Package ‘FuzzyClass’

February 14, 2024

Title  Fuzzy and Non-Fuzzy Classifiers
Version 0.1.6
Description  It provides classifiers which can be used for discrete variables and for continuous variables based on the Naive Bayes and Fuzzy Naive Bayes hypothesis. Those methods were developed by researchers belong to the 'Laboratory of Technologies for Virtual Teaching and Statistics (LabTEVE)' and 'Laboratory of Applied Statistics to Image Processing and Geoprocessing (LEAPIG)' at 'Federal University of Paraiba, Brazil'. They considered some statistical distributions and their papers were published in the scientific literature, as for instance, the Gaussian classifier using fuzzy parameters, proposed by 'Moraes, Ferreira and Machado' (2021) <doi:10.1007/s40815-020-00936-4>.

License  MIT + file LICENSE
Encoding UTF-8
LazyData true
RoxygenNote 7.2.3
Imports caTools, doParallel, e1071, EnvStats, foreach, MASS, mvtnorm, parallel, purrr, Rdpack, rootSolve, stats, trapezoid
RdMacros Rdpack
Depends R (>= 2.10)
Suggests testthat (>= 3.0.0), knitr, rmarkdown, maxLik, dplyr
VignetteBuilder knitr
Config/testthat/edition 3
URL  https://github.com/leapigufpb/FuzzyClass
BugReports  https://github.com/leapigufpb/FuzzyClass/issues
Language en-US
NeedsCompilation no
Author Jodavid Ferreira [aut, cre] (<https://orcid.org/0000-0002-2131-6464>), Ronei Moraes [ctb] (<https://orcid.org/0000-0001-8436-8950>), Arthur Ricardo [ctb]
Maintainer Jodavid Ferreira <jodavid@protonmail.com>
Repository CRAN
Date/Publication 2024-02-14 17:50:02 UTC
R topics documented:

DWFuzzyGammaNaiveBayes ........................................ 2
ExpNBfuzzyParam ........................................ 3
FuzzyBayesRule ........................................ 5
FuzzyBetaNaiveBayes ........................................ 6
FuzzyBinomialNaiveBayes ....................................... 7
FuzzyExponentialNaiveBayes .................................... 8
FuzzyGammaNaiveBayes .......................................... 9
FuzzyGaussianNaiveBayes ....................................... 10
FuzzyGeoNaiveBayes .......................................... 11
FuzzyNaiveBayes ........................................ 13
FuzzyPoissonNaiveBayes ....................................... 14
FuzzyTrapezoidalNaiveBayes ................................... 16
FuzzyTriangularNaiveBayes .................................... 17
GamWeightData ........................................ 18
GauNBfuzzyParam .......................................... 18
HouseVotes84 ........................................ 20
PoiNBfuzzyParam .......................................... 21
SimulatedData ........................................ 22
VirtualRealityData ....................................... 23

Index 24

DWFuzzyGammaNaiveBayes

Double Weighted Fuzzy Gamma Naive Bayes

Description

DWFuzzyGammaNaiveBayes Double Weighted Fuzzy Gamma Naive Bayes

Usage

DWFuzzyGammaNaiveBayes(train, cl, cores = 2, fuzzy = TRUE, wdelta, weta)

Arguments

train matrix or data frame of training set cases.
cl factor of true classifications of training set
cores how many cores of the computer do you want to use to use for prediction (default = 2)
fuzzy boolean variable to use the membership function
wdelta vector weight each class
weta vector weight each feature
ExpNBfuzzyParam

Value

A vector of classifications

References


Examples

```r
set.seed(1) # determining a seed
data(GamWeightData)

# Splitting into Training and Testing
split <- caTools::sample.split(t(GamWeightData[, 1]), SplitRatio = 0.7)
Train <- subset(GamWeightData, split == "TRUE")
Test <- subset(GamWeightData, split == "FALSE")
# ----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -4]
fit_NBT <- DWFuzzyGammaNaiveBayes(
  train = Train[, -4],
  cl = Train[, 4], cores = 2,
  wdelta = c(2.002/6, 1.998/6, 2.000/6),
  weta = c(3/10, 2/10, 5/10)
)
pred_NBT <- predict(fit_NBT, test)

head(pred_NBT)
head(Test[, 4])
```

ExpNBfuzzyParam      Fuzzy Exponential Naive Bayes Classifier with Fuzzy parameters

