TRABALHO FINAL

Help

Por favor, salve o teste em sua pasta diretório.

```
diag.tu()                    package:unknown                              R

Diagnóstico de turmas

Descrição:

A função diag.tu salva um .csv com as médias, podendo ser média ponderada ou
comum;
gera e salva gráficos de diagnóstico; lista e salva os nomes de alunos com
médias baixas;
e pode gerar um teste de ANOVA comparando as médias das turmas.

Argumentos:

nomes = “character”. Nome da coluna ou vetor que contem o nome dos alunos.
notas = “numeric”. Data.frame contendo as colunas de interesse.
prof = “character”. Vetor contendo o nome do professor.
pesos = “numeric”. Vetor contendo um peso por coluna de notas, referente ao
cálculo da média ponterada. Exemplo: “c(peso para prova 1, peso para prova 2
(...))”. Padrão média não-ponderada.
corte = “numeric”. Valor inteiro do corte que a escola considera entre nota
azul e vermelha. Ex.: 5.
turma = “character”. Nome da coluna ou vetor referente à qual turma
pertencem os alunos, na mesma ordem de nomes. Máximo recomendado de 10
turmas.
graf = Padrão (b) cria boxplots por turma. Caso (h) substitui pelo
histograma por turmas.
stt = Argumento por padrão FALSE, caso TRUE faz um teste tipo anova das
medias das turma.

Uso:

diag.tu<-function(nomes, prof, notas, pesos=NULL, corte, turma, graf="b",
stt=FALSE)

Detalhes:

A função faz um breve diagnóstico de turmas, calculando as médias por aluno
(ponderada ou não),
e gerando gráficos de boxplot ou histogramas, conforme o interesse do
educador ou coordenador.
A saída apresenta os nomes dos alunos que tiveram notas baixas.
```

```
A Função oferece também a possibilidade de realizar um teste tipo ANOVA, a fim de detectar
diferenças reais entre as turmas.

Avisos:

Para que os gráficos sejam legíveis, se atenha ao máximo de 10 turmas por vez.
É necessário atenção às classes dos argumentos. A função não é capaz de detectar possíveis notas
faltantes, por isso é sempre bom averiguar o banco de notas.

Autora:

Giulia Baldaconi S. Bispo
giulia.bispo@usp.br / gi.baldaconi@gmail.com

Exemplos:

##CSV##
#Médias não ponderadas, gráfico “b” e com teste estatístico.

Por favor, salve o teste em sua pasta diretório.
teste<-read.csv("teste_diagtu.csv",sep=";", as.is=T)
turmas<-teste$Turma
nome<-teste$Alunos
notas<-data.frame(teste$P1,teste$P2,teste$Seminario,teste$P3)
prof<-"Prof.Xis"

diag.tu(nomes=nome, prof=prof, notas=notas, corte=6, turma=turmas, stt=T)

##Vetores##
#Média ponderada com histograma, de um ano inteiro.
