Package 'cleanerR'

February 10, 2019

Type Package

Title How to Handle your Missing Data

Version 0.1.1

Author Rafael Silva Pereira

Maintainer Rafael Silva Pereira <r.s.p.models@gmail.com>

Description How to deal with missing data?Based on the concept of almost functional dependencies, a method is proposed to fill missing data, as well as help you see what data is missing.
The user can specify a measure of error and how many combinations he wish to test the dependencies against, the closer to the length of the dataset, the more precise.
But the higher the number, the more time it will take for the process to finish.
If the program cannot predict with the accuracy determined by the user it shall not fill the data, the user then can choose to increase the error or deal with the data another way.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0.9000

Imports plyr, data.table

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2019-02-10 14:03:11 UTC

R topics documented:

`AutoComplete` `AutoComplete` *Asks for a dataframe, a vector of collumn indices and the goal collumn and returns the data frame with the values filled*

Description

`AutoComplete` Asks for a dataframe, a vector of collumn indices and the goal collumn and returns the data frame with the values filled

Usage

```
AutoComplete(df, goal, maxi, repetitions, trigger = 1, ratio = 0.99)
```

Arguments

`df`	A dataframe with the missing values you wish to fill
`goal`	The collum with the missing values you wish to fill
`maxi`	What will be the length of possible combinations you will test example if 2 they will test up to all possible pairs of collums
`repetitions`	Measure of error, the bigger the less likely you will get the right prediction
`trigger`	When you pair all possible combination of tuples a percentage of them will show only once, trigger rejects the set if this percentage is higher than this value
`ratio`	Rejects collumns that the ratio of unique values to total values is higher than this value, primary keys have ratio equal to 1

Examples

```
#The Auto Complete Function shall do the following
#Take a dataframe and a goal collumn to predict
#Tests every combination of vectors limited by a parameter length
#Use the best set to predict with accuracy given by MeanAccuracy function
#Then to run some experiments first lets build a dataframe
e=sample(1:5,1e4,replace=TRUE)
e1=sample(1:5,1e4,replace=TRUE)
e2=sample(1:5,1e4,replace=TRUE)
e=data.frame(e,e1,e2,paste(LETTERS[e],LETTERS[e1]),paste(LETTERS[e],LETTERS[e1],LETTERS[e2])  )
#Now we got a dataframe lets create a copy of it
ce=e
ce[sample(1:nrow(e),0.3*nrow(e)),5]=NA
#So 30 percent of the data is now missing
#Lets try to recover it then with autocomplete
ce1=AutoComplete(df=ce,goal=5,maxi=3,repetitions=nrow(ce),trigger=1)
#We can see how many values are still missing with NA_VALUES
print(NA_VALUES(ce1) )
#And check how many we got wrong by
print(sum(ce1[,5]!=e[,5]) )
# The process could be done for the 4 collum as well
ce=e
ce[sample(1:nrow(e),0.5*nrow(e)),4]=NA
#So 50 percent of the data is now missing
#Lets try to recover it then with autocomplete
ce1=AutoComplete(df=ce,goal=4,maxi=4,repetitions=nrow(ce),trigger=1)
#We can see how many values are still missing with NA_VALUES
print(NA_VALUES(ce1) )
#And check how many we got wrong by
print(sum(ce1[,4]!=e[,4]) )
#Here we can easily see e holds the original data
#ce1 is the recovered data
```

`AutoCompleteTable` `AutoCompleteTable` *Asks for a data.table, a vector of collumn indices and the goal collumn and returns the data frame with the values filled*

Description

`AutoCompleteTable` Asks for a data.table, a vector of collumn indices and the goal collumn and returns the data frame with the values filled

Usage

```
AutoCompleteTable(df, goal, maxi, repetitions, trigger = 1,
  ratio = 0.99)
```

Arguments

df	A data.table with the missing values you wish to fill
goal	The collum with the missing values you wish to fill
maxi	What will be the length of possible combinations you will test example if 2 they will test up to all possible pairs of collums
repetitions	Measure of error, the bigger the less likely you will get the right prediction
trigger	When you pair all possible combination of tuples a percentage of them will show only once, trigger rejects the set if this percentage is higher than this value
ratio	Rejects collumns that the ratio of unique values to total values is higher than this value, primary keys have ratio equal to 1