Description

ExpNBfuzzyParam Fuzzy Exponential Naive Bayes Classifier with Fuzzy parameters

Usage

ExpNBfuzzyParam(train, cl, alphacut = 1e-04, metd = 2, cores = 2)
**ExpNBFuzzyParam**

**Arguments**

- **train**: matrix or data frame of training set cases
- **cl**: factor of true classifications of training set
- **alphacut**: value of the alpha-cut parameter, this value is between 0 and 1.
- **metd**: Method of transforming the triangle into scalar. It is the type of data entry for the test sample, use **metd 1** if you want to use the Yager technique, **metd 2** if you want to use the Q technique of the uniformity test (article: Directional Statistics and Shape analysis), and **metd 3** if you want to use the Thorani technique
- **cores**: how many cores of the computer do you want to use to use for prediction (default = 2)

**Value**

A vector of classifications

**References**


**Examples**

```r
set.seed(1) # determining a seed
data(VirtualRealityData)

# Splitting into Training and Testing
split <- caTools::sample.split(t(VirtualRealityData[, 1]), SplitRatio = 0.7)
Train <- subset(VirtualRealityData, split == "TRUE")
Test <- subset(VirtualRealityData, split == "FALSE")

# ----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -4]
fit_FENB <- ExpNBFuzzyParam(
    train = Train[, -4],
    cl = Train[, 4], metd = 1, cores = 2
)

pred_FENB <- predict(fit_FENB, test)
head(pred_FENB)
head(Test[, 4])
```
FuzzyBayesRule

Description

FuzzyBayesRule Fuzzy Bayes Rule

Usage

FuzzyBayesRule(train, cl, cores = 2, fuzzy = TRUE)

Arguments

- **train**: matrix or data frame of training set cases.
- **cl**: factor of true classifications of training set.
- **cores**: how many cores of the computer do you want to use to use for prediction (default = 2)
- **fuzzy**: boolean variable to use the membership function

Value

A vector of classifications

References


Examples

```r
set.seed(1) # determining a seed
data(iris)

# Splitting into Training and Testing
split <- caTools::sample.split(t(iris[, 1]), SplitRatio = 0.7)
Train <- subset(iris, split == "TRUE")
Test <- subset(iris, split == "FALSE")
# ----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -5]
fit_NBT <- FuzzyBayesRule(
  train = Train[, -5],
  cl = Train[, 5], cores = 2
)
pred_NBT <- predict(fit_NBT, test)
```
**FuzzyBetaNaiveBayes**

**Fuzzy Beta Naive Bayes**

**Description**

FuzzyBetaNaiveBayes Fuzzy Beta Naive Bayes

**Usage**

FuzzyBetaNaiveBayes(train, cl, cores = 2, fuzzy = TRUE)

**Arguments**

- **train**: matrix or data frame of training set cases.
- **cl**: factor of true classifications of training set
- **cores**: how many cores of the computer do you want to use to use for prediction (default = 2)
- **fuzzy**: boolean variable to use the membership function

**Value**

A vector of classifications

**References**


**Examples**

```r
set.seed(1) # determining a seed
data(iris)

# Splitting into Training and Testing
split <- caTools::sample.split(t(iris[, 1]), SplitRatio = 0.7)
Train <- subset(iris, split == "TRUE")
Test <- subset(iris, split == "FALSE")
#----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -5]
fit_NBT <- FuzzyBetaNaiveBayes(
```
FuzzyBinomialNaiveBayes

Fuzzy Binomial Naive Bayes

Description

FuzzyBinomialNaiveBayes Fuzzy Binomial Naive Bayes

Usage

FuzzyBinomialNaiveBayes(train, cl, cores = 2, fuzzy = TRUE)

Arguments

- **train**: matrix or data frame of training set cases.
- **cl**: factor of true classifications of training set
- **cores**: how many cores of the computer do you want to use to use for prediction (default = 2)
- **fuzzy**: boolean variable to use the membership function

Value

A vector of classifications

References


Examples

```r
set.seed(1) # determining a seed
class1 <- data.frame(var1 = rbinom(100, size = 10, prob = 0.2),
                     var2 = rbinom(100, size = 10, prob = 0.2),
                     var3 = rbinom(100, size = 10, prob = 0.2), class = 1)
class2 <- data.frame(var1 = rbinom(100, size = 10, prob = 0.5),
                     var2 = rbinom(100, size = 10, prob = 0.5),
                     var3 = rbinom(100, size = 10, prob = 0.5), class = 2)
```
FuzzyExponentialNaiveBayes

