



Janaina Gomes da Silva

Possui mestrado pelo Instituto de Botânica de São Paulo e é aluna do doutorado do programa de pós da Biologia Vegetal da UNICAMP. Trabalha com ecofisiologia do uso do nitrogênio em espécies arbóreas da Mata Atlântica.

exec

Trabalho Final

Proposta A

A proposta A tem como objetivo gerar uma função que pegue uma planilha com dados de resposta de alunos numa prova de alternativas e um data frame com o gabarito e a partir desses dois documentos gerasse uma lista com gráficos, dados de estatística e uma planilha com a nota e o número de acertos de cada aluno para avaliar o rendimento da classe. Os gráficos seriam um histograma com as notas da classe e outro com número de acertos e erros por questão.

Eu tive essa ideia porque ano passado dei aula numa escola que realizava uma série de simulados preparatórios para o SARESP e depois deles sempre havia uma reunião analisando o desempenho geral por turma, em que eram analisados gráficos de desempenho por matéria e desempenho geral. A parte anterior a essa reunião que era a correção e contagem de acertos e erros por questão era muito demorada e trabalhosa e talvez essa função facilitasse nessas avaliações gerais.

Proposta B

Essa proposta seria uma etapa anterior à proposta A, ela geraria o data frame que tem a tabela de respostas dos alunos a partir da imagem das fichas de resposta preenchidas pelos alunos. Essas fichas seriam células com o número das questões e cada alternativa, os alunos pintariam a alternativa que acham corretas. Então essa função seria capaz de reconhecer qual célula estaria preenchida e não preenchida pelo padrão claro/escuro presentes nas células. Elas poderiam comparadas com um gabarito correto da prova que seria uma imagem scaneada, gerando a resposta certo ou errado de acordo com a imagem ou sem essa imagem dando as letras correspondentes de resposta de cada questão, seria uma dessas duas opções.

Então as entradas seriam as imagens scaneadas, que estariam numa pasta ou no próprio diretório de trabalho, que gerariam um data frame com as letras correspondentes as células preenchidas. Nesse data frame também haveria uma coluna com o nome do arquivo que identificaria de que arquivo saiu cada imagem.

Comentários

Ola Janaina,

Apesar de que é uma tarefa bem geral e poderia ser útil para outras pessoas, achei a proposta A pouco desafiadora. Eu procuraria um problema mais biológico relacionado com sua pesquisa em ecofisiologia. Existe algum cálculo geral ou tarefa que poderia ser resolvida com uma função no R? Se depois de esgotar as possibilidades não tiver nenhuma ideia melhor, vá em frente e tente a proposta A.

Sobre a proposta B, ela é uma função que eu nunca vi ser realizada no R, apesar de que deve ser possível. Mas não faço ideia de como e acho que foge um pouco do escopo do curso. Portanto, sugiro descartar completamente.

Eu tive essa ideia da proposta B, porque tinha visto um trabalho de uma aluna do ano passado que trabalhava com uma imagem e usava o R pra fazer a contagem do número de células de cria nos favos de mel, então achei que poderia também trabalhar com leitura de uma imagem de outra forma. Como essa foi descartada, apresento outra proposta.

Proposta C

Vou analisar a presença de gases NH₃, NO e NO₂ presentes na atmosfera na minha área de trabalho e pretendo fazer a correlação dessas medidas com as condições climáticas (temperatura, umidade e precipitação) em cada estação. Para isso, farei medidas mensais, sendo várias medidas num mesmo mês (5 dias por mês) e várias medidas por dia (4 medidas por dia), esses dados seriam organizados num data frame. A partir desses dados, gostaria que a minha função fizesse a média diária e mensal e a partir da média mensal fizesse a correlação com os dados de clima gerando os resultados de correlação por estação. Então a minha entrada seria um data frame, com os dados climáticos, mês de coleta e concentração dos gases. A saída seria uma tabela com os dados da correlação por estação de cada gás.

Comentários C

Opa! Acho que melhorou bastante e este pode ser um caminho interessante. Lembre-se que a função deve ter generalidade e isto deve estar explícito na sua função e no help dela. Então pense e formule melhor qual é a tarefa “geral” que a sua função irá realizar? Transformar observações diárias em médias diárias e mensais? Este é um primeiro passo. Depois ela irá gerar análises de correlação e irá sumarizar isto em uma tabela, certo?

