Camila Cristina Avila Martins

1. Apresentação

Mestranda pelo Programa de Pós-Graduação em Genética e Biologia Evolutiva pelo Instituto de Biociências da Universidade de São Paulo (IB-USP). O título de minha tese é: “Estudo da Metilação do DNA na Hipertensão Essencial em Populações Afro-Brasileiras.”. Sou orientada pela Profª. Dr.ª Regina Mingoni-Netto.

2. Meus Exercícios

Link para a página com os meus exercícios resolvidos: Exercícios

3. Trabalho final: Criando uma função

Olá Camila,

Sou Gustavo A. Ballen, monitor encargado de dar um retorno sobre suas propostas. Vou comentar no final de cada proposta algumas coisas, indicando qual seria melhor delas, ou então sugerindo desenvolver uma outra se for o caso. Por favor entre em contato caso tenha alguma dúvida ou comentario.

Em geral acho que a proposta A deve ser fortemente reformulada e talvez possa funcionar como trabalho final da disciplina. A proposta B não é suficiente e sugiro fortemente preparar uma proposta C.

Gustavo A. Ballen

3.1 Proposta A: Diagnóstico e tratamento de doenças em peixes.

//Contextualização//

Para quem já teve algum tipo de peixe como pet fica fácil entender o porquê grande parte deles vão parar na privada antes do primeiro ano de vida: Não é que a expectativa de vida média desses animais seja tão baixa, mas sim que ser um piscicultor de primeira viagem pode ser uma tarefa mais complicada do que se imagina. Diagnosticar e tratar problemas de saúde nesses animais tende a ser mais complicado. Entender quais sãos as principais características que devem ser observadas nesses pets e, uma vez que algo estranho tenha sido detectado, decidir qual o melhor tratamento, pode ser difícil. Afim de facilitar o diagnóstico de doenças em peixes e optar pelo melhor tratamento desses, algumas empresas passaram a disponibilizar um fluxograma, como o exemplo na imagem a baixo, em que, a partir dos principais sintomas apresentados pelo animal, pode-se deduzir qual provável enfermidade ele apresenta.

O fluxograma utilizado como exemplo foi elaborado pela empresa Alcon, sendo disponibilizado na página:

http://alconpet.com.br/home.

Tendo como base esse fluxograma, a tarefa, portanto, que minha função deve executar é, a partir de dados fornecidos pelo usuário a respeito dos peixes, determinar qual a melhor intervenção e calcular a dosagem do medicamento que deve ser aplicada, tendo em vista que cada intervenção requer uma dosagem distinta.

//Planejamento da função//

Entrada e ajuste de parâmetros:

Nome da função:
fish.disease ()

• Entrada de h,altura do aquário.
h ← readline(“Qual a altura do aquário em cm? (ex.:30): ”)
• h é um número maior do que zero? Enquanto a antrada não for um número maior do que zero, pede uma nova entrada.

• Entrada de L, largura do aquário.
L ← readline(“Qual a largura do aquário em cm? (ex.:20): ”)
• L é um número maior do que zero? Enquanto a antrada não for um número maior do que zero, pede uma nova entrada.

• Entrada de comp, comprimento do aquário.
comp ← readline(“Qual o comprimento do aquário em cm? (ex.:40): ”)
• comp é um número maior do que zero? Enquanto a antrada não for um número maior do que zero,

pede uma nova entrada.

Pseudo-código:

1. Cálculo do volume do aquário em litros: (h*L*comp)/1000.
#Alcali, Acid, Ictio, Aqualife e Bacter são diferentes tipos de medicamentos.

#Transposição do fluxograma em código:

2. A transposição do fluxograma em código será feita através da função `readline`que interage com o usuário, como no exemplo a seguir, e por meio de estruturas com `if`s e `else`s. Desse modo, os dados dos sintomas são introduzidos pelo usuário ao digitar sim (s) ou não (n) para as respostas.

```
      resp1 <- readline("O peixe apresenta manchas ou estruturas anormais na
superficie do corpo (s/n):  ")
```

Assim, o código será elaborado por meio da estrutura:

- resp1 ← readline(O peixe apresenta manchas ou estruturas anormais na superficie do corpo?)

