Como testar a validade dos tipos mistos criados utilizando o R

O mesmo teste feito acima no programa Stata pode ser executado no R. Basta seguir o *script* abaixo, criado por Wesley H. S. Pereira, do Departamento de Estatística da UFMG (email: wesleyhspereira@gmail.com). Estamos assumindo que temos os mesmos bancos de dados salvos em Excel: "teste_gam.xls" e "teste_cel.xls". Esse *script* do R já faz todas as operações necessárias para criar um banco de dados (*data.frame*) apto para o teste de médias comparando todos os 10 perfis criados, conforme a Tabela 2: 1) lê os arquivos em Excel e os salva como *data.frame* no R, 2) Faz a junção dos bancos de dados "teste_cel" com "teste_gam" e cria um novo banco, "teste_celgam".

```
# LIMPANDO A WORKSPACE #
rm(list=ls(all=TRUE))

# INDICANDO O CAMINHO #
setwd("C:\\gom\\")
getwd() # Verificando o diretório

# IMPORTANDO OS DADOS PARA O R #

# Instalando o pacote xlsx
install.packages("xlsx")
# Carregando o pacote xlsx
library(xlsx)

# Importando os arquivos gam e cel
teste_gam <- read.xlsx("teste_gam.xlsx",
                       sheetIndex=1)
teste_cel <- read.xlsx("teste_cel.xlsx",
                       sheetIndex=1)

# Juntando os bancos de dados
teste_celgam <- merge(x=teste_cel,
                      y=teste_gam,
                      by="idv_gom",
                      all.x=TRUE)
```

O *script* a seguir apresenta a implementação de uma rotina otimizada para serem comparadas as 45 combinações possíveis de perfis, 2 a 2, para cada cada uma das 28 variáveis, através de um teste de média para amostras independentes, com variância desconhecida.

```
# Rotina para Teste de Médias entre Perfis #

construir.t.test = function(dados,mu = 0){
 t = sort(unique(dados$perfil))
 n = length(t)
 name = list()
 for (i in 1:n){
     name2 = list()
```

```
    c = 1
    for (j in 1:n){
         name2[[c]] = paste(paste(paste("dados_",
t[i],sep=""),"_",sep=""),t[j],sep="")
         assign(name2[[c]],dados[dados$perfil == t[i] | dados$perfil
== t[j],])
         c = c + 1
    }
    name[[i]] = name2
}

# Mudança no nome das variáveis #

v29 = list()
for (i in 1:n){
    v = list()
    for (j in 1:n){
         k = get(name[[i]][[j]])
         v[[j]] = paste(paste(paste(paste(paste(paste(
            paste("Perfil_",
t[i],sep=""),"_",sep=""),"x",sep=""),"_",
            sep = ""),"Perfil",sep = ""),"_",sep = ""),t[j],sep = "")
         names(k)[29] = v[[j]]
         assign(name[[i]][[j]],k)
    }
    v29[[i]] = v
}

# Criação das fórmulas #

fmla = list()
for (i in 1:n){
    fmla2 = list()
    for (j in 1:n){
         fmla3 = list()
         for (k in 1:(dim(dados)[2]-1)){
              fmla3[[k]] = as.formula(paste(paste(paste("V",k,sep="")
              ,"~",sep = " "),v29[[i]][[j]],sep = " "))
         }
         fmla2[[j]] = fmla3
    }
    fmla[[i]] = fmla2
}

#### Fazendo os Testes # Confere

lista = list()
for (i in 1:n){
    perfis = list()
    for (j in 1:n){
         perfis2 = list()
         for(k in 1:(dim(dados)[2]-1)){
```

```
            if( i != j){
                perfis2[[k]] = t.test(fmla[[i]][[j]][[k]],
                data = get(name[[i]][[j]]),mu)
            }
        }
        perfis[[j]] = perfis2
    }
    lista[[i]] = perfis
}
return(lista)
}
```

O *script* acima implementou a função, denominada "construir.t.test()". Essa função possui os seguintes argumentos:

- dados = conjunto de dados a ser utilizado (dataframe) contendo, na última coluna, uma variável com o nome "perfil", a qual conterá n perfis. No nosso exemplo, o dataframe test_celgam contém a variável perfil com 10 perfis (0 a 9) conforme Tabela 2. Todas as demais colunas devem ser consideradas como variáveis contínuas a serem comparadas em suas médias entre duplas de perfis distintos.
- mu = esse é o valor teórico, sob H_0, para o teste de diferença de médias. O padrão é utilizarmos 0, assumindo que a média de uma variável entre dois perfis é idêntica em seus valores populacionais.

