Package 'FuzzyClass'

October 9, 2026

Title Fuzzy and Non-Fuzzy Classifiers

Version 0.2.0

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

Imports caTools, e1071, EnvStats, MASS, mvtnorm, Rdpack, rootSolve, stats, trapezoid

RdMacros Rdpack

Depends R (>= 2.10)

Suggests testthat (>= 3.0.0), knitr, rmarkdown, maxLik

VignetteBuilder knitr

Config/testthat/edition 3

URL https://leapigufpb.github.io/FuzzyClass/, https://github.com/leapigufpb/FuzzyClass

BugReports https://github.com/leapigufpb/FuzzyClass/issues

Language en-US

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Jodavid Ferreira [aut, cre] (ORCID: <https://orcid.org/0000-0002-2131-6464>), Ronei Moraes [ctb] (ORCID: <https://orcid.org/0000-0001-8436-8950>), Liliane Machado [ctb] (ORCID: <https://orcid.org/0000-0002-1182-2929>),

Arthur Ricardo [ctb],
Isaac Araújo [ctb]

Maintainer Jodavid Ferreira <jodavid@protonmail.com>

Repository CRAN

Date/Publication 2026-10-09 14:00:17 UTC

Contents

DWFuzzyGammaNaiveBayes . . . 2
DWFuzzyHipergeometricNaiveBayes . . . 4
ExpNBFuzzyParam . . . 6
FuzzyBayesRule . . . 7
FuzzyBetaNaiveBayes . . . 8
FuzzyBinomialNaiveBayes . . . 9
FuzzyExponentialNaiveBayes . . . 10
FuzzyGammaNaiveBayes . . . 12
FuzzyGaussianNaiveBayes . . . 13
FuzzyGeoNaiveBayes . . . 14
FuzzyHipergeometricNaiveBayes . . . 15
FuzzyNaiveBayes . . . 17
FuzzyPoissonNaiveBayes . . . 18
FuzzyTrapezoidalNaiveBayes . . . 19
FuzzyTriangularNaiveBayes . . . 20
GamWeightData . . . 21
GauNBFuzzyParam . . . 22
HouseVotes84 . . . 23
PoiNBFuzzyParam . . . 25
SimulatedData . . . 26
VirtualRealityData . . . 27

Index **28**

`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	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
fuzzy	boolean variable to use the membership function
wdelta	vector weight each class
weta	vector weight each feature

Value

A vector of classifications

References

Moraes RM, Soares EAMG, Machado LS (2020). "A double weighted fuzzy gamma naive bayes classifier." *Journal of Intelligent & Fuzzy Systems*, **38**(1), 577–588.

Examples

```
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])
```

DWFuzzyHipergeometricNaiveBayes
Double Weighted Fuzzy Hipergeometric Naive Bayes

Description

DWFuzzyHipergeometricNaiveBayes Double Weighted Fuzzy Hipergeometric Naive Bayes

Usage

```
DWFuzzyHipergeometricNaiveBayes(
  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	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
fuzzy	boolean variable to use the membership function
wdelta	vector weight each class
weta	vector weight each feature

Value

A vector of classifications

References

Ferreira J, Machado LS, de Moraes RM (2025). "A New Double Weighted Fuzzy Hypergeometric Naive Bayes Network and its Application for User's Assessment in Virtual Reality Simulators." *International Journal of Computational Intelligence Systems*, **18**(1), 1–17.

Examples

```
set.seed(1)

substituir_zero <- function(x) {
  if (x == 0) {
    while(1){
```

```
     new_valor <- rhyper(1,3,30,10)
      if(new_valor != 0){
        return(new_valor)
      }
    }
  } else {
    return(x)
  }
}

#Building dataframe
class1 <- data.frame(rhyper(72,3,30,10),
                     sample(seq(1,5),72,replace = TRUE), class = "Bem treinado")
colnames(class1)[1] <- "sucessos"
class1$sucessos <- sapply(class1$sucessos, substituir_zero)
colnames(class1)[2] <- "tentativas"
colnames(class1)[3] <- "avaliação"
class2 <- data.frame(rhyper(72,3,17,12),
                     sample(seq(4,7),72,replace = TRUE), class = "Mediano")
colnames(class2)[1] <- "sucessos"
#class2$sucessos <- ifelse(class2$sucessos == 0, 2, class2$sucessos)
colnames(class2)[2] <- "tentativas"
colnames(class2)[3] <- "avaliação"
class3 <- data.frame(rhyper(72,3,9,11),
                     sample(seq(7,10),72,replace = TRUE), class = "Precisa treinar")
colnames(class3)[1] <- "sucessos"
#class3$sucessos <- ifelse(class3$sucessos == 0, 3, class3$sucessos)
colnames(class3)[2] <- "tentativas"
colnames(class3)[3] <- "avaliação"
data <- rbind(class1,class2,class3)

