Raquel Monteiro Silva

Mestranda em Ecologia no IB-USP

exec

Trabalho Final

Propostas

Função amostragem()

Página de ajuda

```
amostragem                package:nenhum                R Documentation

Amostragem de data frame, com base nos critérios dados pelo usuário

Description:
Estratifica e amostra um data frame grande, com base nas informações
fornecidas referentes ao tamanho da amostra desejada,
a coluna a ser estratificada, as categorias a serem buscadas nas linhas, o
tipo de amostragem a ser feito, se a
amostragem deve ser com reposição e quais colunas do data frame original
devem ser excluídas.

Usage:

amostragem(dados,amostra,colun,categ,tipo=c("aleat",
"prop"),proporcao=NULL,reposicao=FALSE,remov)

Arguments:

dados:      Dataframe; dados originais a serem amostrados.

amostra:    Numerico positivo; valor sera transformado em integer. Tamanho
da amostra a ser gerada (em número de
        linhas).

colun:      Numerico ou caracter; deve especificar o nome da coluna do
dataframe a ser estratificada. Veja 'details'.
```

```
categ:      Numerico ou caracter; informações a serem buscadas nas linhas
para estratificação.

tipo:       Caracter; tipo de amostragem a ser feita. Veja 'details'.

proporcao:  Numerico; proporcoes a serem seguidas para amostragem (valores
informados pelo usuário). Veja 'details'.

reposicao:  Logico; determina se a amostragem é com reposição.

remov:      Numerico ou caracter; colunas a serem removidas do data frame.

Details:

Os argumentos "dados" e "amostra" devem ser obrigatoriamente fornecidos na
função.

Se o argumento "colun" não for informado, a amostragem feita sera aleatória.
Se "colun" for dado, mas o argumento "categ"
não, a estratificação sera feita a partir da primeira coluna informada em
"colun". A estratificação e amostragem é feita
apenas para uma coluna por vez, portanto se "colun" tiver duas ou mais
informações apenas a primeira será considerada.

O argumento "tipo" aceita: "aleat", onde faz amostragem aleatória, e "prop",
em que amostra respeitando a proporção
original das linhas da coluna indicada. Se "tipo" não for informado, o
default é fazer amostragem aleatória, caso
"proporcao" também não tenha sido informado.

O argumento "proporcao" faz amostragem dos dados de acordo com a proporção
informada pelo usuário. Os valores numéricos
inseridos em "proporcao", correspondentes a proporção da amostra para cada
categoria, devem seguir a ordem
crescente/alfabética das categorias. Se "proporcao" não for informado, o
default é NULL.

Se "reposição" não for informado, a amostragem é feita sem reposição.

Value:

A função retorna:
        Um dataframe com as linhas amostradas de acordo com as
especificações do usuário e colunas que não foram
        removidas.

Warning:

A função é interrompida e retorna mensagem de erro se "dados" não for um
dataframe, e se "amostra" não for um valor
```

```
numérico.

Se o usuário inserir em "proporcao" uma quantidade de valores não compatível
com a quantidade de categorias nas linhas a
função retorna mensagem de erro. Se o usuário inserir valores que nao somam
=1 a função também retorna mensagem de erro.

Author(s):

Raquel Monteiro Silva
raquel.monteiro.silva@usp.br

References:

Multinomial distribution. From Wikipedia, the free encyclopedia.