Vamos agora a um exemplo. Imagine que queiramos criar uma lista com todos os testes de comparação de médias possíveis, entre duplas de perfis. Se temos 28 variáveis e 10 perfis, isso gerará 1260 testes de diferenças de média. A função "construir.t.test()" foi feita de modo a gerar 2520 testes, pois ele assume a comparação, por exemplo, do Perfil 1 com Perfil 2, ou do Perfil 2 com o Perfil 1. O resultado tem que ser exatamente o mesmo.

Para criar a lista com os testes, execute no R:

```
lista = construir.t.test(test_celgam)
```

Caso queira ver o resultado diretamente da lista, você pode digitar:

```
lista[[3]][[2]][23]
```

ou

```
lista[[2]][[3]][23]
```

Ou seja, o comando acima vai mostar a comparação entre o segundo e terceiro nível da variável Perfil (ou seja, Perfil 1 x Perfil 2, pois a variável "perfil" começa no valor 0 neste exemplo, conforme Tabela 2), para a variável 23. É importante lembrar que [23] corresponde à posição da variável v23. Se essa variável estivesse na posição 20 do banco de dados, o comando deveria ser:

```
lista[[3]][[2]][20]
```

ou

```
lista[[2]][[3]][20]
```

A próxima função, “explore.t.test()”, foi criada para facilitar a busca de resultados na lista gerada. Digite a função abaixo no R:

```
explore.t.test = function(objeto,perfil1,perfil2,var,dados){
 t = sort(unique(dados$perfil))
 if(!perfil1%in%t | !perfil2%in%t){
     return("Erro: Perfil não existente")
 }
 else{
     objeto[[match(perfil1,t)]][[match(perfil2,t)]][var]
 }
}
```

A função “explore.t.test()” possui 5 argumentos, quais sejam:

- objeto: a lista de testes de diferença de média criados acima, com a função “construir.t.test()”. No nosso caso, chamamos essa lista de “lista”
- perfil1 = o primeiro perfil que se quer comparar
- perfil2 = o Segundo perfil que se quer comparar
- var = a posição das variáveis que utilizaremos para comparar suas médias entre os perfis. Por exemplo, se a variável v23 estiver na posição 20 do banco de dados, então se deve usar 20, e não 23. Caso se queira usar um conjunto de variáveis simultaneamente, pode-se usar o símbolo “:”. Por exemplo, se queremos os testes para as variáveis 23 a 28, e a variável 23 está na posição 20, então poderíamos escrever var = 20:25, pressupondo que as variáveis 24 a 28 estão em posições consecutivas à variável 23.
- dados = o conjunto de dados a ser utilizado (dataframe) contendo, na última coluna, uma variável com o nome “perfil”, a qual conterá n perfis. No nosso exemplo, o dataframe test_celgam contém a variável perfil com 10 perfis (0 a 9) conforme Tabela 2. Todas as demais colunas devem ser consideradas como variáveis contínuas a serem comparadas em suas médias entre duplas de perfis distintos.

Para saber se as variáveis 23 a 28 têm médias distintas entre os perfis 1 e 5, assumindo que as variáveis 23 a 28 estão nas posições 23 a 28 do *dataframe*, execute no R:

```
explore.t.test(objeto = lista,
               perfil1 = 1,
               perfil2 = 5,
               var = 23:28,
               dados = teste_gamcel)
```

Veja a diferença de se utilizar a lista. Se usarmos diretamente a lista, deveriamos digitar:

```
lista[[2]][[6]][23:28]
```