BestAccuracy BestAccuracy *Asks for a dataframe, a vector of collumn indices and the goal collumn and returns the maximum possible value of accuracy of filling missing values*

Description

BestAccuracy Asks for a dataframe, a vector of collumn indices and the goal collumn and returns the maximum possible value of accuracy of filling missing values

Usage

```
BestAccuracy(df, VECTORS, goal)
```

Arguments

df	A dataframe that you intend to fill missing values, warning this dataframe shall contain no missing values so the user must drop the lines it happens
VECTORS	The columns you wish to use to predict the missing values
goal	The collum with the missing values you wish to fill

Examples

```
#The Best accuracy function tells its user the best accuracy possible.
#Code with two ## is working code but takes longer than 5 seconds
#Given a set and a goal to predict it supposes the following.
#All missing values are contained in the possible values with lowest uncertainty.
#Lets first Consider the iris dataset
#It has the following parameters
print(names(iris))
#As we can see the 5 collumn is species
#Lets use Sepal.Length to predict Species and see Best accuracy
print(BestAccuracy(iris,1,5))
#Now lets use both sepal parameters
print(BestAccuracy(iris,1:2,5))
```

```
#And when using a Petal parameter as well
##print(BestAccuracy(iris,1:3,5))
#We can see that iris even in the best case scenario species can be defined by these 3
#Now lets take a look at the mtcars dataset
##print(names(mtcars))
#Predicting gear using mpg
##print(BestAccuracy(mtcars,1,10))
#But if we try to predict mpg using gear
##print(BestAccuracy(mtcars,10,1))
#So using the best accuracy function we can know whats the best case accuracy
#If the user requires he can also predict more than 1 goal for example
##print(BestAccuracy(mtcars,c(1,3,5),c(10,11)))
#In this case we want to use mpg,disp,drat to predict a pair gear,carb
#To check the confidence of predicted values the user should use all three accuracy functions
```

`BestAccuracyTable` `BestAccuracyTable` *Asks for a data.table, a vector of collumn indices and the goal collumn and returns the maximum possible value of accuracy of filling missing values*

Description

`BestAccuracyTable` Asks for a data.table, a vector of collumn indices and the goal collumn and returns the maximum possible value of accuracy of filling missing values

Usage

```
BestAccuracyTable(df, VECTORS, goal)
```

Arguments

`df`	A data.table that you intend to fill missing values, warning this dataframe shall contain no missing values so the user must drop the lines it happens
`VECTORS`	The collumns you wish to use to predict the missing values
`goal`	The collum with the missing values you wish to fill

`BestVector` `BestVector` *Asks for a dataframe and some parameters and returns the best combination of collums to predict the missing value*

Description

`BestVector` Asks for a dataframe and some parameters and returns the best combination of collums to predict the missing value

Usage

```
BestVector(df, goal, maxi, repetitions, trigger = 1, ratio = 0.99)
```

Arguments

df	A dataframe with the missing values you wish to fill
goal	The collum with the missing values you wish to fill
maxi	What will be the length of possible combinations you will test example if 2 they will test up to all possible pairs of collums
repetitions	Measure of error, the bigger the less likely you will get the right prediction
trigger	When you pair all possible combination of tuples a percentage of them will show only once, trigger rejects the set if this percentage is higher than this value
ratio	Rejects collumns that the ratio of unique values to total values is higher than this value, primary keys have ratio equal to 1