Fuzzy Exponential Naive Bayes

Description
FuzzyExponentialNaiveBayes Fuzzy Exponential Naive Bayes

Usage
FuzzyExponentialNaiveBayes(train, cl, cores = 2, fuzzy = TRUE)

Arguments

train matrix or data frame of training set cases.
cl factor of true classifications of training set
cores how many cores of the computer do you want to use to use for prediction (default = 2)
fuzzy boolean variable to use the membership function

Value
A vector of classifications
References


Examples

```r
set.seed(1)  # determining a seed
data(iris)

# Splitting into Training and Testing
split <- caTools::sample.split(t(iris[, 1]), SplitRatio = 0.7)
Train <- subset(iris, split == "TRUE")
Test <- subset(iris, split == "FALSE")

# ----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -5]
fit_NBT <- FuzzyExponentialNaiveBayes(
  train = Train[, -5],
  cl = Train[, 5], cores = 2
)

pred_NBT <- predict(fit_NBT, test)

head(pred_NBT)
head(Test[, 5])
```

FuzzyGammaNaiveBayes  

**Fuzzy Gamma Naive Bayes**

Description

FuzzyGammaNaiveBayes Fuzzy Gamma Naive Bayes

Usage

`FuzzyGammaNaiveBayes(train, cl, cores = 2, fuzzy = TRUE)`

Arguments

- `train`: matrix or data frame of training set cases.
- `cl`: factor of true classifications of training set
- `cores`: how many cores of the computer do you want to use to use for prediction (default = 2)
- `fuzzy`: boolean variable to use the membership function
Value

A vector of classifications

References


Examples

```r
set.seed(1) # determining a seed
data(iris)

# Splitting into Training and Testing
split <- caTools::sample.split(t(iris[, 1]), SplitRatio = 0.7)
Train <- subset(iris, split == "TRUE")
Test <- subset(iris, split == "FALSE")
# ----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -5]
fit_NBT <- FuzzyGammaNaiveBayes(
  train = Train[, -5],
  cl = Train[, 5], cores = 2
)
pred_NBT <- predict(fit_NBT, test)

head(pred_NBT)
head(Test[, 5])
```

---

**FuzzyGaussianNaiveBayes**

*Fuzzy Gaussian Naive Bayes Classifier Zadeh-based*

**Description**

FuzzyGaussianNaiveBayes Fuzzy Gaussian Naive Bayes Classifier Zadeh-based

**Usage**

FuzzyGaussianNaiveBayes(train, cl, cores = 2, fuzzy = TRUE)
Arguments

train       matrix or data frame of training set cases.
cl          factor of true classifications of training set
cores       how many cores of the computer do you want to use to use for prediction (default = 2)
fuzzy       boolean variable to use the membership function

Value

A vector of classifications

References


Examples

set.seed(1) # determining a seed
data(iris)

# Splitting into Training and Testing
split <- caTools::sample.split(t(iris[, 1]), SplitRatio = 0.7)
Train <- subset(iris, split == "TRUE")
Test <- subset(iris, split == "FALSE")

# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -5]
fit_GNB <- FuzzyGaussianNaiveBayes(
    train = Train[, -5],
    cl = Train[, 5], cores = 2
)

pred_GNB <- predict(fit_GNB, test)

head(pred_GNB)
head(Test[, 5])

FuzzyGeoNaiveBayes       Fuzzy Naive Bayes Geometric Classifier

Description

FuzzyGeoNaiveBayes Naive Bayes Geometric Classifier
**Usage**

FuzzyGeoNaiveBayes(train, cl, cores = 2, fuzzy = T)

**Arguments**

- `train` : matrix or data frame of training set cases.
- `cl` : factor of true classifications of training set.
- `cores` : how many cores of the computer do you want to use (default = 2).
- `fuzzy` : boolean variable to use the membership function.

**Value**

A vector of classifications.

