Package ‘GTbasedIM’

January 20, 2025

Title Game Theory-Based Influence Measures

Version 1.0.0

Description
Understanding how features influence a specific response variable becomes crucial in classification problems, with applications ranging from medical diagnosis to customer behavior analysis. This packages provides tools to compute such an influence measure grounded on game theory concepts.
In particular, the influence measures presented in Davila-Pena, Saavedra-Nieves, and Casas-Méndez (2024) <doi:10.48550/arXiv.2408.02481> can be obtained.

Date 2024-10-02

License GPL-2

Encoding UTF-8

RoxygenNote 7.3.1

Depends R (>= 2.10)

Imports CoopGame, RWeka, stats

NeedsCompilation no

Author Laura Davila-Pena [aut, cre] (<`https://orcid.org/0000-0003-2175-2546`>)

Maintainer Laura Davila-Pena <`lauradavila.pena@usc.es`>

Repository CRAN

Date/Publication 2024-10-07 09:30:09 UTC

Contents

`cars_binary` *Cars Binary Dataset*

Description

This dataset contains data on car crash fatalities after transforming the original categorical variables of the dataset `cars_original` into binary variables. Details about such transformations are available in Davila-Pena et al. (2024). (Section 6.1, Table 5).

Usage

```
data(cars_binary)
```

Format

A dataframe with 17565 rows (representing individuals) and 12 columns, one corresponding to the index number (first column, `X`), one to the binary response (last column, `deceased`), and the remaining 10 representing the following binary features:

`dvcat` Binary variable indicating whether the vehicle, at the moment of the accident, was traveling at a speed higher than 55 km/h (1) or not (0).

`airbag` Binary variable indicating whether the vehicle had an airbag system (1) or not (0).

`seatbelt` Binary variable indicating whether the person involved was wearing a seat belt (1) or not (0).

`frontal` Binary variable indicating whether the vehicle crash was frontal (1) or nonfrontal (0).

`sex` Binary variable indicating the sex of the person involved: 1 for male and 0 for female.

`abcat` Binary variable indicating airbag activation: 1 if one or more airbags in the vehicle were activated (even if not deployed) and 0 if none were deployed (either due to malfunction or being disabled).

`occRole` Binary variable indicating whether the person involved was the driver (1) or a passenger (0) of the vehicle.

`deploy` Binary variable indicating whether the airbag functioned correctly (1) or was unavailable or not functioning (0).

`ageOFocc` Binary variable indicating whether the person was 30 years old or less (1) or over 30 years old (0).

`age` Binary variable indicating whether the vehicle was 10 years old or more (1) or less than 10 years old (0).

Source

Generated data for illustrative purposes, see: Davila-Pena, L., Saavedra-Nieves, A., & Casas-Méndez, B. (2024). *On the influence of dependent features in classification problems: a game-theoretic perspective*. arXiv preprint. doi:10.48550/arXiv.2408.02481.

Examples

```
data(cars_binary)
head(cars_binary)
```

`cars_original` *Cars Original Dataset*

Description

This dataset contains data on car crash fatalities.

Usage

```
data(cars_original)
```

Format

A dataframe with 17565 rows (representing individuals) and 16 columns, one corresponding to the index number (first column, `X`) and the remaining ones corresponding to different variables. For details, please refer to the source below.

Source

This dataset contains data on car crash fatalities from the nassCDS dataset via the "DAAG" R package (doi:10.32614/CRAN.package.DAAG) from Maindonald & Braun (2021).

References

Maindonald, J. H. & Braun W. J. (2021). *Data analysis and graphics using R. An example-based approach (3rd edition)*. Cambridge University Press, Cambridge. The DAAG package was created to support this text. `https://CRAN.R-project.org/package=DAAG` (R package version 1.25.6.)

Examples

```
data(cars_original)
head(cars_original)
```

checking — *checking Function: Check Specified Rows Within Dataset*

Description

The `checking` function checks whether a given pair `(Xdata.i.ell, Yell.i)` is present in the dataset `(Xdata, Ydata)` and returns the corresponding indices, if any.

Usage

```
checking(Xdata, Ydata, Xdata.i.ell, Yell.i)
```

Arguments

`Xdata`	Matrix. A dataset where rows represent observations and columns represent features.
`Ydata`	Vector. The response variable associated with each row in `Xdata`.
`Xdata.i.ell`	Vector. A vector of feature values, potentially representing a row of `Xdata`.
`Yell.i`	Integer. The specific response value to check.

Value

A vector of indices where the match occurs.