Examples

```
#The Best Vector Function shall do the following
#Take a dataframe and a goal collumn to predict
#Tests every combination of vectors limited by a parameter length
#Returns the best set to predict the goal
#Then to run some experiments first lets build a dataframe
e=sample(1:2,1e2,replace=TRUE)
e1=sample(1:2,1e2,replace=TRUE)
e2=sample(1:2,1e2,replace=TRUE)
e=data.frame(e,e1,e2,paste(LETTERS[e],LETTERS[e1]),paste(LETTERS[e],LETTERS[e1],LETTERS[e2])  )
#We can easily see that to predict the last collumn you need the first three.
#Lets Check what the function will find
z=BestVector(e,5,3,nrow(e),1)
print(z)
#Lets now check what is the best set if we use only 2 collumns maximum
z1=BestVector(e,5,2,nrow(e),1)
print(z1)
#We could also predict which collumn is best to predict the fourth one
z2=BestVector(e,4,2,nrow(e),1)
print(z2)
#We could also take a look at the dataset iris.
#Since this dataset does not repeat lines we must use trigger=0
#To predict Species
z3=BestVector(iris,5,2,nrow(iris),0)
print(names(iris))[z3]
#We can check the accuracy of these predictions with the accuracy functions
print(MeanAccuracy(iris,z3,5))
print(MeanAccuracy(e,z2,4))
print(MeanAccuracy(e,z1,5))
print(MeanAccuracy(iris,z,5))
```

`BestVectorTable` `BestVectorTable` *Asks for a data.table and some parameters and returns the best combination of collums to predict the missing value*

Description

`BestVectorTable` Asks for a data.table and some parameters and returns the best combination of collums to predict the missing value

Usage

```
BestVectorTable(df, goal, maxi, repetitions, trigger = 1, ratio = 0.99)
```

Arguments

`df`	A data.table with the missing values you wish to fill
`goal`	The collum with the missing values you wish to fill
`maxi`	What will be the length of possible combinations you will test example if 2 they will test up to all possible pairs of collums
`repetitions`	Measure of error, the bigger the less likely you will get the right prediction
`trigger`	When you pair all possible combination of tuples a percentage of them will show only once, trigger rejects the set if this percentage is higher than this value
`ratio`	Rejects collumns that the ratio of unique values to total values is higher than this value, primary keys have ratio equal to 1

`Candidates` `Candidates` *Asks for a dataframe and some parameters and returns how close the collums chosen can predict the goal collum Should be used mostly with generate_candidates or preferably BestVector in case you only want the best combination possible for prediction*

Description

`Candidates` Asks for a dataframe and some parameters and returns how close the collums chosen can predict the goal collum Should be used mostly with generate_candidates or preferably BestVector in case you only want the best combination possible for prediction

Usage

```
Candidates(df, goal, vec, repetitions, trigger = 1)
```

Arguments

df	A dataframe with the missing values you wish to fill
goal	The collum with the missing values you wish to fill
vec	a vector of collums you wish to test if can be used to predict the values
repetitions	Measure of error, the bigger the less likely you will get the right prediction
trigger	When you pair all possible combination of tuples a percentage of them will show only once, trigger rejects the set if this percentage is higher than this value

---

CandidatesTable — CandidatesTable *candidates implementation that asks for a data.table object*

Description

CandidatesTable candidates implementation that asks for a data.table object

Usage

```
CandidatesTable(df, goal, vec, repetitions, trigger = 1)
```

Arguments

df	A data.table with the missing values you wish to fill
goal	The collum with the missing values you wish to fill
vec	a vector of collums you wish to test if can be used to predict the values
repetitions	Measure of error, the bigger the less likely you will get the right prediction
trigger	When you pair all possible combination of tuples a percentage of them will show only once, trigger rejects the set if this percentage is higher than this value

---

CompleteDataset — CompleteDataset *Asks for a dataframe, a vector of collumn indices and the goal collumn and returns the data frame with the values filled*

Description

CompleteDataset Asks for a dataframe, a vector of collumn indices and the goal collumn and returns the data frame with the values filled

Usage

```
CompleteDataset(df, rows, goal)
```

Arguments

df	A dataframe with the missing values you wish to fill
rows	The collumns you wish to use to predict the missing values
goal	The collum with the missing values you wish to fill