**References**


**Examples**

```r
set.seed(1) # determining a seed
class1 <- data.frame(vari1 = rgeom(100,prob = 0.2),
                     vari2 = rgeom(100,prob = 0.2),
                     vari3 = rgeom(100,prob = 0.2), class = 1)
class2 <- data.frame(vari1 = rgeom(100,prob = 0.5),
                     vari2 = rgeom(100,prob = 0.5),
                     vari3 = rgeom(100,prob = 0.5), class = 2)
class3 <- data.frame(vari1 = rgeom(100,prob = 0.9),
                     vari2 = rgeom(100,prob = 0.9),
                     vari3 = rgeom(100,prob = 0.9), class = 3)
data <- rbind(class1,class2,class3)

# Splitting into Training and Testing
split <- caTools::sample.split(t(data[, 1]), SplitRatio = 0.7)
Train <- subset(data, split == "TRUE")
Test <- subset(data, split == "FALSE")
# ----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -4]
fit_NBT <- FuzzyGeoNaiveBayes(
    train = Train[, -4],
    cl = Train[, 4], cores = 2)
)
pred_NBT <- predict(fit_NBT, test)
```
FuzzyNaiveBayes

Description

FuzzyNaiveBayes Fuzzy Naive Bayes

Usage

FuzzyNaiveBayes(train, cl, fuzzy = TRUE, m = NULL, Pi = NULL)

Arguments

- **train**: matrix or data frame of training set cases
- **cl**: factor of true classifications of training set
- **fuzzy**: boolean variable to use the membership function
- **m**: is M/N, where M is the number of classes and N is the number of train lines
- **Pi**: is 1/M, where M is the number of classes

Value

A vector of classifications

References


Examples

```r
# Example Fuzzy with Discrete Features
set.seed(1) # determining a seed
data(HouseVotes84)

# Splitting into Training and Testing
split <- caTools::sample.split(t(HouseVotes84[, 1]), SplitRatio = 0.7)
Train <- subset(HouseVotes84, split == "TRUE")
Test <- subset(HouseVotes84, split == "FALSE")
# ----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -1]
fit_FNB <- FuzzyNaiveBayes(
```
train = Train[, -1],
cl = Train[, 1]
)
pred_FNB <- predict(fit_FNB, test)

head(pred_FNB)
head(Test[, 1])

# Example Fuzzy with Continuous Features
set.seed(1) # determining a seed
data(iris)

# Splitting into Training and Testing
split <- caTools::sample.split(t(iris[, 1]), SplitRatio = 0.7)
Train <- subset(iris, split == "TRUE")
Test <- subset(iris, split == "FALSE")

# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -5]
fit_FNB <- FuzzyNaiveBayes(
  train = Train[, -5],
  cl = Train[, 5]
)
pred_FNB <- predict(fit_F NB, test)

head(pred_FNB)
head(Test[, 5])

FuzzyPoissonNaiveBayes

Fuzzy Poisson Naive Bayes

Description
FuzzyPoissonNaiveBayes Fuzzy Poisson Naive Bayes

Usage
FuzzyPoissonNaiveBayes(train, cl, cores = 2, fuzzy = TRUE)

Arguments

train    matrix or data frame of training set cases.
cl       factor of true classifications of training set
cores how many cores of the computer do you want to use to use for prediction (default = 2)
fuzzy boolean variable to use the membership function

Value
A vector of classifications

References

Examples

```r
set.seed(1) # determining a seed
class1 <- data.frame(vari1 = rpois(100,lambda = 2),
                     vari2 = rpois(100,lambda = 2),
                     vari3 = rpois(100,lambda = 2), class = 1)
class2 <- data.frame(vari1 = rpois(100,lambda = 1),
                     vari2 = rpois(100,lambda = 1),
                     vari3 = rpois(100,lambda = 1), class = 2)
class3 <- data.frame(vari1 = rpois(100,lambda = 5),
                     vari2 = rpois(100,lambda = 5),
                     vari3 = rpois(100,lambda = 5), class = 3)
data <- rbind(class1, class2, class3)

# Splitting into Training and Testing
split <- caTools::sample.split(t(data[, 1]), SplitRatio = 0.7)
Train <- subset(data, split == "TRUE")
Test <- subset(data, split == "FALSE")
# ----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -4]
fit_NBT <- FuzzyPoissonNaiveBayes(
  train = Train[, -4],
  cl = Train[, 4], cores = 2
)
pred_NBT <- predict(fit_NBT, test)

head(pred_NBT)
head(Test[, 4])
```
FuzzyTrapezoidalNaiveBayes