https://en.wikipedia.org/wiki/Multinomial_distribution

See Also:

sample(), rmultinom(), subset()

Examples:

#criando vetores para formar o data frame
tipo_doc<-rep(letters,len=100)
id_doc<-round(abs(rnorm(100,2,1)),4)
local<-rep(1:7,each=3,len=100)
ano<-c(rep(2013,20),rep(2014,14),rep(2015,30),rep(2016,26),rep(2017,10))

#criando o data frame de exemplo
df<-data.frame(tipo_doc,id_doc,local,ano)

amostragem(df,amostra=10,categ=c(2013,2015),reposicao=TRUE)
amostragem(df,amostra=15,tipo="prop")  ##como "colun" não foi informado,
mensagem de aviso emitida
amostragem(df,amostra=10,colun="local",
tipo="prop",remov="id_doc",reposicao=TRUE)
amostragem(df,amostra=15,colun="local",proporcao=c(0.2,0.2,0.3,0.3))  ##erro
na definição das proporcões

##amostragem com proporção dada, apenas para a primeira coluna informada
amostragem(df,amostra=12,colun=c("ano","tipo_doc"),categ=c(2016,2017),propor
cao=c(0.3,0.7))

##é necessário que a coluna que você deseja amostrar seja da mesma classe
que as informações forncecidas para a função
df$tipo_doc<-as.character(df$tipo_doc)
amostragem(df,amostra=10,colun="tipo_doc",categ=c("r","a","q","u","e","l"))
```

Código da função

```
################FUNCAO AMOSTRAGEM################
amostragem<-function(dados,amostra,colun,categ,tipo=c("aleat",
"prop"),proporcao=NULL,reposicao=FALSE,remov) #argumentos da função. Valores
colocados são o default. Argumento "dados" = data frame de onde vem os dados
a serem amostrados. "amostra" = tamanho da amostra desejada pelo usuário em
numero de linhas. "colun" é a coluna(s) escolhida pelo usuario para
estratificar.
  #"categ" são os dados a serem buscados e selecionados nas linhas. "tipo" é
o tipo de amostragem a ser feito (podendo ser aleatorio ou proporcional),
sendo "aleat" o default. caso o usuario queira dar as proporcoes da amostra,
as insere em "proporcao". "reposicao" define se se a amostragem sera com
reposicao, sendo FALSE o default. "remov" indica as colunas a serem
removidas da amostra final
{
  if(class(dados)!="data.frame"){ #caso os dados inseridos na funcao nao
seja um data frame
    stop(cat("Objeto precisa ser um data frame")) #para e manda msg de erro
  }
  if(class(amostra)!="numeric"){ #caso o argumento amostra não seja numerico
    stop(cat("Amostra precisa ser um número inteiro"))#para e manda msg de
erro
  }
  if(is.null(proporcao)){ #se o argumento proporcao nao for dado
    tipo<-match.arg(tipo) #o valor dado para o argumento tipo  sera usado
como argumento, sendo "aleat" o default
  }
  amostra<-as.integer(amostra) #transforma o argumento amostra em integer
(numero inteiro), ja que não é possível ter uma amostra não inteira de
linhas
  if(missing(colun)){ #se o argumento "colun" nao for especificado
    if(missing(categ)){ #se o argumento "categ" tambem nao for dado
      subs.dados<-dados #o objeto subs.dados é igual a dados (ja que nao
houve estratificacao nem por linhas nem por colunas)
    }
    else{ #se "categ" for informado
      subs.dados<-data.frame(NULL) #cria data frame vazio
      for(i in 1:ncol(dados)){ #se for dado o argumento "categ", ciclos para
o numero de colunas de dados
        for(j in 1:length(categ)){ #ciclos para todas as posicoes em "categ"
        selec_linhas<-subset(dados,dados[,i]==categ[j], drop=FALSE) #faz
ciclos em que busca em cada coluna as linhas que são iguais ao argumento
"categ", mantendo as colunas que não tem essa informação
        subs.dados<-rbind(subs.dados,selec_linhas) #preenche data frame
vazio com os subconjuntos correspondetes as categorias informadas em "categ"
        }
      }
    }
```

```
    if(tipo!="aleat"){  #nesse caso a amostragem sempre vai ser aleatoria,
ja que não da para estratificar os dados sem especificar com qual coluna
deve ser estratificado
      message("Como colun não foi definida, fazendo amostragem aleatória")
#logo, uma mensagem de aviso é dada quando tenta fazer uma amostragem
diferente da aleatoria
    }
