Tutorial FlexDashboards-Jornada da Ciencia Aberta

Profa. Dra. Elisangela Lizzi(UTFPR)

Tutorial: Como Criar um Dashboard Interativo no R com Flexdashboard

Introdução

Este tutorial vai guiar você passo a passo na criação de um dashboard interativo para análise de dados clínicos usando R e flexdashboard. O dashboard foi desenvolvido para a Jornada da Ciência Aberta e mostra dados do AEROBICOVID. Este material fornece uma excelente introdução à criação de dashboards interativos em R, cobrindo desde manipulação básica de dados até visualizações e modelagem estatística.

Pré-requisitos

- R e RStudio instalados
- Pacotes necessários: flexdashboard, tidyverse, gt, ggplot2, plotly, broom, DT

Passo 1: Configuração Inicial

1.1 Estrutura Básica do Arquivo

Crie um novo arquivo R Markdown (.Rmd) e cole o cabeçalho:

```
---
title: "Jornada da Ciencia Aberta Dashboard – AEROBICOVID"
output:
  flexdashboard::flex_dashboard:
    orientation: rows
    vertical_layout: scroll
---
```

Explicação:

title: Define o título do dashboard

output: Especifica que usaremos o formato flexdashboard

orientation: rows: Organiza o dashboard em linhas

vertical_layout: scroll: Permite rolagem vertical quando o conteúdo for longo

1.2 Carregar Pacotes e Dados

No primeiro chunk R:

```
library(flexdashboard)
library(tidyverse)
library(gt)
library(ggplot2)
library(plotly)
library(broom)
library(DT)

# Definir diretório de trabalho
setwd("/caminho/para/seu/diretorio")

# Carregar dados
dados <- read.csv2("Dados.csv")

# Ajustar tipos de variáveis
dados$Sexo <- factor(dados$Sex, labels = c("Masculino", "Feminino"))
dados$Classificacao <- as.factor(dados$Classific_AEROBICOVID_grav)
```

Explicação:

Pacotes usados flexdashboard: Para criar dashboards interativos tidyverse: Conjunto de pacotes para manipulação e visualização de dados gt: Para criar tabelas bonitas e formatadas ggplot2: Para criação de gráficos plotly: Para tornar gráficos interativos broom: Para organizar saídas de modelos estatísticos DT: Para criar tabelas interativas

setwd(): Define o diretório de trabalho read.csv2(): Lê arquivos CSV com separador ; (comum em alguns países)

head(): Mostra as primeiras linhas dos dados

names(): Mostra os nomes das colunas

factor(): Converte variáveis categóricas em fatores

as.factor(): Converte uma coluna em fator (variável categórica)

Passo 2: Criando o Cabeçalho do Dashboard

Adicione após o chunk de setup:

```
# Painel interativo do AEROBICOVID -GEEPESC-EEFERP-USP

Este painel é um produto da **Palestra sobre software replicável**, produzido para Jo
rnada da Ciência Aberta com abordagem didático-pedagógica, para uso do programa R par
a gerar relatórios automatizados com informações de pesquisa e visualização de dados.
```

Passo 3: Primeira Seção - Visão Geral

3.1 Medidor de Frequência Cardíaca

```
max_freq <- max(dados$HRpeak, na.rm = TRUE)

gauge(max_freq, min = 50, max = 200, label = "Máx FC (bpm)",
      gaugeSectors(success = c(50, 120), warning = c(120, 160), danger = c(160, 20
0)))
```

3.2 Contagem por Sexo

```
# Contagem por sexo
sexo_counts <- dados %>% count(Sexo)
masculino <- as.numeric(sexo_counts$n[sexo_counts$Sexo == "Masculino"])
feminino <- as.numeric(sexo_counts$n[sexo_counts$Sexo == "Feminino"])

# ValueBox para Masculino
valueBox(masculino, caption = "Participantes Masculinos", icon = "fa-mars", color =
"primary")

# ValueBox para Feminino
valueBox(feminino, caption = "Participantes Femininos", icon = "fa-venus", color = "i
nfo")
```

Passo 4: Visualização das Categorias AEROBICOVID

4.1 Gráfico de Barras Interativo

```
dados_plot <- dados %>% count(Classificacao)

grafico <- ggplot(dados_plot, aes(x = Classificacao, y = n)) +
  geom_bar(stat = "identity", fill = "gray80", color = "black") +
  labs(x = "Categoria AEROBICOVID", y = "Frequência") +
  theme_minimal()

ggplotly(grafico)
```