Examples

```
#The CompleteDataset Function shall do the following
#Take a dataframe and a goal collumn to predict
#Takes a set of vectors to use for prediction
#Use this set to predict with accuracy given by MeanAccuracy function
#Then to run some experiments first lets build a dataframe
e=sample(1:5,1e4,replace=TRUE)
e1=sample(1:5,1e4,replace=TRUE)
e2=sample(1:5,1e4,replace=TRUE)
e=data.frame(e,e1,e2,paste(LETTERS[e],LETTERS[e1]),paste(LETTERS[e],LETTERS[e1],LETTERS[e2])  )
#Now we got a dataframe lets create a copy of it
ce=e
ce[sample(1:nrow(e),0.3*nrow(e)),5]=NA
#So 30 percent of the data is now missing
#Lets try to recover it then with CompleteDataset
#First we must choose a set of vectors to use
#Lets first try with BestVector
vector_c=BestVector(ce,5,4,nrow(ce),1)
ce1=CompleteDataset(ce,rows=vector_c,goal=5)
#We can see how many values are still missing with NA_VALUES
print(NA_VALUES(ce1) )
#And check how many we got wrong by
print(sum(ce1[,5]!=e[,5]) )
#If the user wanted he of course could choose a set of his own, for example
user_set=c(1,3)
ce1=CompleteDataset(ce,rows=user_set,goal=5)
#We can see how many values are still missing with NA_VALUES
print(NA_VALUES(ce1) )
#And check how many we got wrong by
print(sum(ce1[,5]!=e[,5]) )
#But we can see that is not the best solution
#To see how to check the best sets take a look at generate_candidates
# The process could be done for the 4 collum as well
ce=e
ce[sample(1:nrow(e),0.5*nrow(e)),4]=NA
#So 50 percent of the data is now missing
#Lets try to recover it then with CompleteDataset
vector_c=BestVector(ce,4,4,nrow(ce),1)
ce1=CompleteDataset(ce,rows=vector_c,goal=4)
#We can see how many values are still missing with NA_VALUES
print(NA_VALUES(ce1) )
#And check how many we got wrong by
print(sum(ce1[,4]!=e[,4]) )
#Here we can easily see e holds the original data
#ce1 is the recovered data
```

`CompleteDatasetTable` `CompleteDatasetTable` *Asks for a data.table, a vector of collumn indices and the goal collumn and returns the data frame with the values filled*

Description

`CompleteDatasetTable` Asks for a data.table, a vector of collumn indices and the goal collumn and returns the data frame with the values filled

Usage

```
CompleteDatasetTable(df, rows, goal)
```

Arguments

`df`	A data.table with the missing values you wish to fill
`rows`	The collumns you wish to use to predict the missing values
`goal`	The collum with the missing values you wish to fill

`GenerateCandidates` `GenerateCandidates` *Asks for a dataframe and some parameters and returns all possible combinations of collums for prediction that satisfy a given error in input in a list the first element of the list are the combinations while the second is its measure of error,to get the best parameters call BestVector*

Description

`GenerateCandidates` Asks for a dataframe and some parameters and returns all possible combinations of collums for prediction that satisfy a given error in input in a list the first element of the list are the combinations while the second is its measure of error,to get the best parameters call BestVector

Usage

```
GenerateCandidates(df, goal, maxi, repetitions, trigger = 1)
```

Arguments

df	A dataframe with the missing values you wish to fill
goal	The collum with the missing values you wish to fill
maxi	What will be the length of possible combinations you will test example if 2 they will test up to all possible pairs of collums
repetitions	Measure of error, the bigger the less likely you will get the right prediction
trigger	When you pair all possible combination of tuples a percentage of them will show only once, trigger rejects the set if this percentage is higher than this value