# Weights
weta1 <- c(0.22, 0.33, 0.45)
wdelta1 <- c(0.33,0.33,0.33)

#spliting dataframe
split <- caTools::sample.split(t(data[, 1]), SplitRatio = 0.7)
Train <- subset(data, split == "TRUE")
Test <- subset(data, split == "FALSE")

test <- Test[, -3]
fit_NHT <- DWFuzzyHipergeometricNaiveBayes(
 train = Train[, -3],
 cl = Train[, 3], cores = 2,
 fuzzy = TRUE,
 wdelta = wdelta1,
 weta = weta1)

pred_NHT <- predict(fit_NHT, test)
```

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,
  alp = c(0.35, 0.7, 0.86),
  w = c(0.1, 0.3, 0.6),
  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), 'metd=3' if you want to use the Thorani technique, 'metd=4' if you want to use the alpha-order technique and 'metd=5' if you want to use the GTFN technique (article: A novel approach for arithmetic operations and ranking of generalized fuzzy numbers with application).
alp	When metd for 4, it is necessary to have alp which are alpha-cut defined
w	When metd for 4, it is necessary to have w which are alpha-cut weights defined
cores	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)

Value

A vector of classifications

References

Rodrigues AK, Batista TV, Moraes RM, Machado LS (2016). "A new exponential naive bayes classifier with fuzzy parameters." In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 1188–1194. IEEE.

Examples

```
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` *Fuzzy Bayes Rule*

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`	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
`fuzzy`	boolean variable to use the membership function

Value

A vector of classifications

References

Moraes R, Machado L (2008). "Fuzzy Bayes Rule for On-Line Training Assessment in Virtual Reality Simulators." *Multiple-Valued Logic and Soft Computing*, **14**, 325-338.

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_NBT <- FuzzyBayesRule(
  train = Train[, -5],
  cl = Train[, 5], cores = 2
)

pred_NBT <- predict(fit_NBT, test)

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

`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`	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
`fuzzy`	boolean variable to use the membership function

Value

A vector of classifications

References

Moraes RM, Rodrigues AKG, Soares EAMG, Machado LS (2020). "A new fuzzy beta naive Bayes classifier." In *Developments of Artificial Intelligence Technologies in Computation and Robotics: Proceedings of the 14th International FLINS Conference (FLINS 2020)*, 437–445. World Scientific.

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_NBT <- FuzzyBetaNaiveBayes(
  train = Train[, -5],
  cl = Train[, 5], cores = 2
)

pred_NBT <- predict(fit_NBT, test)

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

`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`	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
`fuzzy`	boolean variable to use the membership function

Value

A vector of classifications

References

Moraes RM, Machado LS (2016). "A Fuzzy Binomial Naive Bayes classifier for epidemiological data." In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 745–750. IEEE.

Examples

```
set.seed(1) # determining a seed
class1 <- data.frame(vari1 = rbinom(100,size = 10, prob = 0.2),
                     vari2 = rbinom(100,size = 10, prob = 0.2),
                    vari3 = rbinom(100,size = 10, prob = 0.2), class = 1)
class2 <- data.frame(vari1 = rbinom(100,size = 10, prob = 0.5),
                     vari2 = rbinom(100,size = 10, prob = 0.5),
                    vari3 = rbinom(100,size = 10, prob = 0.5), class = 2)
class3 <- data.frame(vari1 = rbinom(100,size = 10, prob = 0.8),
                     vari2 = rbinom(100,size = 10, prob = 0.8),
                     vari3 = rbinom(100,size = 10, prob = 0.8), 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 <- FuzzyBinomialNaiveBayes(
  train = Train[, -4],
  cl = Train[, 4], cores = 2
)

pred_NBT <- predict(fit_NBT, test)

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

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	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
fuzzy	boolean variable to use the membership function

Value

A vector of classifications

References

Moraes RM, Machado LS (2016). "A fuzzy exponential naive bayes classifier." In *Uncertainty Modelling in Knowledge Engineering and Decision Making: Proceedings of the 12th International FLINS Conference*, 207–212. World Scientific.

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_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	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
fuzzy	boolean variable to use the membership function

Value

A vector of classifications

References

Moraes RM, Soares EAMG, Machado LS (2018). "A Fuzzy Gamma Naive Bayes Classifier." In *Data Science and Knowledge Engineering for Sensing Decision Support: Proceedings of the 13th International FLINS Conference (FLINS 2018)*, 691–699. World Scientific.

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_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	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
fuzzy	boolean variable to use the membership function

Value

A vector of classifications

References

Moraes RM, Machado LS (2012). "Online Assessment in Medical Simulators Based on Virtual Reality Using Fuzzy Gaussian Naive Bayes." *Journal of Multiple-Valued Logic & Soft Computing*, **18**.

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`	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
`fuzzy`	boolean variable to use the membership function

Value

A vector of classifications

References

Ferreira J, Lopes A, Machado L, Moraes R (2023). "A Novel Fuzzy Geometric Naive Bayes Network for Online Skills Assessment in Training Based on Virtual Reality." *In Proceedings of the 15th International Joint Conference on Computational Intelligence*, 395–401.

Examples