Outro aspecto importante é a função ter algum grau de desafio. A proposta C é factível mas me parece pouco desafiadora. Talvez incluir alguma outra tarefa importante seja interessante para deixá-la mais atraente e podermos avaliar melhor o que você aprendeu durante o curso. Por exemplo, outra saída da função poderia ser um painel de diagramas de dispersão com as correlações plotadas.

Resposta

Estou postando a minha função e na hora de criá-la vi que não era necessário calcular as médias para realizar o cálculo de correlação, mas como tinha falado que faria isso, resolvi criar um argumento e fazer o cálculo das médias a partir de um fator como um resultado separado. Coloquei a outra saída sugerida, um painel com diagramas de dispersão com as correlações plotadas.

Página de Ajuda

CorPlotM

package:unknown

R Documentation

Cálcula as correlações e gera um gráfico com os dados utilizados

Description:

A função calcula as correlações em um grupo de dados por subgrupos, gera uma figura com um painel de diagrama por dispersão de x por y e calcula as médias a partir de uma lista.

Usage:

```
CorPlotM(x, y, INDEX, na.rm=FALSE, plot=TRUE, save.plot=FALSE, media.fac="")
```

Arguments:

x	um vetor numérico ou um data frame nos quais se deseja
comparar	com cada coluna de y.
y	um vetor numérico ou data frame com número de
linhas	compatíveis com x.
INDEX	uma lista de fatores, do mesmo tamanho do número de linhas.
na.rm	lógico. Indica se NAs devem ser removidos.
plot	lógico. Se verdadeiro (TRUE), um painel com os diagramas
de	dispersão são plotados (uma janela para cada x) com
as	correlações plotadas. Se falso (FALSE), somente as
correlações	são informadas.
save.plot	lógico. Se verdadeiro (TRUE), os gráficos plotados são
salvos	no diretório de trabalho em uso em formato PNG.
media.fac	uma lista sobre como os elementos devem ser agrupados
caso	queira calcular da média dos dados por coluna.

Details:

x e y são subdivididos de acordo com os fatores presentes no argumento INDEX e a correlação é realizada em cada grupo de valores presentes por coluna de x com todas as colunas de y, sendo realizada a correlação por um y por vez.

A função `media.fac` faz a média tanto de x como de y, por coluna segundo

o agrupamento informado no formado lista, é necessário colocar `list(vetor)` para que esse calculo seja utilizado.

Ao colocar `na.rm` igual a TRUE, todos os NAs presentes serão retirados do calculo pela deleção de Casewise. Se for FALSE, quando houver dados perdidos o resultado informado será NA.

Value:

Retorna uma lista com as correlações entre x e y por INDEX. Se o plot for igual a TRUE também retorna um painel com os diagramas de dispersão e as correlações plotadas.

Se informada uma lista em `media.fac`, dentro da lista é retornada a média por agrupamento informado.

Author(s):

Janaina Gomes da Silva

Examples:

```
#criando um data frame
Estação<-rep(c("Inverno","Verão"),each=12)
Mês<-
rep(c("Junho","Julho","Agosto","Dezembro","Janeiro","Fevereiro"),each=4)
Dia<-rep(c(1,2), times=12)
set.seed(42)
Conc.NH3<-c(runif(12,1,3.8),runif(12,2,5))
set.seed(42)
Conc.NO2<-c(rnorm(12,45,10),rnorm(12,15,5))
Prep<-rep(c(25,15,12,150,200,300),each=4)
Temp<-c(rep(c(20,23),times=2),rep(c(25,21),times=2), rep(c(25,23),times=2),
rep(c(32,30),times=2),rep(c(30,28),times=2),rep(c(33,31),times=2))
Umi<-c(rep(c(30,32),times=2),rep(c(40,50),times=2), rep(c(20,27),times=2),
rep(c(50,55),times=2),rep(c(60,70),times=2),rep(c(55,60),times=2))
dados<-data.frame(Estação,Mês, Dia,Conc.NH3,Conc.NO2,Prep,Temp, Umi)
#correlação e médias sem plot
CorPlotM(dados[,4:5],dados[,6:8],dados$Estação, plot=FALSE,
media.fac=list(dados[,2],dados[,3]))
#correlação com plot e sem calculo de médias
CorPlotM(dados[,4:5],dados[,6:8],dados$Estação)
```