Examples

```
# Example usage from Example 5.2 in Davila-Pena et al. (2024):

library(CoopGame)
n.user <- 16
Xdata <- createBitMatrix(4)[,-5]
Xdata <- rbind(c(0,0,0,0),Xdata)
Ydata <- rep(0,n.user)
Ydata[1+c(10,11,13,14,15)] <- 1

Xdata.i.ell <- c(1,2,0,1) # obviously, considering `Xdata` is binary, this cannot be present.
Yell.i <- 1

checking(Xdata, Ydata, Xdata.i.ell, Yell.i)

Xdata.i.ell <- c(1,1,0,1)
Yell.i <- 0

checking(Xdata, Ydata, Xdata.i.ell, Yell.i)
```

`IM_dep`	*IM_dep Function: Calculate Influence Measure with Dependent Features*

Description

The `IM_dep` function calculates the influence measure of equation (2) in Davila-Pena et al. (2024). The features are grouped into unions based on the `index` vector.

Usage

```
IM_dep(Xdata, Ydata, index)
```

Arguments

`Xdata`	Matrix. A dataset where rows represent observations and columns represent features.
`Ydata`	Vector. The response variable associated with each row in `Xdata`.
`index`	Vector. A grouping vector that assigns each feature in `Xdata` to a specific union. E.g., if we have the partition set P={{1},{2,4},{3}}, then index=c(1,2,3,2).

Details

The `IM_dep` function calculates the weighted average of the number of times a change in the value of a feature associated to a specific union influences the response value.

Value

A vector of influences for each feature.

References

Davila-Pena, L., Saavedra-Nieves, A., & Casas-Méndez, B. (2024). *On the influence of dependent features in classification problems: a game-theoretic perspective*. arXiv preprint. doi:10.48550/arXiv.2408.02481.

Examples

```
# Example usage from Example 5.2 in Davila-Pena et al. (2024):

library(CoopGame)
n.user <- 16
Xdata <- createBitMatrix(4)[,-5]
Xdata <- rbind(c(0,0,0,0),Xdata)
Ydata <- rep(0,n.user)
Ydata[1+c(10,11,13,14,15)] <- 1

# Scenario 1:
```

```
IM_dep(Xdata,Ydata,index = c(1,2,3,4))
# Scenario 9:
IM_dep(Xdata,Ydata,index = c(1,2,1,2))
```

IM_nodep *IM_nodep Function: Calculate Influence Measure for Features Without Dependency*

Description

The IM_nodep function calculates the influence measure of equation (2) in Davila-Pena et al. (2024) when the partition set is P={{1},{2},{3},{4}}, which is equivalent to the influence measure in Datta et al. (2015).

Usage

```
IM_nodep(Xdata, Ydata)
```

Arguments

Xdata	Matrix. A dataset where rows represent observations and columns represent features.
Ydata	Vector. The response variable associated with each row in Xdata.

Details

The IM_nodep function calculates the weighted average of the number of times a change in the value of a feature influences the response value.

Value

A vector of influences for each feature.

References

Datta, A., Datta, A., Procaccia, A., & Zick, Y. (2015). *Influence in classification via cooperative game theory*. Proceedings of the Twenty–fourth International Joint Conference on Artificial Intelligence, 511–517. https://www.ijcai.org/Proceedings/15/Papers/078.pdf.

Davila-Pena, L., Saavedra-Nieves, A., & Casas-Méndez, B. (2024). *On the influence of dependent features in classification problems: a game-theoretic perspective*. arXiv preprint. doi:10.48550/arXiv.2408.02481.

Examples

```
# Example usage from Example 5.2 in Davila-Pena et al. (2024):

library(CoopGame)
n.user <- 16
Xdata <- createBitMatrix(4)[,-5]
Xdata <- rbind(c(0,0,0,0),Xdata)
Ydata <- rep(0,n.user)
Ydata[1+c(10,11,13,14,15)] <- 1

# Scenario 1:
IM_nodep(Xdata,Ydata)
```

`Mell` *Mell Function: Subset Data Based on Group Index Conditions*

Description

The `Mell` function computes equation (1) in page 9 of Davila-Pena et al. (2024) when the dependency is positive and features are binary. It filters rows from `Xdata` and `Ydata` based on specific conditions of features' dependency as defined by `index`. It selects rows where the values of features within the same union, except for those in union `ell`, coincide.

Usage

```
Mell(Xdata, Ydata, index, ell)
```

Arguments

`Xdata`	Matrix. A dataset where rows represent observations and columns represent features.
`Ydata`	Vector. The response variable associated with each row in `Xdata`.
`index`	Vector. A grouping vector that assigns each feature in `Xdata` to a specific union. E.g., if we have the partition set P={{1},{2,4},{3}}, then index=c(1,2,3,2).
`ell`	Integer. The index of the union to exclude from the comparison.