```
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)

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

`FuzzyHipergeometricNaiveBayes`
Fuzzy Hipergeometric Naive Bayes

Description

`FuzzyHipergeometricNaiveBayes` Fuzzy Hipergeometric Naive Bayes

Usage

```
FuzzyHipergeometricNaiveBayes(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`	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
`fuzzy`	boolean variable to use the membership function

Value

A vector of classifications

References

Ferreira J, Machado LS, de Moraes RM (2025). "A New Double Weighted Fuzzy Hypergeometric Naive Bayes Network and its Application for User's Assessment in Virtual Reality Simulators." *International Journal of Computational Intelligence Systems*, **18**(1), 1–17.

Examples

```
set.seed(1) # determining a seed

substituir_zero <- function(x) {
  if (x == 0) {
    while(1){
     new_valor <- rhyper(1,3,30,10)
      if(new_valor != 0){
        return(new_valor)
      }
    }
  } else {
    return(x)
  }
}

#Building dataframe
class1 <- data.frame(rhyper(72,3,30,10),
                     sample(seq(1,5),72,replace = TRUE), class = "Bem treinado")
colnames(class1)[1] <- "sucessos"
class1$sucessos <- sapply(class1$sucessos, substituir_zero)
colnames(class1)[2] <- "tentativas"
colnames(class1)[3] <- "avaliação"
class2 <- data.frame(rhyper(72,3,17,12),
                     sample(seq(4,7),72,replace = TRUE), class = "Mediano")
colnames(class2)[1] <- "sucessos"
colnames(class2)[2] <- "tentativas"
colnames(class2)[3] <- "avaliação"
class3 <- data.frame(rhyper(72,3,9,11),
                     sample(seq(7,10),72,replace = TRUE), class = "Precisa treinar")
colnames(class3)[1] <- "sucessos"
colnames(class3)[2] <- "tentativas"
colnames(class3)[3] <- "avaliação"
data <- rbind(class1,class2,class3)


#spliting dataframe
split <- caTools::sample.split(t(data[, 1]), SplitRatio = 0.7)
Train <- subset(data, split == "TRUE")
Test <- subset(data, split == "FALSE")

test <- Test[, -3]
fit_NHT <- FuzzyHipergeometricNaiveBayes(
 train = Train[, -3],
 cl = Train[, 3], cores = 2
)
```

```
pred_NHT <- predict(fit_NHT, test)
```

FuzzyNaiveBayes *Fuzzy Naive Bayes*

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

Moraes RM, Machado LS (2009). "Another approach for fuzzy naive bayes applied on online training assessment in virtual reality simulators." In *Proceedings of Safety Health and Environmental World Congress*, 62–66.

Examples

```
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])
```

`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`	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
`fuzzy`	boolean variable to use the membership function

Value

A vector of classifications

References

Moraes RM, Machado LS (2015). "A fuzzy poisson naive bayes classifier for epidemiological purposes." In *2015 7th International Joint Conference on Computational Intelligence (IJCCI)*, volume 2, 193–198. IEEE.

Examples

```
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`	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
`fuzzy`	boolean variable to use the membership function

Value

A vector of classifications

References

Lopes A, Ferreira J, Machado LS, Moraes RM (2022). "A New Fuzzy Trapezoidal Naive Bayes Network as basis for Assessment in Training based on Virtual Reality." In *The 15th International FLINS Conference on Machine learning, Multi agent and Cyber physical systems (FLINS 2022)*. Nankai University.

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_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	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)
fuzzy	boolean variable to use the membership function

Value

A vector of classifications

References

Moraes RM, Silva ILA, Machado LS (2020). "Online skills assessment in training based on virtual reality using a novel fuzzy triangular naive Bayes network." In *Developments of Artificial Intelligence Technologies in Computation and Robotics: Proceedings of the 14th International FLINS Conference (FLINS 2020)*, 446–454. World Scientific.