Fuzzy Naive Bayes Trapezoidal Classifier

Description

FuzzyTrapezoidalNaiveBayes Fuzzy Naive Bayes Trapezoidal Classifier

Usage

FuzzyTrapezoidalNaiveBayes(train, cl, cores = 2, fuzzy = T)

Arguments

- `train`: matrix or data frame of training set cases.
- `cl`: factor of true classifications of training set
- `cores`: how many cores of the computer do you want to use for prediction (default = 2)
- `fuzzy`: boolean variable to use the membership function

Value

A vector of classifications

References


Examples

```r
set.seed(1) # determining a seed
data(iris)

# Splitting into Training and Testing
split <- caTools::sample.split(t(iris[, 1]), SplitRatio = 0.7)
Train <- subset(iris, split == "TRUE")
Test <- subset(iris, split == "FALSE")

# A vector will be interpreted as a row vector for a single case.
test <- Test[, -5]
fit_NBT <- FuzzyTrapezoidalNaiveBayes(
  train = Train[, -5],
  cl = Train[, 5], cores = 2
)```
pred_NBT <- predict(fit_NBT, test)

head(pred_NBT)
head(Test[, 5])

FuzzyTriangularNaiveBayes

Fuzzy Naive Bayes Triangular Classifier

Description
FuzzyTriangularNaiveBayes Fuzzy Naive Bayes Triangular Classifier

Usage
FuzzyTriangularNaiveBayes(train, cl, cores = 2, fuzzy = TRUE)

Arguments
- train: matrix or data frame of training set cases.
- cl: factor of true classifications of training set
- cores: how many cores of the computer do you want to use for prediction (default = 2)
- fuzzy: boolean variable to use the membership function

Value
A vector of classifications

References

Examples

set.seed(1) # determining a seed
data(iris)

# Splitting into Training and Testing
split <- caTools::sample.split(t(iris[, 1]), SplitRatio = 0.7)
Train <- subset(iris, split == "TRUE")
Test <- subset(iris, split == "FALSE")
# GauNBFuzzyParam

## GamWeightData

### Description

A dataset simulated containing training data from a Gamma Distribution

### Usage

GamWeightData

### Format

A dataset with 600 rows and 4 variables with 1 label.

## GauNBFuzzyParam

### Description

GauNBFuzzyParam Fuzzy Gaussian Naive Bayes Classifier with Fuzzy parameters

### Usage

GauNBFuzzyParam(train, cl, alphacut = 1e-04, metd = 2, cores = 2)
**Arguments**

- **train**: matrix or data frame of training set cases.
- **cl**: factor of true classifications of training set.
- **alphacut**: value of the alpha-cut parameter, this value is between 0 and 1.
- **metd**: Method of transforming the triangle into scalar. It is the type of data entry for the test sample, use metd 1 if you want to use the Yager technique, metd 2 if you want to use the Q technique of the uniformity test (article: Directional Statistics and Shape analysis), and metd 3 if you want to use the Thorani technique.
- **cores**: how many cores of the computer do you want to use for prediction (default = 2)

**Value**

A vector of classifications

**References**


**Examples**

```r
set.seed(1) # determining a seed
data(iris)

# Splitting into Training and Testing
split <- caTools::sample.split(t(iris[, 1]), SplitRatio = 0.7)
Train <- subset(iris, split == "TRUE")
Test <- subset(iris, split == "FALSE")
# ----------------
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
test <- Test[, -5]
fit_FGNB <- GauNBFuzzyParam(
  train = Train[, -5],
  cl = Train[, 5], metd = 1, cores = 2)