Código da Função

```
CorPlotM<-
function(x,y,INDEX,na.rm=FALSE,plot=TRUE,save.plot=FALSE,media.fac="")
{
  #cria objetos para usar na função
  INDEX<-INDEX
  y.names<-y
  x<-x
  xdata<-data.frame(x,INDEX)
  y<-data.frame(y,INDEX)
  #conta número de colunas
  nx=length(x)
  ny=length(y)-1
  nindex=length(unique(INDEX))
  # cria uma matriz e uma lista para colocar resultados
  resultados.cor<-matrix(NA,nindex,ny)
  resultados.tudo<-list()

  # atribui nomes para as linhas e colunas
  colnames(resultados.cor)<-names(y.names)
  rownames(resultados.cor)<-names(x)
  #cria um objeto para cores e símbolos
  cor.p<-c(2:50)
  simb.p<-c(16:25,1:15)

  for(i in 1:nindex)
  {
    #separar por grupo de interesse
    xsub<-subset(xdata,subset=xdata$INDEX==unique(INDEX)[i])
    ysub<-subset(y,subset=y$INDEX==unique(INDEX)[i])
    for(j in 1:nx)
    {#seleciona a coluna de x que será feita a cor
      x.col<-data.frame(xsub[,j],xsub$INDEX)
      for(l in 1:ny)#para calcular por coluna de y
      {
        if(na.rm==FALSE)#se for falso, quando NAs presentes
        retorna NA como resultado
        {
          resultados.cor[j,l]<-
            cor(x.col[,1],ysub[,l],use="everything",method="pearson")
        }
        if(na.rm==TRUE)#se for verdadeiro, os NAs presentes são
        removidos do calculo
        {
          resultados.cor[j,l]<-
            cor(x.col[,1],ysub[,l],use="na.or.complete",method="pearson")
        }
      }
    }
  }
}
```

```
#armazenas os resultados da matriz em uma lista
resultados.tudo[[i]]<-resultados.cor
}
media.fac<-media.fac

if(class(media.fac)=="list")
{
#cria um data frame com as colunas de x e y juntas
dataxy<-data.frame(x,y[,-(length(y))])
#conta o número de colunas e armazena num objeto
ndata<-length(dataxy)
if(na.rm==FALSE)
{
result.med<-aggregate(dataxy[,1:ndata],media.fac,mean)
}
if(na.rm==TRUE)
{
result.med<-aggregate(dataxy[,1:ndata],media.fac,mean,na.rm=TRUE)
}
}

if(plot==TRUE)#mostra o painel de dispersão
{
    if(save.plot==TRUE)#salva os gráficos no formato PNG
    {
        png(filename = "Plot%02d.png",
            bg = "transparent")
        #criação do gráfico
        for(m in 1:nx)
        {
            #definições do gráfico- geração do número de linhas e colunas
            #a partir do número de colunas de y,
            #caixa do gráfico em L, tamanho das marcas de escala,
            #proximidade da legenda xy dos eixos,
            #largura da área exterior ao gráfico,das margens, plotagens
            #somente na área do gráfico,
            #tamanho da letra nos eixos e tamanho da legenda das legendas
            #dos eixos
            par(mfrow=rev(n2mfrow(ny)),bty="l",tcl=-0.3,
                mgp=c(1.2,0.3,0),oma=c(0,0,2,0),
                mar=c(3.8,2.5,4.1,1), xpd=FALSE,
                cex.axis=0.8, cex.lab=0.8)
            #nome da coluna de x utilizada e criação do objeto X que será
            #usado no gráfico
            ylab.scp<-colnames(x[m])
            X<-x[,m]
            for(f in 1:ny)
            {
                Y<-y[,f]#criação do objeto com a coluna correspondente que
                #será utilizada no gráfico
            }
        }
    }
}
```

```

xlab.scp<-colnames(y[f])#nome da coluna y utilizada
#gráfico xy com legendas a partir dos objetos já
criados, direção da legenda ao eixo,
#definição de cores e do símbolo a ser utilizado
plot(X~Y, xlab=xlab.scp,ylab=ylab.scp,
las=1,col=cor.p[INDEX],
pch=simb.p[INDEX])
#geração das linhas de regressão por INDEX
for(ab in 1:nindex)
{
Xd<-data.frame(X,INDEX)
Yd<-data.frame(Y,INDEX)
#subdividindo de acordo com grupo de interesse
Xlm<-subset(Xd,subset=Xd$INDEX==unique(INDEX)[ab])
Ylm<-subset(Yd,subset=Yd$INDEX==unique(INDEX)[ab])
cor.l<-1+ab
abline(lm(Xlm[,1]~Ylm[,1]),col=cor.l)
}
}
#criação da legenda, objetos para cores e pontos
cor.p<-2:(1+(length(unique(INDEX))))
pnt<-15:(14+(length(unique(INDEX))))
#permite plotagem fora da área do gráfico
par(xpd=NA)
#legenda, com localização fora da área do gráfico e sem
caixa desenhada
legend("topright",inset=c(0,-0.25),
legend=unique(INDEX),
pch =pnt, col=cor.p, bty="n")
#título fora da área do gráfico
mtext("Painel com Diagramas de Dispersão",
outer=TRUE)
}
#fecha janela dos gráficos e salva as figuras no diretório de
trabalho
dev.off()
}
#criação do gráfico, como as funções PNG e X11 apresentavam
conflito, a solução foi repetir a criação do gráfico
for(m in 1:nx)
{
X11()
par(mfrow=rev(n2mfrow(ny)),bty="l",tcl=-0.3,
mgp=c(1.2,0.3,0),oma=c(0,0,2,0),
mar=c(3.8,2.5,4.1,1), xpd=FALSE,
cex.axis=0.8, cex.lab=0.8)
ylab.scp<-colnames(x[m])
X<-x[,m]
for(f in 1:ny)
{
Y<-y[,f]