    if(reposicao==TRUE){ #amostragem aleatoria com reposição
      dados.tot<-subs.dados[sample(nrow(subs.dados),amostra,replace=TRUE),]
#amostra aleatoriamente as linhas de subs.dados para o tamanho dado no
argumento "amostra"
    }
    else{ #amostragem aleatoria sem reposição
      dados.tot<-subs.dados[sample(nrow(subs.dados),amostra,replace=FALSE),]
#amostra aleatoriamente as linhas de subs.dados para o tamanho dado no
argumento "amostra"
    }
    if(missing(remov)){ #se o argumento remov nao for dado
      return(dados.tot) #retorna para o usuario o data frame amostrado
    }
    dados.amost<-dados.tot[ , !(names(dados.tot) %in% remov)] #se remov for
informado, cria objeto com as colunas de dados.tot diferentes das dadas no
argumento "remov" (que devem ser removidas)
    return(dados.amost) #retorna para o usuario um data frame amostrado a
partir de "dados" com as informações e tamanho especificados
  }
  if(missing(categ)){ #se o argumento categ não for dado, mas colun sim
    subs.dados<-dados #o objeto subs.dados é igual a dados (ja que nao foi
selecionada nenhum categrmação a ser buscada nas linhas)
    if(is.null(proporcao)==FALSE & missing(tipo)){ ##caso o argumento "tipo"
nao tenha sido dado e "proporcao" não seja nulo, ira fazer amostragem de
acordo com as proporcoes dadas pelo usuario. As entradas devem ser dadas em
ordem afabetica/numerica das categorias das linhas (ao inves de ordem de
aparecimento)
      prop.dados<-as.numeric(proporcao) #cria um objeto que guarda os
valores dados em "proporcao" como valores numericos
      unicos<-sort(unique(subs.dados[,colun[1]])) #criando um objeto que
guarda os nomes únicos de linhas (categorias) da coluna selecionada, em
ordem crescente/alfabetica
      if(sum(proporcao)!=1){ #caso a soma dos valores dados em "proporcao"
nao seja igual a 1 (ou seja, as propocoes nao vao dar 100% da amostra)
        stop(print("Valores em proporcao devem somar 1")) #para e emite uma
mensagem de erro
      }
      if(length(proporcao)!=length(unicos)){ #caso o comprimento do objeto
que contem os valores das proporcoes nao seja do mesmo comprimento do objeto
que contem os nomes das categorias (ou seja, ha mais/menos proporcoes do que
categorias nas linhas)
        stop(cat("Quantidade de valores em proporcao inconsistente")) #para
e emite uma mensagem de erro
      }
```

```
      else{ #caso nao aconteca nenhum dos dois if anteriores
        linhas<-
rmultinom(1:length(unique(subs.dados[,colun[1]])),amostra,prob=prop.dados)
#definindo o numero de linhas de cada categoria dentro das linhas totais da
amostra (isso é feito gerando uma distribuicao multinomial com as proporcoes
de cada categoria informadas pelo usuario)
        linhas2<-data.frame(linhas) #criando um data frame com a quantidade
de linhas de cada categoria (nao importa que nao tenha o nome das
categorias, mantem as posicoes das categorias)
        num_linhas<-linhas2$linhas #criando um vetor a partir da coluna do
data frame com a quantidade de linhas de cada categoria
        dados.tot<-data.frame(NULL) #criando um data frame vazio para
guardar a amostra
        if(reposicao==TRUE){ #amostragem com reposicao caso o argumento
"reposicao" for true
          for(i in 1:length(unicos)){ #criando um ciclo com for para criar a
amostra
            subs.cat<-subset(subs.dados,subs.dados[,colun[1]]==unicos[i])
#criando um subset do data frame dos dados que possui, para cada valor de i,
apenas as linhas de cada uma das categorias
            cada_categ<-
subs.cat[sample(nrow(subs.cat),num_linhas[i],replace=TRUE),] #a partir do
data frame com apenas as linhas de cada categoria, faz uma amostragem das
linhas com a quantidade de linhas correspondente a categoria determinada por
i, com reposição
            dados.tot<-rbind(dados.tot,cada_categ) #usa o data frame vazio
para guardar os resultados gerados pela amostragem de cada categoria,
gerando o output
          }
        }