Testa se a resposta armazenada em resp1 é válida. Se não for, refaz a pergunta.

• Se resposta armazenada em resp1 for s:

- resp2 ← readline(Estas estruturas são pontos visíveis na pele?) (primeira versão)

• Se a resposta armazenada em resp1 for não (n):

- resp2 ← readline(O peixe apresenta natação irregular e tremores?) (segunda versão)

Depois para primeira versão de resp2:

Testa se a resposta armazenada em resp2 é válida. Se não for, refaz a pergunta.

• Se resposta armazenada na primeira versão de resp2 for s:

- resp3 ← readline(São pequenos pontos brancos espalhados por todo o corpo e nadadeiras?)

• Se a resposta armazenada na primeira versão de resp2 for não (n):

- resp3 ← readline(Há uma crosta aparentando limo ou manchas como mofo?)

O mesmo é feito para a segunda versão de resp2, resp3... e assim por diante, até finalizar as possibilidade de caminhos dentro do fluxograma.

Ao final de cada caminho dentro do fluxograma há um diagnóstico.

3. Ao final das estruturas de `if`s e `else`s para cada um dos diagnóstico é feito o cálculo de medicamento que deve ser utilizado e informado o tratamento:

- Cálculo da dosagem do tratamento com Alcali ou Acid: 1 gota para 3 litros.
- Cálculo da dosagem do tratamento com Ictio ou Aqualife: 1 gota para 2 litros.

- Cálculo da dosagem do tratamento com Bacter: 1 cápsula para cada 12,5 litros.
- Cálculo da dosagem do tratamento com Ictio e Aqualife: 1 gota de aqualife para para 4 litros e 1 gota de Ictio para 2 litros

(como há interação medicamentosa, a dosagem de cada medicamento varia quando usada em conjunto).

4. Dado que nenhuma configuração de tratamento se encaixe, o usuário recebe 2 opções: refazer o teste ou não.
Se a resposta for sim, o teste recomeça, se a resposta for não, a função termina.

Saída

• Retorna a informação da provável doença e tratamento.
Ex.:

```
     "Doença dos Pontos Brancos"
Seu(s) peixe(s) apresenta(m) Doença dos pontos Brancos ou Ictio.
Ela é causada pelo protozoário Ichthyophthyrius multifiliis.
Outros sintomas: Nadadeiras fechadas, peixe se esfregando nas pedras
e no fundo do aquário, dificuldade para respirar.
Tratamento: Adicione  12  gota(s) de Ictio na água do aquário.
```

—-

> — *Alexandre Adalardo de Oliveira 2019/06/18 11:22* Oi Camila,
>
> Veja as sugestoes do Gustavo abaixo. Além delas, minha sugestão é que utilize a função ``readline``para interagir com o usuário ao invés de entrar os dados dos sintomas. Daria para refazer todos o fluxograma, com perguntas ao usuário e guardando as informações em objetos nas função. Por exemplo:
>
> ```
> resp1 <- readline("O peixe apresenta manchas
> ou estruturas anormais na superficie do corpo
> (s/n):  ")
> ```
>
> A linha de comando acima guarda a resposta do usuário da primeira pergunta do fluxograma que apresentou no objeto ``resp1``. A partir dessa interação e das resposta, pode refazer o fluxograma da figura, chegando ao diagnóstico. Inclua isso e retorne ao usuário o diagnóstico e o tratamento. Isso já é um bom desafio, deixe o cálculo das dosagens do tratamento como extra, caso consiga implementar o fluxograma. Aguardamos a reformulação o quanto antes

para podermos fechar a proposta para você iniciar os trabalhos.

Proposta A

Não fica claro exatamente quais dados serão fornecidos pelo usuário, em especial quando colocado no contexto da justificativa de função. Neste ponto e depois da disciplina acredito que seja possível ser mais especifico sobre o modo (mode) e a clase (class) de tais dados. Para resumir, é necessário colocar exatamente que dados o usuário irá fornecer.