Explicação:

ggplot(): Inicia um gráfico ggplot2

geom_bar(): Cria um gráfico de barras

ggplotly(): Converte um gráfico ggplot em interativo (plotly)

4.2 Tabela de Classificação

```
dados3 <- data.frame(
  Classif = c(1, 2, 3, 4),
  Contagem = c(16, 50, 10, 8)
) %>%
  mutate(
    Gravidade = case_when(
      Classif == 1 ~ "Leve",
      Classif == 2 ~ "Moderada",
      Classif == 3 ~ "Grave",
      Classif == 4 ~ "Crítico"
    ),
    Percentual = round(Contagem / sum(Contagem) * 100, 1)
  ) %>%
  select(Gravidade, Contagem, Percentual)

tabela <- dados3 %>%
  gt() %>%
  tab_header(title = "Classificação AEROBICOVID por Gravidade") %>%
  cols_label(Gravidade = "Gravidade", Contagem = "Casos (n)", Percentual = "%") %>%
  fmt_number(columns = c(Contagem, Percentual), decimals = c(0, 1))
```

Explicação:

gt(): Inicia uma tabela formatada

tab_header(): Adiciona título e subtítulo

cols_label(): Renomeia colunas

fmt_number(): Formata números com casas decimais específicas

Passo 5: Análise de VO2 e BMI

5.1 Boxplot de VO2 por Sexo

```
grafico_box <- ggplot(dados, aes(x = Sexo, y = Relative.VO2)) +
  geom_boxplot(fill = "gray80", color = "black") +
  labs(title = "VO2 Relativo por Sexo", x = "Sexo", y = "VO2 (ml/kg/min)") +
  theme_minimal()

ggplotly(grafico_box)
```

Explicação:

geom_boxplot(): Cria um boxplot

notch = TRUE: Adiciona entalhes para comparação de medianas

5.2 Gráfico de Dispersão Idade vs BMI

```
ggplot(dados, aes(x = Age, y = BMI, color = Classificacao)) +
  geom_point(size = 3, alpha = 0.7) +
  labs(title = "Idade vs BMI por Classificação") +
  theme_minimal()
```

Explicação:

cor.test(): Calcula teste de correlação

geom_point(): Cria gráfico de dispersão

alpha: Define transparência dos pontos

Passo 6: Tabela Interativa Completa

```
datatable(
  dados,
  options = list(
    scrollX = TRUE,
    scrollY = "500px",
    scrollCollapse = TRUE,
    paging = FALSE,
    dom = 't'
  ),
  rownames = FALSE,
  caption = "Dados completos para consulta e download"
)
```

Explicação:

datatable(): Cria tabela interativa (DataTables)

scrollX/Y: Permite rolagem horizontal/vertical

paging = FALSE: Desativa paginação

Passo 7: Modelagem Estatística

7.1 Modelo Linear

```
modelo <- glm(Relative.VO2 ~ factor(Classificacao) + BMI + factor(Sexo) + Age,
              data = dados, family = gaussian())

resumo <- tidy(modelo, conf.int = TRUE)

resumo %>%
  gt() %>%
  tab_header(title = "Modelo Linear Generalizado") %>%
  fmt_number(columns = c(estimate, std.error, statistic, p.value, conf.low, conf.hig
h),
             decimals = 2)
```

7.2 Análise de Resíduos

```
# Histograma
residuos <- residuals(modelo)
ggplot(data.frame(residuos), aes(x = residuos)) +
  geom_histogram(fill = "gray80", color = "black", bins = 20) +
  labs(title = "Histograma dos Resíduos")

# QQ-plot
qqnorm(residuos, main = "QQ-plot dos Resíduos")
qqline(residuos, col = "red")
```

Explicação:

glm(): Ajusta um modelo linear generalizado

tidy(): Organiza resultados do modelo em tabela

geom_histogram(): Cria histograma dos resíduos

qqnorm()/qqline(): Cria QQ-plot para verificar normalidade

Dicas Finais

1. **Organização**: Use `## Row {data-height=XXX}` para controlar o espaço das seções
2. **Interatividade**: Gráficos com `ggplotly()` permitem zoom e tooltips
3. **Formatação**: Use HTML/CSS para textos centralizados e destacados
4. **Exportação**: Clique em “Knit” para gerar o HTML final

Este dashboard combina: - Visualizações interativas - Tabelas formatadas profissionalmente - Análise estatística - Dados completos para download

Pronto para personalizar com seus próprios dados!