        if(reposicao==FALSE){#amostragen sem reposição quando o argumento
reposicao=false
          for(i in 1:length(unicos)){ #criando um ciclo com for para criar a
amostra
            subs.cat<-subset(subs.dados,subs.dados[,colun[1]]==unicos[i])
#criando um subset do data frame dos dados que possui, para cada valor de i,
apenas as linhas de cada uma das categorias
            cada_categ<-subs.cat[sample(nrow(subs.cat),num_linhas[i]),] #a
partir do data frame com apenas as linhas de cada categoria, faz uma
amostragem das linhas com a quantidade de linhas correspondente a categoria
determinada por i, sem reposição
            dados.tot<-rbind(dados.tot,cada_categ) #usa o data frame vazio
para guardar os resultados gerados pela amostragem de cada categoria,
gerando o output
          }
        }
        if(missing(remov)){ #se o argumento remov nao for dado
          return(dados.tot) #retorna para o usuario o data frame amostrado
          }
        dados.amost<-dados.tot[ , !(names(dados.tot) %in% remov)] #se remov
```

```
for informado, cria objeto com as colunas de dados.tot diferentes das dadas
no argumento "remov" (que devem ser removidas)
          return(dados.amost) #retorna para o usuario um data frame
amostrado a partir de "dados" com as informações e tamanho especificados
      }
    }
    if(tipo=="aleat"){ #caso o argumento "tipo" tenha sido para fazer
amostragem aleatoria
      if(reposicao==TRUE){ #amostragem aleatoria com reposição
        dados.tot<-
subs.dados[sample(nrow(subs.dados),amostra,replace=TRUE),] #amostra
aleatoriamente as linhas de subs.dados para o tamanho dado no argumento
"amostra"
      }
      else{ #amostragem aleatoria sem reposição
        dados.tot<-
subs.dados[sample(nrow(subs.dados),amostra,replace=FALSE),] #amostra
aleatoriamente as linhas de subs.dados para o tamanho dado no argumento
"amostra"
      }
      if(missing(remov)){ #se o argumento remov nao for dado
        return(dados.tot)#retorna para o usuario o data frame amostrado
      }
      dados.amost<-dados.tot[ , !(names(dados.tot) %in% remov)] #se remov
for informado, cria objeto com as colunas de dados.tot diferentes das dadas
no argumento "remov" (que devem ser removidas)
      return(dados.amost) #retorna para o usuario um data frame amostrado a
partir de "dados" com as informações e tamanho especificados
    }
    if(tipo=="prop"){ #caso o argumento "tipo" tenha sido para fazer
amostragem proporcional (respeitando a proporcao dos dados originais)
      data.freq<-data.frame(table(subs.dados[,colun[1]])) #chamando um data
frame com as frequencias de cada categoria unica das linhas da coluna dada
no argumento "colun" (so considera a primeira coluna dada no argumento, nao
estratifica para mais de uma coluna)
      freq<-data.freq$Freq #colocando a coluna freq do objeto data.freq em
um objeto novo
      freq.rel<-freq/nrow(dados) #calculando a frequencia de cada categoria
das linhas em relacao ao total de linhas na coluna selecionada
      unicos<-sort(unique(subs.dados[,colun[1]])) #criando um objeto que
guarda os nomes das categorias da coluna selecionada, em ordem
crescente/alfabetica (pois data.freq é dado em ordem crescente das linhas,
logo as frequencias seguem esta ordem)
      linhas<-
rmultinom(1:length(unique(subs.dados[,colun[1]])),amostra,prob=freq.rel)
#definindo o numero de linhas de cada categoria dentro das linhas totais da
amostra (isso é feito gerando uma distribuicao multinomial com as proporcoes
de cada categoria)
      linhas2<-data.frame(linhas) #criando um data frame com a quantidade de
linhas de cada categoria (nao importa que nao tenha o nome das categorias,
mantem as posicoes das categorias)
```

```
      num_linhas<-linhas2$linhas #criando um vetor a partir da coluna do
data frame com a quantidade de linhas de cada categoria
      dados.tot<-data.frame(NULL) #criando um data frame vazio para guardar
a amostra