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_NBT <- FuzzyTriangularNaiveBayes(
  train = Train[, -5],
  cl = Train[, 5], cores = 2
)

pred_NBT <- predict(fit_NBT, test)

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

GamWeightData *Gamma Weighted Data*

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` *Fuzzy Gaussian Naive Bayes Classifier with Fuzzy parameters*

Description

`GauNBFuzzyParam` Fuzzy Gaussian Naive Bayes Classifier with Fuzzy parameters

Usage

```
GauNBFuzzyParam(
  train,
  cl,
  alphacut = 1e-04,
  metd = 2,
  alp = c(0.35, 0.7, 0.86),
  w = c(0.1, 0.3, 0.6),
  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), 'metd=3' if you want to use the Thorani technique, 'metd=4' if you want to use the alpha-order technique and 'metd=5' if you want to use the GTFN technique (article: A novel approach for arithmetic operations and ranking of generalized fuzzy numbers with application).
`alp`	When metd for 4, it is necessary to have alp which are alpha-cut defined
`w`	When metd for 4, it is necessary to have w which are alpha-cut weights defined
`cores`	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)

Value

A vector of classifications

References

Moraes RM, Ferreira JA, Machado LS (2021). "A New Bayesian Network Based on Gaussian Naive Bayes with Fuzzy Parameters for Training Assessment in Virtual Simulators." *International Journal of Fuzzy Systems*, **23**(3), 849–861.

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_FGNB <- GauNBFuzzyParam(
  train = Train[, -5],
  cl = Train[, 5], metd = 1, cores = 2
)

pred_FGNB <- predict(fit_FGNB, test)

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

HouseVotes84 *United States Congressional Voting Records 1984*

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

- Source: Congressional Quarterly Almanac, 98th Congress, 2nd session 1984, Volume XL: Congressional Quarterly Inc., ington, D.C., 1985
- 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

Newman, D.J. & Hettich, S. & Blake, C.L. & Merz, C.J. (1998). UCI Repository of machine learning databases [https://archive.ics.uci.edu/datasets]. Irvine, CA: University of California, Department of Information and Computer Science.

Examples

```
data(HouseVotes84)
summary(HouseVotes84)
```

PoiNBFuzzyParam *Fuzzy Poisson Naive Bayes Classifier with Fuzzy parameters*

Description

PoiNBFuzzyParam Fuzzy Poisson Naive Bayes Classifier with Fuzzy parameters

Usage

```
PoiNBFuzzyParam(
  train,
  cl,
  alphacut = 1e-04,
  metd = 2,
  alp = c(0.35, 0.7, 0.86),
  w = c(0.1, 0.3, 0.6),
  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), 'metd=3' if you want to use the Thorani technique, 'metd=4' if you want to use the alpha-order technique and 'metd=5' if you want to use the GTFN technique (article: A novel approach for arithmetic operations and ranking of generalized fuzzy numbers with application).
alp	When metd for 4, it is necessary to have alp which are alpha-cut defined
w	When metd for 4, it is necessary to have w which are alpha-cut weights defined
cores	kept for backward compatibility; prediction is vectorized and does not start parallel workers (default = 2)

Value

A vector of classifications

References

Soares E, Machado L, Moraes R (2016). "Assessment of poisson naive bayes classifier with fuzzy parameters using data from different statistical distributions." In *IV Bazilian Congress on Fuzzy Sistems (CBSF 2016)*, volume 1, 57–68.

Examples

```
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 Gammma Distribuition

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, 21
HouseVotes84, 23
SimulatedData, 26
VirtualRealityData, 27

DWFuzzyGammaNaiveBayes, 2
DWFuzzyHipergeometricNaiveBayes, 4

ExpNBFuzzyParam, 6

FuzzyBayesRule, 7
FuzzyBetaNaiveBayes, 8
FuzzyBinomialNaiveBayes, 9
FuzzyExponentialNaiveBayes, 10
FuzzyGammaNaiveBayes, 12
FuzzyGaussianNaiveBayes, 13
FuzzyGeoNaiveBayes, 14
FuzzyHipergeometricNaiveBayes, 15
FuzzyNaiveBayes, 17
FuzzyPoissonNaiveBayes, 18
FuzzyTrapezoidalNaiveBayes, 19
FuzzyTriangularNaiveBayes, 20

GamWeightData, 21
GauNBFuzzyParam, 22

HouseVotes84, 23

PoiNBFuzzyParam, 25

SimulatedData, 26

VirtualRealityData, 27