grad<-seq(0,10, by=0.25)
altas<-seq(6,10, by=0.25)
baixas<-seq(0,7, by=0.25)
p1<-sample(grad,280,replace=T)
p2<-sample(altas,280,replace=T)
p3<-sample(grad,280,replace=T)
p4<-sample(baixas,280,replace=T)
p5<-sample(baixas,280,replace=T)
p6<-sample(altas,280,replace=T)
p7<-sample(grad,280,replace=T)
p8<-sample(altas,280,replace=T)
pfinal<-sample(altas,280,replace=T)
prec<-sample(altas,280,replace=T)
seminar<-sample(altas,280,replace=T)

notasano<-data.frame(p1,p2,p3,p4,p5,p6,p7,p8,pfinal,prec,seminar)
nomes<-rep("fulano",280)
```

```
X<-"Prof.Xis"
turma<-c(rep("1A",30), rep("1B",20), rep("1C",30), rep("1D", 20),
rep("2A",30), rep("2B",30), rep("2C",30), rep("3A",30), rep("3B",30),
rep("3C",30))
pesos<-c(1,2,1,2,1,2,1,3,4,1,1)
corte<-6

diag.tu(nomes=nomes, prof=X, notas=notasano, corte=6, pesos=pesos,
turma=turma, graf="h", stt=F)
```

Código diag.tu

```
diag.tu <- function(nomes, prof, notas, pesos=NULL, corte, turma, graf="b",
stt=FALSE)
{#Normalmente escolas tem muitos alunos, então coloquei essa primeira
mensagem pra pessoa ler enquanto o R trabalha:
  cat("\t Bem vindo! Isso pode levar alguns segundos, favor aguarde. É
sempre bom verificar se todas as notas foram colocadas em seus devidos
lugares, para que ninguem seja prejudicado ;). Caso a funcao de erros
relacionados a NAs, verifique seus dados. \n")
  #Testando as premissas dos argumentos
  if(missing(nomes)) stop("insira coluna Nomes dos Alunos")  # Caso o
usuario nao insira $nomes a funcao emite um aviso
  if(missing(prof)) stop("insira coluna Nome do Professor")  # Caso o
usuario nao insira o $prof a funcao emite um aviso
  if(missing(notas)) stop("insira coluna Notas dos Alunos")  # Caso o
usuario nao insira $$notas, a funcao emite um aviso
  if(missing(corte)) stop("insira a Nota de Corte")  # Caso o usuario nao
insira a nota considerada corte na escola, a funcao emite um aviso
  ###
  if(class(nomes)!="character")      # Caso nomes nao seja da classe correta
  {
    stop ("nomes precisa ser character")  # A funcao emite um aviso
  }
  if(class(prof)!="character")   # Caso profs nao seja da classe correta
  {
    stop ("prof precisa ser character")   # A funcao emite um aviso
  }
  if(class(notas)!="data.frame")   # Caso notas nao seja DA CLASSE CORRETA
  {
    stop ("notas precisa ser um data.frame numerico")   # A funcao emite um
aviso
  }
  if(class(corte)!="numeric")   # Caso a nota de corte nao seja DA CLASSE
CORRETA
  {
    stop ("o corte precisa ser numerico")   # A funcao emite um aviso
  }
```

```
  ### Calculo das medias
  if(is.null(pesos)) #pesos tem padrão NULL, na ausencia deles, faz media
comum:
  {
  medias<-apply(notas,1,mean) #fazendo a media simples
  }
  if(length(pesos) != 0) #caso pesos exista, ele calcula a media ponderada,
mas antes testa algumas premissas:
  {
    if(class(pesos)!="numeric")   # Caso os pesos nao sejam DA CLASSE
CORRETA
    {
      stop ("pesos precisam ser numericos")   # A funcao emite um aviso
    }
    if(dim(notas)[2]!= length(pesos)) # Caso o numero de colunas de notas
seja diferente do n de pesos
    {
      stop("Insira pesos para todas as notas, em igual ordem")   # A funcao
emite um aviso solicitando a correcao
    }
    else{} #premissas cumpridas! segue normalmente
  medias<-rep(NA, nrow(notas)) #abrindo um vetor vazio para guardar os
resultados do for de i
  umporum<-matrix(ncol=ncol(notas), nrow=nrow(notas)) #matriz vazia pra
resultados do for de j por linha
      for(i in 1:(length(medias)))
      {
        for(j in 1:nrow(notas)) #multiplicara as notas pelos pesos delas
        {
          for (m in 1:ncol(notas)) #abre for para a dimensao m da matriz por
linha
          {
              umporum[j,m]<-pesos[m]*notas[j,m] #e guarda aqui
          }
          }
      denominador<-sum(pesos) #soma os pesos
      medias[i]<-(sum(umporum[i,]))/denominador #calcula e guarda as medias
ponderadas por aluno
      }
      }
  DIAG_tu<-data.frame(nomes,turma,medias) #criando o data.frame de output
  write.csv(DIAG_tu, file="DIAG_tu.csv") #salvando os resultados das medias.