      if(reposicao==TRUE){ #amostragem com reposicao caso o argumento
"reposicao" for true
        for(i in 1:length(unicos)){ #criando um ciclo com for para criar a
amostra
          subs.cat<-subset(subs.dados,subs.dados[,colun[1]]==unicos[i])
#criando um subset do data frame dos dados que possui, para cada valor de i,
apenas as linhas de cada uma das categorias
          cada_categ<-
subs.cat[sample(nrow(subs.cat),num_linhas[i],replace=TRUE),] #a partir do
data frame com apenas as linhas de cada categoria, faz uma amostragem das
linhas com a quantidade de linhas correspondente a categoria determinada por
i, com reposição
          dados.tot<-rbind(dados.tot,cada_categ) #usa o data frame vazio
para guardar os resultados gerados pela amostragem de cada categoria,
gerando o output
        }
      }
      if(reposicao==FALSE){ #se o argumento reposicao for FALSE ou não for
dado, amostragem sem reposicao
        for(i in 1:length(unicos)){ #criando um ciclo com for para criar a
amostra
          subs.cat<-subset(subs.dados,subs.dados[,colun[1]]==unicos[i])
#criando um subset do data frame dos dados que possui, para cada valor de i,
apenas as linhas de cada uma das categorias
          cada_categ<-subs.cat[sample(nrow(subs.cat),num_linhas[i]),] #a
partir do data frame com apenas as linhas de cada categoria, faz uma
amostragem das linhas com a quantidade de linhas correspondente a categoria
determinada por i, sem reposição
          dados.tot<-rbind(dados.tot,cada_categ) #usa o data frame vazio
para guardar os resultados gerados pela amostragem de cada categoria,
gerando o output
        }
      }
        if(missing(remov)){ #se o argumento remov nao for dado
          return(dados.tot) #retorna para o usuario o data frame amostrado
        }
        dados.amost<-dados.tot[ , !(names(dados.tot) %in% remov)] #se remov
for informado, cria objeto com as colunas de dados.tot diferentes das dadas
no argumento "remov" (que devem ser removidas)
          return(dados.amost) #retorna para o usuario um data frame
amostrado a partir de "dados" com as informações e tamanho especificados
    }
  }
  if(!is.null(colun)&!is.null(categ)){ #se "colun" for dado, e "categ" tiver
comprimento 1
    subs.dados<-data.frame(NULL) #criando um data frame vazio para guardar o
```

```
subconjunto selecionado de dados
    for(i in 1:length(categ)){ #ciclos para o comprimento de "categ"
      selec_linhas<-subset(dados,dados[,colun[1]]==categ[i], drop=FALSE)
#cria um objeto que contenha um subset de dados, contendo apenas as linhas
de "colun"(posição 1, se for dada mais de uma posicao) iguais ao dado em
todas posicoes de "categ"
      subs.dados<-rbind(subs.dados,selec_linhas) #coloca no data frame
criado anteriormente a junção de todos os subconjuntos criados na linha
anterior
    }
    if(is.null(proporcao)==FALSE & missing(tipo)){ ##caso o argumento "tipo"
nao tenha sido dado e "proporcao" não seja nulo, ira fazer amostragem de
acordo com as proporcoes dadas pelo usuario. As entradas devem ser dadas em
ordem afabetica/numerica das categorias das linhas (ao inves de ordem de
aparecimento)
      prop.dados<-as.numeric(proporcao) #cria um objeto que guarda os
valores dados em "proporcao" como valores numericos
      unicos<-sort(unique(subs.dados[,colun[1]])) #criando um objeto que
guarda os nomes únicos de linhas (categorias) da coluna selecionada, em
ordem crescente/alfabetica
      if(sum(proporcao)!=1){ #caso a soma dos valores dados em "proporcao"
nao seja igual a 1 (ou seja, as propocoes nao vao dar 100% da amostra)
        stop(print("Valores em proporcao devem somar 1")) #para e emite uma
mensagem de erro
      }
      if(length(proporcao)!=length(unicos)){ #caso o comprimento do objeto
que contem os valores das proporcoes nao seja do mesmo comprimento do objeto
que contem os nomes das categorias (ou seja, ha mais/menos proporcoes do que
categorias nas linhas)
        stop(cat("Quantidade de valores em proporcao inconsistente")) #para