Para fins puramente ilustrativos, vamos separar o *script* em partes. Vamos também simular variáveis contínuas e binárias para fazer os testes de média e de proporções, respectivamente, e mostrar como seria exatamente o resultado obtido. Como são dados simulados, eles não terão nenhuma interpretação prática. Vamos começar com a simulação de variáveis contínuas. Estamos simulando nesse caso uma amostra de 1000 observações, em que as variáveis contínuas "var1" até "var28" seguem distribuições normais, com pequenas diferenças de média e desvio-padrão entre elas. Essa primeira rotina criará o banco de dados simulado com as variáveis contínuas.

```
# LIMPANDO A WORKSPACE #
rm(list=ls(all=TRUE))

# CRIANDO VARIÁVEIS CONTÍNUAS SIMULADAS #
dados_sim <- matrix(ncol=28,nrow=1000)

for (i in 1:28) {
  dados_sim[,i] <- rnorm(1:1000,5+i/100,3+i/100)
}

dados_sim <- as.data.frame(dados)
dados_sim$perfil <- sample(c(0:9),1000,replace=TRUE)

# Dando nome às colunas do banco de dados
names(dados_sim) <- c("var1", "var2", "var3", "var4", "var5", "var6",
"var7", "var8", "var9", "var10", "var11", "var12", "var13", "var14",
"var15", "var16", "var17", "var18", "var19", "var20", "var21",
"var22", "var23", "var24", "var25", "var26", "var27", "var28",
"Perfil")
```

Uma vez criado o banco de dados com 28 variáveis contínuas e uma variável representando os perfis (tipos puros e mistos), executamos o *script* similar ao que foi apresentado acima, de modo que possamos efetuar os testes de média. Não precisamos refazer a rotina, pois as duas funções já foram criadas acima, quais sejam, "construir.t.test()" e "explore.t.test()". Vamos começar criando a lista com todos os testes de média entre as 28 variáveis contínuas comparadas entre todas as possíveis combinações de perfis, 2 a 2, como fizemos antes.

```
# Construindo a lista de testes de médias #

lista_sim = construir.t.test(dados_sim)
```

Suponhamos que estamos interessados em saber quais variáveis diferem significativamente entre o Perfil 1 e o Tipo Misto 12. Com a função "explore.t.test()" essa tarefa é simples, bastando informar os perfis como argumento e a amplitude de variáveis (variável 1 a variável 28).

```
# Explorando as diferenças de médias entre Perfil 1 e TM12 #

explore.t.test(objeto = lista_sim,
               perfil1 = 1,
               perfil2 = 2,
```

```
                var = 1:28,
                dados = dados_sim)
```

Caso queira apenas o teste para apenas uma variável, por exemplo comparando os perfis 1 e 5 da variável 23, digitar:

```
explore.t.test(objeto = lista_sim,
                perfil1 = 1,
                perfil2 = 5,
                var = 23,
                dados = dados_sim)
```

Pela lista diretamente, o mesmo resultado poderia ser obtido com o seguinte comando:

```
lista[[2]][[6]][23]
```

Nesse caso, o *output* do R traria algo parecido com:

```
      Welch Two Sample t-test

data:  V23 by Perfil_1_x_Perfil_5
t = 0.67959, df = 203.32,
p-value = 0.4975
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.6046697  1.2407301
sample estimates:
mean of x mean of y
 5.539681  5.221651
```

Nesse *output* fica claro que estamos testando o Perfil 1 contra o Perfil 5 em termos de diferença na media da variável "var23". A hipótese nula é que a média da variável "var23" é a mesma para o Perfil 1 (PE1) e o Perfil 5 (TM21). O resultado, neste caso, mostra que a estatística de teste, t = 0.67959, não foi maior que o valor de t na Tabela t de Student com 203.32 graus de liberdade. Portanto, o p-valor = 0.4975 (> 0.05) sugere que não temos evidência estatística suficiente para rejeitar a hipótese nula. Portanto, não podemos afirmar, com base nos dados, que haja diferença significativa entre a média de "var23" entre esses dois perfis. Dito de outra forma, não rejeitamos a hipótese de que o verdadeiro valor médio de "var23" é idêntico entre os perfis PE1 e TM21. O teste também traz o interval de confiança para a diferença de medias, e mostra o valor das medias para a "var23" no Perfil PE1 (5.54) e no Perfil TM21 (5.22).