Examples

```
#The GenerateCandidates function generates all sets of maximum length maxi.
#Maxi is a measure of error.
#This measure of error is related to the repetitions parameter.
#This parameter should range from 0 (rejects anything less to 100 percent accuracy)
#To number of rows of the dataframe to accept all.
#Lets generate a dataset
e=sample(1:5,1e4,replace=TRUE)
e1=sample(1:5,1e4,replace=TRUE)
e2=sample(1:5,1e4,replace=TRUE)
e=data.frame(e,e1,e2,paste(LETTERS[e],LETTERS[e1]),paste(LETTERS[e],LETTERS[e1],LETTERS[e2])  )
names(e)=c("random1","random2","random3","2randoms","3randoms")
#We can then generate all candidates to predict the 5 collumn
#We shall determine the reject part to 80 percent of the dataframe length
z=GenerateCandidates(df=e, goal=5, maxi=4, repetitions=0.8*nrow(e), trigger = 1)
#We can see z is a list
#z[[1]] is another list that contains all sets that satisfy our request
#z[[2]] is a measure of error, the smaller the more accurate
#Lets then order z[[1]] by z[[2]]
m=z[[1]][order(z[[2]])]
print(m)
#We can see then that m[[1]] holds the best set for prediction, while m[[length(m)]] the worst
#To prove it we can do the following
##cat("The best set to predict",names(e)[5],"is ",names(e)[m[[1]]],"\n"   )
##cat("Its expected accuracy is",MeanAccuracy(e,m[[1]],5),"\n"  )
##cat("The worst set to predict",names(e)[5],"is ",names(e)[m[[length(m)]]],"\n"   )
##cat("Its expected accuracy is",MeanAccuracy(e,m[[length(m)]],5),"\n"  )
```

GenerateCandidatesTable

GenerateCandidatesTable *Asks for a data.table and some parameters and returns all possible combinations of collums for prediction that satisfy a given error in input in a list the first element of the list are the combinations while the second is its measure of error,to get the best parameters call BestVector*

Description

GenerateCandidatesTable Asks for a data.table and some parameters and returns all possible combinations of collums for prediction that satisfy a given error in input in a list the first element of the list are the combinations while the second is its measure of error,to get the best parameters call BestVector

Usage

```
GenerateCandidatesTable(df, goal, maxi, repetitions, trigger = 1)
```

Arguments

df	A data.table with the missing values you wish to fill
goal	The collum with the missing values you wish to fill
maxi	What will be the length of possible combinations you will test example if 2 they will test up to all possible pairs of collums
repetitions	Measure of error, the bigger the less likely you will get the right prediction
trigger	When you pair all possible combination of tuples a percentage of them will show only once, trigger rejects the set if this percentage is higher than this value

---

MeanAccuracy MeanAccuracy *Asks for a dataframe, a vector of collumn indices and the goal collumn the expected value of accuracy of filling missing values if the dataset is representative*

Description

MeanAccuracy Asks for a dataframe, a vector of collumn indices and the goal collumn the expected value of accuracy of filling missing values if the dataset is representative

Usage

```
MeanAccuracy(df, VECTORS, goal)
```

Arguments

df	A dataframe that you intend to fill missing values, warning this dataframe shall contain no missing values so the user must drop the lines it happens
VECTORS	The collumns you wish to use to predict the missing values
goal	The collum with the missing values you wish to fill

Examples

```
#The Mean accuracy function tells its user the expected accuracy.
#Code with two ## is working code but takes longer than 5 seconds
#Given a set and a goal to predict it supposes the following.
#All missing values are representative of the dataset.
#Lets first Consider the iris dataset
#It has the following parameters
print(names(iris))
#As we can see the 5 collumn is species
#Lets use Sepal.Length to predict Species and see Mean accuracy
print(MeanAccuracy(iris,1,5))
#Now lets use both sepal parameters
##print(MeanAccuracy(iris,1:2,5))
#And when using a Petal parameter as well
##print(MeanAccuracy(iris,1:3,5))
#We can see that iris even in the Mean case scenario species can be defined by these 3
#Now lets take a look at the mtcars dataset
##print(names(mtcars))
#Predicting gear using mpg
##print(MeanAccuracy(mtcars,1,10))
#But if we try to predict mpg using gear
##print(MeanAccuracy(mtcars,10,1))
#So using the Mean accuracy function we can know whats the mean case accuracy
#If the user requires he can also predict more than 1 goal for example
##print(MeanAccuracy(mtcars,c(1,3,5),c(10,11)))
#In this case we want to use mpg,disp,drat to predict a pair gear,carb
#To check the confidence of predicted values the user should use all three accuracy functions
```