e emite uma mensagem de erro
      }
      else{ #caso nao aconteca nenhum dos dois if anteriores
        linhas<-
rmultinom(1:length(unique(subs.dados[,colun[1]])),amostra,prob=prop.dados)
#definindo o numero de linhas de cada categoria dentro das linhas totais da
amostra (isso é feito gerando uma distribuicao multinomial com as proporcoes
de cada categoria informadas pelo usuario)
        linhas2<-data.frame(linhas) #criando um data frame com a quantidade
de linhas de cada categoria (nao importa que nao tenha o nome das
categorias, mantem as posicoes das categorias)
        num_linhas<-linhas2$linhas #criando um vetor a partir da coluna do
data frame com a quantidade de linhas de cada categoria
        dados.tot<-data.frame(NULL) #criando um data frame vazio para
guardar a amostra
        if(reposicao==TRUE){ #amostragem com reposicao caso o argumento
"reposicao" for true
          for(i in 1:length(unicos)){ #criando um ciclo com for para criar a
amostra
            subs.cat<-subset(subs.dados,subs.dados[,colun[1]]==unicos[i])
#criando um subset do data frame dos dados que possui, para cada valor de i,
```

```
apenas as linhas de cada uma das categorias
            cada_categ<-
subs.cat[sample(nrow(subs.cat),num_linhas[i],replace=TRUE),] #a partir do
data frame com apenas as linhas de cada categoria, faz uma amostragem das
linhas com a quantidade de linhas correspondente a categoria determinada por
i, com reposição
            dados.tot<-rbind(dados.tot,cada_categ) #usa o data frame vazio
para guardar os resultados gerados pela amostragem de cada categoria,
gerando o output
          }
        }
        if(reposicao==FALSE){#amostragen sem reposição quando o argumento
reposicao=false
          for(i in 1:length(unicos)){ #criando um ciclo com for para criar a
amostra
            subs.cat<-subset(subs.dados,subs.dados[,colun[1]]==unicos[i])
#criando um subset do data frame dos dados que possui, para cada valor de i,
apenas as linhas de cada uma das categorias
            cada_categ<-subs.cat[sample(nrow(subs.cat),num_linhas[i]),] #a
partir do data frame com apenas as linhas de cada categoria, faz uma
amostragem das linhas com a quantidade de linhas correspondente a categoria
determinada por i, sem reposição
            dados.tot<-rbind(dados.tot,cada_categ) #usa o data frame vazio
para guardar os resultados gerados pela amostragem de cada categoria,
gerando o output
          }
        }
        if(missing(remov)){ #se o argumento remov nao for dado
          return(dados.tot) #retorna para o usuario o data frame amostrado
        }
        dados.amost<-dados.tot[ , !(names(dados.tot) %in% remov)] #se remov
for informado, cria objeto com as colunas de dados.tot diferentes das dadas
no argumento "remov" (que devem ser removidas)
        return(dados.amost) #retorna para o usuario um data frame amostrado
a partir de "dados" com as informações e tamanho especificados
      }
    }
    if(tipo=="aleat"|length(categ)==1){ #se o valor dado em "tipo" for
"aleat" OU "categ" tiver apenas uma categoria especificada (amostragem é
aleatória, já que só uma categoria foi dada e não é possível fazer
amostragem proporcional de apenas uma categoria)
      if(reposicao==TRUE){ #amostragem aleatoria com reposição
      dados.tot<-subs.dados[sample(nrow(subs.dados),amostra,replace=TRUE),]
#amostra aleatoriamente as linhas de subs.dados para o tamanho dado no
argumento "amostra", com reposicao
      }
      else{ #amostragem aleatoria sem reposição
      dados.tot<-subs.dados[sample(nrow(subs.dados),amostra,replace=FALSE),]
#amostra aleatoriamente as linhas de subs.dados para o tamanho dado no
argumento "amostra", sem reposicao
```

```
      }
      if(missing(remov)){ #se o argumento remov nao for dado
        return(dados.tot) #retorna para o usuario o data frame amostrado
      }
      dados.amost<-dados.tot[ , !(names(dados.tot) %in% remov)] #se remov