Por quê um data frame? Se estamos com uma lista simples de sintomas bem poderiamos usar um vetor de modo “character”. Os data frames tendem a ser mais úteis em outros contextos (e.g., para duas dimensões e várias variáveis associadas ao mesmo registro, e também de modos diferentes). É importante justificar a escolha das estruturas de dados para nortear melhor o desenvolvimento da função. Se bem a estrutura do objeto a inserir como argumento sintomas é de fato de duas colunas, não precisa ser, como foi indicado anteriormente.

Como a dosagem irá ser calculada? Essa informação é relevante para avaiar o grau de complexidade da função. Calcular a dosagem para todos os tratamentos (ítens 2-6 e potencialmente os 7-9) sendo que na maior parte dos casos só um ou dois são necessários é um desperdiço de esforço computacional.

O item 10 seriam 19 pares if-else. Acho que daria para simplificar bastante a escolha e eventualmente produzir um procedimento mais elegante para identificar o resultado desejado. Me vem à mente operações vetorizadas como uma possível alternativa a tanto if-else.

As verificações poderiam ser desnecessárias quando reavailada a informação que o usuário ira submeter como argumento.

A saída, se bem usa o poder das listas como estrutura de armazanamento de dados, não indica em detalhe qual tipo de informação irá ser retornada. Qual informação iria em “tratamento”?

A função tem quatro argumentos, sendo que três deles são para calcular um volume. Acredito que isso seja um desperdiço de espaço toda vez que o usuário bem poderia colocar como input o volume do aquário. Se o usuário conseguir abrir o R, ele pode saber usar uma calculadora, ou mesmo o R para calcular isso pra ele.

Dadas as dúvidas ainda acho que precisa especificar melhor a proposta, pois ela tem o potencial de ser um desafio ou uma coisa simples demais para o trabalho final da disciplina, dependendo da abordagem. Leve em consideração que se a função vira um procedimento “calculadora” no qual o input é X, Y, e Z, e o procedimento é A = X + Y - Z, produzindo uma saída A, então nem precisamos muito de uma função para aquilo, é só usar os valores numa sessão interativa e usar o R como calculadora. Isso é particularmente relevante para a parte onde o volume do aquário é calculado. Tirando esse calculo simples, a função basicamente segue uma sequência de pares if-else, que desaproveita o enorme arsenal de ferramentas computacionais que foram apresentadas na disciplina.

> Por favor considerar ou reformular esta proposta numa maneira desafiadora ou ainda apresentar uma proposta C se achar que a proposta A não tem muito potencial.

3.2 Proposta B: Aprovado, em recuperação ou reprovado.

//Contextualização//

Ser professor é uma tarefa que demanda tempo e esforço, de modo que gastar parte dessas propriedades para contabilizar as notas dos alunos e decidir quais deles serão aprovados, reprovados ou ficarão em recuperação pode ser um mau uso das mesmas. Tendo isso em vista, a tarefa que essa função deve executar é, a partir, de um data.frame contendo os nomes dos alunos e as respectivas notas para cada critério da aprovação, retornar uma lista contendo um data.frame com as médias ponderadas de cada aluno, quais deles foram aprovados, reprovados, ao ficarão em recuperação, a menor e maior nota e a média geral da classe.

//Planejamento da função//

Entrada:

students.status (`notas`, `min.aprov`, `min.rec`)
• `notas` = um data.frame com dados das notas dos alunos (a primeira coluna com todos os nomes dos alunos e as demais com as notas dos alunos para cada critério de aprovação).
Segue o exemplo:

alunos	ex1	ex2	ex3	ex...
aluno1	1.0	10.0	7.0	...
aluno2	2.0	3.9	9.7	...
aluno3	4.0	5.0	8.0	...
aluno4	7.5	2.0	7.0	...

• `min.aprov` = valor mínimo para aprovação de um aluno.
• `min.rec` = valor mínimo para que o aluno fique em recuperação e não seja automaticamente reprovado.