`MeanAccuracyTable` `MeanAccuracyTable` *Asks for a data.table, a vector of collumn indices and the goal collumn the expected value of accuracy of filling missing values if the dataset is representative*

Description

`MeanAccuracyTable` Asks for a data.table, a vector of collumn indices and the goal collumn the expected value of accuracy of filling missing values if the dataset is representative

Usage

```
MeanAccuracyTable(df, VECTORS, goal)
```

Arguments

`df`	A data.table that you intend to fill missing values, warning this dataframe shall contain no missing values so the user must drop the lines it happens
`VECTORS`	The collumns you wish to use to predict the missing values
`goal`	The collum with the missing values you wish to fill

NA_VALUES NA_VALUES *Asks for a dataframe and returns a table of how many missing values are in each collum*

Description

NA_VALUES Asks for a dataframe and returns a table of how many missing values are in each collum

Usage

```
NA_VALUES(df)
```

Arguments

df	A dataframe with the missing values you wish to fill

Examples

```
#This function is used to detect how many NA values are in a dataframe
# the use is pretty much always the same
#Lets consider the dataset iris
i=iris
print(NA_VALUES(i)  )
#Since it has no missing values it shows none, now lets insert some NA_VALUES there
i[sample(1:nrow(i),0.3*nrow(i)),1]=NA
i[sample(1:nrow(i),0.2*nrow(i)),2]=NA
i[sample(1:nrow(i),0.5*nrow(i)),3]=NA
print(NA_VALUES(i))
#For every dataframe the user just uses this
```

WorstAccuracy WorstAccuracy *Asks for a dataframe, a vector of collumn indices and the goal collumn and returns the minimum possible value of accuracy of filling missing values*

Description

WorstAccuracy Asks for a dataframe, a vector of collumn indices and the goal collumn and returns the minimum possible value of accuracy of filling missing values

Usage

```
WorstAccuracy(df, VECTORS, goal)
```

Arguments

df	A dataframe that you intend to fill missing values, warning this dataframe shall contain no missing values so the user must drop the lines it happens
VECTORS	The collumns you wish to use to predict the missing values
goal	The collum with the missing values you wish to fill

Examples

```
#The Worst accuracy function tells its user the worst case accuracy possiblle.
#Code with two ## is working code but takes longer than 5 seconds
#Given a set and a goal to predict it supposes the following.
#All missing values are contained in the possible values with highest uncertainty.
#Lets first Consider the iris dataset
#It has the following parameters
print(names(iris))
#As we can see the 5 collumn is species
#Lets use Sepal.Length to predict Species and see Worst accuracy
print(WorstAccuracy(iris,1,5))
#Now lets use both sepal parameters
print(WorstAccuracy(iris,1:2,5))
#And when using a Petal parameter as well
##print(WorstAccuracy(iris,1:3,5))
#We can see that iris even in the worst case scenario species can be defined by these 3
#Now lets take a look at the mtcars dataset
##print(names(mtcars))
#Predicting gear using mpg
##print(WorstAccuracy(mtcars,1,10))
#But if we try to predict mpg using gear
##print(WorstAccuracy(mtcars,10,1))
#So using the Worst accuracy function we can know whats the worst case accuracy
#If the user requires he can also predict more than 1 goal for example
##print(WorstAccuracy(mtcars,c(1,3,5),c(10,11)))
#In this case we want to use mpg,disp,drat to predict a pair gear,carb
#To check the confidence of predicted values the user should use all three accuracy functions
```

WorstAccuracyTable — WorstAccuracyTable *Asks for a data.table, a vector of collumn indices and the goal collumn and returns the minimum possible value of accuracy of filling missing values*

Description

WorstAccuracyTable Asks for a data.table, a vector of collumn indices and the goal collumn and returns the minimum possible value of accuracy of filling missing values

Usage

```
WorstAccuracyTable(df, VECTORS, goal)
```

Arguments

df	A data.table that you intend to fill missing values, warning this dataframe shall contain no missing values so the user must drop the lines it happens
VECTORS	The collumns you wish to use to predict the missing values
goal	The collum with the missing values you wish to fill

Index