Details

The `Mell` function iterates through each row of `Xdata` and compares the values of features within predefined unions (determined by `index`). For each union, excluding the one specified by `ell`, the function checks if all feature values in that union coincide. If this condition is satisfied for all unions except `ell`, the row is selected.

Value

A list containing the following components:

Xdata.ell A subset of `Xdata` that meets the specified group conditions.

Ydata.ell The corresponding values from `Ydata` for the selected rows in `Xdata.ell`.

n.user.ell The number of selected rows that meet the group condition.

References

Davila-Pena, L., Saavedra-Nieves, A., & Casas-Méndez, B. (2024). *On the influence of dependent features in classification problems: a game-theoretic perspective*. arXiv preprint. doi:10.48550/arXiv.2408.02481.

Examples

```
# Example usage from Example 5.2 in Davila-Pena et al. (2024):

library(CoopGame)
n.user <- 16
Xdata <- createBitMatrix(4)[,-5]
Xdata <- rbind(c(0,0,0,0),Xdata)
Ydata <- rep(0,n.user)
Ydata[1+c(10,11,13,14,15)] <- 1

# Scenario 1:
Mell(Xdata = Xdata, Ydata = Ydata, index = c(1,2,3,4), ell = 3)
# Scenario 9:
Mell(Xdata = Xdata, Ydata = Ydata, index = c(1,2,1,2), ell = 2)
```

`predictions` *Predictions Function: Generate predictions using classifiers from RWeka*

Description

This function trains three different classifiers—Random Forest (RF), Support Vector Machine (SVM), and Logistic Regression (LR)—on the input dataset and returns their predictions.

Usage

```
predictions(Xdata, Ydata)
```

Arguments

`Xdata`	Matrix. A dataset where rows represent observations and columns represent features.
`Ydata`	Vector. The actual response variable associated with each row in `Xdata`.

Details

The function utilizes the RWeka package to build three different classifiers: Random Forest (RF), Support Vector Machine (SVM), and Logistic Regression (LR). It then predicts the response variable for the input data using each of these models.

Value

A list containing predictions from the three models:

`predictionRF`	Predictions from the Random Forest (RF) model.
`predictionSVM`	Predictions from the Support Vector Machine (SVM) model.
`predictionLR`	Predictions from the Logistic Regression (LR) model.

References

Davila-Pena, L., Saavedra-Nieves, A., & Casas-Méndez, B. (2024). *On the influence of dependent features in classification problems: a game-theoretic perspective*. arXiv preprint. doi:10.48550/arXiv.2408.02481.

Examples

```
# Example usage:

X <- matrix(rnorm(100), ncol=5)
Y <- sample(0:1, 20, replace=TRUE)
predictions(X, Y)
```

`spotify_15` *Spotify 15 Dataset*

Description

This dataset contains data on musical tastes in Spotify considering only the 15 most listened-to artists from the dataset `spotify_original`. Details about such artists are available in Davila-Pena et al. (2024) (Section 6.2, Table 11).

Usage

```
data(spotify_15)
```

Format

A dataframe with 1226 rows (users) and 16 columns, one corresponding to the index number (first column, `X`) and the remaining 15 corresponding to different artists.

Source

Item-item collaborative filtering with binary or unitary data, https://medium.com/radon-dev/item-item-collaborative-filtering-with-binary-or-unary-data-e8f0b465b2c3. Accessed: 02 October 2024

Examples

```
data(spotify_15)
head(spotify_15)
```

`spotify_75` *Spotify 75 Dataset*

Description

This dataset contains data on musical tastes in Spotify considering only the 75 most listened-to artists from the dataset `spotify_original`. Details about such artists are available in Davila-Pena et al. (2024) (Section 6.2, Table 11).

Usage

```
data(spotify_75)
```

Format

A dataframe with 1226 rows (users) and 76 columns, one corresponding to the index number (first column, `X`) and the remaining 75 corresponding to different artists.

Source

Item-item collaborative filtering with binary or unitary data, https://medium.com/radon-dev/item-item-collaborative-filtering-with-binary-or-unary-data-e8f0b465b2c3. Accessed: 02 October 2024

Examples

```
data(spotify_75)
head(spotify_75)
```

`spotify_original` *Spotify Original Dataset*

Description

This dataset contains data on musical tastes in Spotify from the `last.fm` dataset. It informs about which users have listened to which artists. Refer to the source for details.

Usage

```
data(spotify_original)
```

Format

A dataframe with 1226 rows (users) and 286 columns, one corresponding to the index number (first column, `X`) and the remaining 285 corresponding to different artists.

Source

Item-item collaborative filtering with binary or unitary data, `https://medium.com/radon-dev/item-item-collaborative-filtering-with-binary-or-unary-data-e8f0b465b2c3`. Accessed: 02 October 2024

Examples

```
data(spotify_original)
head(spotify_original)
```

Index