  vermelhas<-nomes[(medias<corte)==T] #alunos com medias vermelhas serao
listados no console
  write.table(vermelhas, file="ATENCAO_para.txt", sep=",") #e serao salvos
num .txt
  ##Graficos##
  if(graf=="b") #padrão boxplot
  {
    colorir<-
```

```
rgb(runif(nlevels(as.factor(turma))),runif(nlevels(as.factor(turma))),runif(
nlevels(as.factor(turma)))) #Cria um vetor de cores para as caixinhas,
aleatorios e adaptaveis ao numero de turmas que vier
    x11() #abre uma janela para o boxplot
    boxplot(medias~turma, las=1, xlab="Distribuicao de medias por turma",
ylab=" ", main=prof, col=colorir, bty="l") #gera o boxplot por turma
    savePlot(filename="DIAG_tu.png", type="png", device=dev.cur()) #salva o
grafico gerado
  }
  if(graf=="h") #histograma
  {
    fator.turma<-as.factor(turma) #transforma turma em fator, para sabermos
os níveis inseridos pelo usuário
    nro.turmas<-nlevels(as.factor(turma)) #quantos níveis tem, ou seja,
quantas turmas?
    colorir<-
rgb(runif(nlevels(as.factor(turma))),runif(nlevels(as.factor(turma))),runif(
nlevels(as.factor(turma)))) #Cria um vetor de cores para as caixinhas
    x11() #abrindo a janela para os histogramas
    if(nro.turmas<2) #se tiver so uma turma, a janela nao precisa ser
dividida
    {
      par(mfrow=c(1,1)) #um por um
    }
    else{ #se tiver mais do que 1 turma
    pares <- function(x) {x %% 2 == 0} #crio uma funcao pra descobrir se o
nro de turmas e par. Ela testa se o resto da divisão de x por 2 e = 0.
         if(pares(nro.turmas)==TRUE) #caso seja par, divido a janela desse
jeito
         {
         par(mfrow=c(2,((nro.turmas)/2))) #dividindo a janela de um jeito
BEM generico
          }
        else{par(mfrow=c(2,((nro.turmas+1)/2)))} #se for impar, divido desse
outro jeito generico
            }
    turma.tal<-rep(NA, nro.turmas) #criando um vetor vazio para os títulos
dos graficos
    for(t in 1:nlevels(as.factor(turma))) #abrindo o for que vai gerar
histogramas
      {
      for(k in 1:nro.turmas) #abrindo o for que vai gerar os titulos dos
histogramas
      {
       turma.tal[k]<-levels(fator.turma)[k]  #gerando titulos para niveis do
fator turma
      }
    hist(medias[turma==levels(fator.turma)[t]], bty="l", xlab="Distribuicao
de notas", ylab="Frequencia das notas", main=turma.tal[t],
col=sample(colorir,1)) #varios histogramas
    }
```

```
    savePlot(filename="DIAG_tu.png", type="png", device=dev.cur()) #salva o
grafico gerado
    }
  cat("\t Pode conferir sua pasta diretorio, os arquivos DIAG_tu e
ATENCAO_para ja estao salvos la! \n") #poe na tela!!
  if(stt==FALSE){return(list(DIAG_tu,vermelhas))}   #caso nao tenha stt,
retorna a lista com as medias e a lista de nomes de alunos que precisam de
uma ajuda extra
  if(stt==TRUE){ #caso tenha, a funcao gera o anova das medias por turma
    stts<-aov(medias~turma) #faz o teste
    anova(stts) #mas mostra só a FAMOSA TABELA DO ANOVA
    return(list(DIAG_tu,vermelhas,anova(stts))) #retorna medias, alunos que
precisam de ajuda e o teste!
  }
      ###
     ##
    ##
   ##
  #### Agradeco ao corpo docente, em especial aos monitores(as) e suas
ajudas inusitadas (ate no corredor)###
  #####
  }
```