15, length = 200)`

Value

An object of class `bnadarayapred`: `prediction` The Nadaraya-Watson prediction for the test data at each value of `hs`. `hs` `Hs` values used for the prediction.

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS
# biology 16(7), 2018.
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")
data = load_data(file1, file2)
nada = nadayara_regression(data, "BMI")
# Example of prediction with the column mean of quantiles
npred = nadayara_prediction(nada, t(colMeans(data$quantiles$data)))
```

nadayara_regression	<i>nadayara_regression</i>
---------------------	----------------------------

Description

Functional non-parametric Nadaraya-Watson regression with 2-Wasserstein distance, using as predictor the distributional representation and as response a scalar outcome.

Usage

```
nadayara_regression(data, response)
```

Arguments

data	A biosensor object.
response	The name of the scalar response. The response must be a column name in data\$variables.

Value

An object of class bnadaraya: prediction The Nadaraya-Watson prediction for each point of the training data at each $h = \text{seq}(0.8, 15, \text{length}=200)$. r2 R2 estimation for the training data at each $h = \text{seq}(0.8, 15, \text{length}=200)$. error Standard mean-squared error after applying leave-one-out cross-validation for the training data at each $h = \text{seq}(0.8, 15, \text{length}=200)$. data A data frame with biosensor raw data. response The name of the scalar response.

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS
# biology 16(7), 2018.
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")
data = load_data(file1, file2)
nada = nadayara_regression(data, "BMI")
```

regmod_prediction	<i>regmod_prediction</i>
-------------------	--------------------------

Description

Performs the Wasserstein regression using quantile functions.

Usage

```
regmod_prediction(data, xpred)
```

Arguments

data	A bregmod object.
xpred	A kxp matrix of input values for regressors for prediction, where k is the number of points we do the prediction and p is the dimension of the input variables.

Value

A kxm array. Qpred(l, :) is the regression prediction of Q given $X = \text{xpred}(l, :)$ where m is the dimension of the grid of quantile function.

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS
# biology 16(7), 2018.
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")
data = load_data(file1, file2)
regm = regmod_regression(data, "BMI")
# Example of prediction
xpred = as.matrix(25)
glrmp = regmod_prediction(regm, xpred)
```

regmod_regression	<i>regmod_regression</i>
-------------------	--------------------------

Description

Performs the Wasserstein regression using quantile functions.

Usage

```
regmod_regression(data, response)
```

Arguments

data	A biosensor object.
response	The name of the scalar response. The response must be a column name in data\$variables.

Value

An object of class `bregmod` containing the components: `beta` The beta coefficient functions of the fitting. `prediction` The prediction for each training data. `residuals` The residuals for each prediction value.

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS
# biology 16(7), 2018.
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")
data = load_data(file1, file2)
regm = regmod_regression(data, "BMI")
```

<code>ridge_regression</code>	<i>ridge_regression</i>
-------------------------------	-------------------------

Description

Performs a Ridge regression.

Usage

```
ridge_regression(data, response, w=NULL, method="manhattan", type="gaussian")
```

Arguments

<code>data</code>	A biosensor object.
<code>response</code>	The name of the scalar response. The response must be a column name in <code>data\$variables</code> .
<code>w</code>	A weight function.
<code>method</code>	The distance measure to be used (@seealso <code>parallelDist::parDist</code>). By default <code>manhattan</code> distance.
<code>type</code>	The kernel type ("gaussian" or "lapla"). By default <code>gaussian</code> distance.

Value

An object containing the components: `best_alphas` Best coefficients obtained with leave-one-out cross-validation criteria. `best_kernel` The kernel matrix of the best solution. `best_sigma` The sigma parameter of the best solution. `best_lambda` The lambda parameter of the best solution. `sigmas` The sigma parameters used in the fitting according to the median heuristic fitting criteria. `predictions` A matrix of predictions. `r2` R-square of the different models fitted. `error` Mean squared-error of the different models fitted. `predictions_cross` A matrix of predictions obtained with leave-one-out cross-validation criteria.

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS
# biology 16(7), 2018.
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")
data = load_data(file1, file2)
regm = ridge_regression(data, "BMI")
```

```
wasserstein_prediction
      wasserstein_prediction
```

Description

Performs the Wasserstein prediction.

Usage

```
wasserstein_prediction(reg, xpred)
```

Arguments

reg	A bwasserstein object.
xpred	A kxp matrix of input values for regressors for prediction, where k is the number of points we do the prediction and p is the dimension of the input variables.

Value

A kxm array. Qpred(l, :) is the regression prediction of Q given $X = \text{xpred}(l, :)$ where m is the dimension of the grid of quantile function.

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS
# biology 16(7), 2018.
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")
data = load_data(file1, file2)
wass = wasserstein_regression(data, "BMI")
# Example of prediction
xpred = as.matrix(25)
pred = wasserstein_prediction(wass, xpred)
```

`wasserstein_regression`*wasserstein_regression*

Description

Performs the Wasserstein regression using a quantile density function.

Usage

```
wasserstein_regression(data, response)
```

Arguments

<code>data</code>	A biosensor object.
<code>response</code>	The name of the scalar response. The response must be a column name in <code>data\$variables</code> .

Value

An object of class `wasserstein` containing the components: `prediction` The fitted regression.
`regression` An internal `bwasserstein` object (@seealso `cpp_wasserstein_regression`)
`data` A data frame with biosensor raw data.
`response` The name of the scalar response.

Examples

```
# Data extracted from the paper: Hall, H., Perelman, D., Breschi, A., Limcaoco, P., Kellogg, R.,  
# McLaughlin, T., Snyder, M., Glucotypes reveal new patterns of glucose dysregulation, PLoS  
# biology 16(7), 2018.  
file1 = system.file("extdata", "data_1.csv", package = "biosensors.usc")  
file2 = system.file("extdata", "variables_1.csv", package = "biosensors.usc")  
data = load_data(file1, file2)  
wass = wasserstein_regression(data, "BMI")
```

Index

biosensors.usc, [2](#)

clustering, [2](#)
clustering_prediction, [3](#)
cpp_confidence_band, [3](#)
cpp_wasserstein_regression, [4](#)

generate_data, [5](#)

hypothesis_testing, [6](#)

load_data, [6](#)

nadayara_prediction, [7](#)
nadayara_regression, [8](#)

regmod_prediction, [9](#)
regmod_regression, [9](#)
ridge_regression, [10](#)

wasserstein_prediction, [11](#)
wasserstein_regression, [12](#)