pred_FGNB <- predict(fit_FGNB, test)
head(pred_FGNB)
head(Test[, 5])
```
Description

This data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the CQA. The CQA lists nine different types of votes: voted for, paired for, and announced for (these three simplified to yea), voted against, paired against, and announced against (these three simplified to nay), voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three simplified to an unknown disposition).

Usage

data(HouseVotes84)

Format

A data frame with 435 observations on 17 variables:

1. Class Name: 2 (democrat, republican)
2. handicapped-infants: 2 (y,n)
3. water-project-cost-sharing: 2 (y,n)
4. adoption-of-the-budget-resolution: 2 (y,n)
5. physician-fee-freeze: 2 (y,n)
6. el-salvador-aid: 2 (y,n)
7. religious-groups-in-schools: 2 (y,n)
8. anti-satellite-test-ban: 2 (y,n)
9. aid-to-nicaraguan-contras: 2 (y,n)
10. mx-missile: 2 (y,n)
11. immigration: 2 (y,n)
12. synfuels-corporation-cutback: 2 (y,n)
13. education-spending: 2 (y,n)
14. superfund-right-to-sue: 2 (y,n)
15. crime: 2 (y,n)
16. duty-free-exports: 2 (y,n)
17. export-administration-act-south-africa: 2 (y,n)

Source

- Donor: Jeff Schlimmer (Jeffrey.Schlimmer@a.gp.cs.cmu.edu)

These data have been taken from the UCI Repository Of Machine Learning Databases at

- https://archive.ics.uci.edu/datasets
- https://archive.ics.uci.edu/datasets
and were converted to R format by Friedrich Leisch.

References


Examples

data(HouseVotes84)
summary(HouseVotes84)

Usage

PoiNBfuzzyParam(train, cl, alphacut = 1e-04, metd = 2, cores = 2)

Arguments

train matrix or data frame of training set cases.
cl factor of true classifications of training set
alphacut value of the alpha-cut parameter, this value is between 0 and 1.
metd Method of transforming the triangle into scalar, It is the type of data entry for the test sample, use metd 1 if you want to use the Yager technique, metd 2 if you want to use the Q technique of the uniformity test (article: Directional Statistics and Shape analysis), and metd 3 if you want to use the Thorani technique
cores how many cores of the computer do you want to use to use for prediction (default = 2)

Value

A vector of classifications

References

SimulatedData

Examples

```r
set.seed(1) # determining a seed
class1 <- data.frame(vari1 = rpois(100,lambda = 2),
                      vari2 = rpois(100,lambda = 2),
                      vari3 = rpois(100,lambda = 2), class = 1)
class2 <- data.frame(vari1 = rpois(100,lambda = 1),
                      vari2 = rpois(100,lambda = 1),
                      vari3 = rpois(100,lambda = 1), class = 2)
class3 <- data.frame(vari1 = rpois(100,lambda = 5),
                      vari2 = rpois(100,lambda = 5),
                      vari3 = rpois(100,lambda = 5), class = 3)
data <- rbind(class1, class2, class3)

# Splitting into Training and Testing
split <- caTools::sample.split(t(data[, 1]), SplitRatio = 0.7)
Train <- subset(data, split == "TRUE")
Test <- subset(data, split == "FALSE")
#
# matrix or data frame of test set cases.
# A vector will be interpreted as a row vector for a single case.
# test <- Test[, -4]
fit_FPoiNB <- PoiNBFuzzyParam(
  train = Train[, -4],
  cl = Train[, 4], metd = 1, cores = 2
)
pred_FPoiNB <- predict(fit_FPoiNB, test)

head(pred_FPoiNB)
head(Test[, 4])
```

---

**SimulatedData**

**Simulated Data**

**Description**

A dataset containing training data from Gamma Distribution

**Usage**

`SimulatedData`

**Format**

A dataset with 600 rows and 4 variables with 1 label.
VirtualRealityData

Virtual Reality Simulator Data

Description

A dataset containing training data from a virtual reality simulator

Usage

VirtualRealityData

Format

A dataset with 600 rows and 4 variables with 1 label.
Index

* datasets
  GamWeightData, 18
  HouseVotes84, 20
  SimulatedData, 22
  VirtualRealityData, 23

DWFuzzyGammaNaiveBayes, 2

ExpNBFuzzyParam, 3

FuzzyBayesRule, 5
FuzzyBetaNaiveBayes, 6
FuzzyBinomialNaiveBayes, 7
FuzzyExponentialNaiveBayes, 8
FuzzyGammaNaiveBayes, 9
FuzzyGaussianNaiveBayes, 10
FuzzyGeoNaiveBayes, 11
FuzzyNaiveBayes, 13
FuzzyPoissonNaiveBayes, 14
FuzzyTrapezoidalNaiveBayes, 16
FuzzyTriangularNaiveBayes, 17

GamWeightData, 18
GauNBFuzzyParam, 18

HouseVotes84, 20

PoiNBFuzzyParam, 21

SimulatedData, 22

VirtualRealityData, 23