```

```
xlab.scp<-colnames(y[f])
plot(X~Y, xlab=xlab.scp,ylab=ylab.scp,
las=1,col=cor.p[INDEX],
pch=simb.p[INDEX])
for(ab in 1:nindex)
{
Xd<-data.frame(X,INDEX)
Yd<-data.frame(Y,INDEX)
Xlm<-subset(Xd,subset=Xd$INDEX==unique(INDEX)[ab])
Ylm<-subset(Yd,subset=Yd$INDEX==unique(INDEX)[ab])
cor.l<-1+ab
abline(lm(Xlm[,1]~Ylm[,1]),col=cor.l)
}
}
cor.p<-2:(1+(length(unique(INDEX))))
pnt<-15:(14+(length(unique(INDEX))))
par(xpd=NA)
legend("topright",inset=c(0,-0.25),
legend=unique(INDEX),
pch =pnt, col=cor.p, bty="n")
mtext("Painel com Diagramas de Dispersão",
outer=TRUE)

}
}
#retorna parametros iniciais do par
par(mfrow=c(1,1),bty="o",tcl=-0.5,
mgp=c(3, 1, 0),
mar=c(5, 4, 4, 2), xpd=FALSE,
cex.axis=1, cex.lab=1)

#nomeia listas com os nomes das colunas de x
names(resultados.tudo)<-paste(c(unique(INDEX)[1:nindex])) )
#nomeia listas com os nomes das colunas de x
lev.INDEX<-unique(INDEX)
fac.INDEX<-factor(INDEX,levels=lev.INDEX)
names(resultados.tudo)<-paste(c(levels(fac.INDEX)[1:nindex])) )

#acrescenta nome por dados de correlação
cat("\n Dados de correlação\n")
#apresenta resultados na tela, com o calculo da média ou sem ele
if(class(media.fac)=="list")
{
list(resultados.tudo,Médias.por.fator=result.med)
}
else
{
list(resultados.tudo)
}
```

```
}
```

[corplotm.r](#)

From:

<http://labtrop.ib.usp.br/> - Laboratório de Ecologia de Florestas Tropicais

Permanent link:

http://labtrop.ib.usp.br/doku.php?id=cursos:ecor:05_curso_antigo:r2013:alunos:trabalho_final:janaina.gomes.silva:start

Last update: 2020/07/27 21:46