for informado, cria objeto com as colunas de dados.tot diferentes das dadas
no argumento "remov" (que devem ser removidas)
        return(dados.amost) #retorna para o usuario um data frame amostrado
a partir de "dados" com as informações e tamanho especificados
    }
    if(tipo=="prop"){ #se o "tipo" de amostragem escolhido for o
proporcional aos dados
      data.freq<-data.frame(table(subs.dados[,colun[1]])) #criando um data
frame com as frequencias de cada categoria selecionada das linhas ("categ")
da coluna dada no argumento "colun" (so considera a primeira coluna dada no
argumento, nao estratifica para mais de uma coluna)
      freq<-data.freq$Freq #colocando a coluna freq do objeto data.freq em
um objeto novo
      freq.rel<-freq/nrow(dados) #calculando a frequencia de cada categoria
das linhas em relacao ao total de linhas na coluna selecionada
      unicos<-sort(unique(subs.dados[,colun[1]])) #criando um objeto que
guarda os nomes das categorias da coluna selecionada, em ordem
crescente/alfabetica (pois data.freq é dado em ordem crescente das linhas,
logo as frequencias seguem esta ordem)
      linhas<-
rmultinom(1:length(unique(subs.dados[,colun[1]])),amostra,prob=freq.rel)
#definindo o numero de linhas de cada categoria dentro das linhas totais da
amostra (isso é feito gerando uma distribuicao multinomial com as proporcoes
de cada categoria)
      linhas2<-data.frame(linhas) #criando um data frame com a quantidade de
linhas de cada categoria (nao importa que nao tenha o nome das categorias,
mantem as posicoes das categorias)
      num_linhas<-linhas2$linhas #criando um vetor a partir da coluna do
data frame com a quantidade de linhas de cada categoria
      dados.tot<-data.frame(NULL) #criando data frame vazio para ser
preenchido dentro do for
      if(reposicao==TRUE){ #amostragem com reposicao caso o argumento
"reposicao" for true
        for(i in 1:length(unicos)){ #criando um ciclo com for para criar a
amostra
          subs.cat<-subset(subs.dados,subs.dados[,colun[1]]==unicos[i])
#criando um subset do data frame dos dados que possui, para cada valor de i,
apenas as linhas de cada uma das categorias
          cada_categ<-
subs.cat[sample(nrow(subs.cat),num_linhas[i],replace=TRUE),] #a partir do
data frame com apenas as linhas de cada categoria, faz uma amostragem das
linhas com a quantidade de linhas correspondente a categoria determinada por
i, com reposição
          dados.tot<-rbind(dados.tot,cada_categ) #usa o data frame vazio
para guardar os resultados gerados pela amostragem de cada categoria,
gerando o output
```

```
        }
      }
      if(reposicao==FALSE){ #se reposicao for FALSE ou nao for dado, faz
amostragem sem reposicao
        for(i in 1:length(unicos)){ #criando um ciclo com for para criar a
amostra
          subs.cat<-subset(subs.dados,subs.dados[,colun[1]]==unicos[i])
#criando um subset do data frame dos dados que possui, para cada valor de i,
apenas as linhas de cada uma das categorias
          cada_categ<-subs.cat[sample(nrow(subs.cat),num_linhas[i]),] #a
partir do data frame com apenas as linhas de cada categoria, faz uma
amostragem das linhas com a quantidade de linhas correspondente a categoria
determinada por i, sem reposição
          dados.tot<-rbind(dados.tot,cada_categ) #usa o data frame vazio
para guardar os resultados gerados pela amostragem de cada categoria,
gerando o output
        }
      }
        if(missing(remov)){ #se o argumento remov nao for dado
          return(dados.tot) #retorna para o usuario o data frame amostrado
        }
        dados.amost<-dados.tot[ , !(names(dados.tot) %in% remov)] #se remov
for informado, cria objeto com as colunas de dados.tot diferentes das dadas
no argumento "remov" (que devem ser removidas)
        return(dados.amost) #retorna para o usuario um data frame amostrado
a partir de "dados" com as informações e tamanho especificados
    }
  }
}
```

Help da função amostragem

Função amostragem