Verificando, ajustando e entrando parâmetros:

• `notas` é um data.frame? Se não, escreve: `notas` precisa ser um data.frame.
• A classe das colunas do data.frame `notas`, de 2 em diante, é numérica? Se não, escreve: coluna x do data.frame `notas` precisa ser da classe numérica.
• `min.aprov` é um número igual ou maior do que zero? Se não, escreve: o valor mínimo para aprovação (`min.aprov`) precisa ser um número inteiro e igual ou maior do que zero
• `min.rec` é um número igual ou maior do que zero? Se não, escreve: o valor mínimo para ficar em recuperação (`min.rec`) precisa ser um número inteiro e igual ou maior do que zero.

• O valor mínimo para recuperação é menor do que o valor mínimo para aprovação? Se não, escreve: o valor mínimo para ficar em recuperação (`min.rec`) precisa ser menor do que o valor mínimo para aprovação (`min.aprov`).
• Número de colunas – 1 é igual ao número de critérios para aprovação: `n.ex` ← ncol (notas)-1.
• Número de linhas é igual ao número de alunos: `n.alunos` ← nrow (notas).
• Inserção dos pesos para cada critério de aprovação:
`crit.prov` ← as.numeric (readline ("insira o peso para o critério de aprovação 1. Ex.: 2.0:")).
Isso será feito para cada um dos critérios, de modo que será necessário o uso de um contador.

Pseudo-código:

1. Criação de vetores contendo as notas dos alunos multiplicadas pelo peso para cada critério de aprovação, por exemplo:
Criação de um vetor `nota.crit1` contendo os valores das notas do critério 1 para todos os alunos * peso para o critério 1.
Isso será feito para todos os critérios.

2. Somas das notas para cada critério multiplicadas pelos pesos para cada um dos critérios para cada aluno:
`soma.nota_peso` ← valor na posição 1 do vetor `nota.crit1` + valor na posição 1 do vetor `nota.crit2` + ...+ valor na posição 1 do vetor `nota.crit`(n.ex).
3. Soma dos pesos para cada critério de aprovação: `soma.pesos`.
4. Média ponderada para cada aluno: `ponderada` = `soma.nota_peso`/ `soma.pesos`.
5. Inserção de uma nova coluna no data.frame notas contendo as notas ponderadas dos alunos.
6. Cálculo da média das médias ponderadas dos alunos: mean (`ponderada`).
7. Cálculo do menor valor de média ponderada dos alunos: min(`ponderada`).
8. Cálculo do maior valor de média ponderada dos alunos: max (`ponderada`).
9. Inserção de uma nova coluna no data.frame `notas` com título "Aprovados".
10. Inserção de uma nova coluna no data.frame `notas` com título "Recuperação".
11. Inserção de uma nova coluna no data.frame `notas` com título "Reprovados".
12. Classificação de cada aluno:
a. Se a nota ponderada do aluno x é maior ou igual a `min.aprov`, adiciona S na coluna Aprovados na linha do aluno x.
b. Se a nota ponderada do aluno x é maior ou igual a `min.rec`, e menor do que `min.aprov` adiciona S na coluna "Recuperação" na linha do aluno x.
c. Se a nota ponderada do aluno x é menor do que `min.rec`, adiciona S na coluna "Reprovado" na linha do aluno x.

Saída

• Retorna uma lista contendo:
a. Data.frame `notas` com colunas para as notas ponderadas dos alunos e as colunas de aprovação, recuperação e reprovação assinaladas com S conforme a situação de cada aluno;
b. Menor e maior nota ponderada e a média geral da classe.

Proposta B

Esta proposta é uma serie de calculos um atrás do outro que bem poderiam ser descritos como uma calculadora sofisticada. Não condiz com o objetivo da proposta de função de ser um desafio que aproveia a grande quantidade de ferramentas computacionais que foram apresentadas. É necessário reformular a proposta A e ainda considerar a opção de desenvolver uma proposta C.

3.3 Links para o trabalho final:

Arquivo da minha função: diagnostico_e_tratamento_de_doencas_em_peixesf.r
Link para a página da minha função: Proposta A: Diagnóstico e tratamento de doenças em peixes
Link para a página de ajuda da minha função: Página de ajuda da função