Veja que estamos assumindo que as variáveis "var1" até "var28" são contínuas. Esse caso se aplicaria quando as variáveis originais são contínuas, mas o pesquisador teve que transformar essas variáveis em categóricas para executar o GoM. Como as variáveis originais são contínuas, seria fácil testar a diferença entre os perfis para cada variável voltando em seus valores originais, na escala continua. No caso de todas serem variáveis binárias, por natureza, devemos executar um teste para proporções. Vamos aqui dar um exemplo mais resumido.

Vamos começar simulando um banco de dados com valores binários para várias variáveis. Portanto, basta rodar o seguinte *script*:

```
# Simulando Banco de Dados com Variáveis Binárias #

set.seed(1234)
dados_bin <- matrix(ncol=28,nrow=1000)

for (i in 1:28) {
  dados_bin[,i] <- rbinom(1000,1,0.3+i/100)
}

dados_bin <- as.data.frame(dados_bin)
dados_bin$perfil <- sample(c(1:5),1000,replace=TRUE)
```

Com o banco criado, podemos agora fazer os testes de proporção que quisermos. Imagine que queiramos comparar as diferenças de proporções para as variáveis binárias "var20", "var21", "var22", "var23", "var24", e "var25" entre os perfis 1 e 2. Nesse caso, podemos usar a função "for () {}" do R (*loop*) para facilitar a repetição de testes. Basta executar o seguinte *script*:

```
lista_bin=list()
c=23
for (i in 20:25) {
lista_bin[[c]]=prop.test(table(dados_bin[,i],dados_bin$perfil)
 [,c(1,2)],
          alternative='two.sided',
          conf.level=0.95,
          correct=FALSE)
   c=c+1
}
```

Para mandar visualizar todos os testes, os quais armazenamos em uma lista denominada "lista_bin", basta digitar o nome da lista criada diretamente no console:

```
lista_bin
```

Caso queiramos acessar especificamente o teste da variável 23 entre os perfis 1 e 2, basta digitarmos:

```
lista_bin[[23]]
```

Se o objetivo é verificar o resultado dos testes numa sequência de variáveis, por exemplo da variável 23 até a variável 25, comparando aqui os perfis 1 e 2, basta digitarmos:

```
lista_bin[23:25]
```

Caso queiramos acessar os testes para variáveis específicas, não sequenciais, comparando os perfis 1 e 2, basta digitarmos quais variáveis queremos com a função concatenar ("c()"). Nesse caso, se queremos as variáveis 21, 23 e 25, digitamos:

```
list[c(21,23,25)]
```

Veja que para acessar um objeto específico dentro da lista, utilizamos dois colchetes: "[[]]". Para assessarmos um grupo de objetos, sequenciais ou não, utilizamos apenas um colchete: "[]". A saída do nosso exemplo para o teste entre perfils 1 e 3 para a variável 21 seria:

```
    2-sample test for equality of proportions without
continuity correction

data:  table(data[, i], data$perfil)[, c(1, 3)]
X-squared = 1.3323, df = 1, p-value = 0.2484
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.05984053  0.23268003
sample estimates:
   prop 1    prop 2
0.5555556 0.4691358
```

Nesse caso vemos que o Perfil 1 possui proporção de sucessos igual a 0.56, contra 0.47 para o Perfil 3. Esses resultados encontram-se no final do *output*, denominados de "*sample estimates*". Vimos que nesse caso o *p-valor* = 0.2484 sugere que não há evidências estatísticas suficientes para rejeitar a hipótese nula de que as proporções da variável "var21" entre os dois perfis sejam idênticas. Assim, baseado nesse teste, não conseguimos encontrar distinção entre os dois perfis comparados para essa variável em específico.

No caso do nosso exemplo com os bancos de dados salvos em Excel: "teste_gam.xls" e "teste_cel.xls", executaríamos os mesmos procedimentos do início dessa seção, ou seja, fazendo o *merge* (junção) desses bancos de dados. Nossas variáveis, "var1" até "var28" seriam todas categóricas. O teste de proporções pode, então, ser aplicado normalmente para as variáveis binárias. Assumindo que o banco continue chamando "teste_celgam", e que todas as 28 variáveis sejam binárias, bastaríamos executar o seguinte script para comparar os perfis 1 e 2, assumindo que os perfis assumam valores de 0 a 9 como na Tabela 2:

```
lista_celgam=list()
for (i in 1:28) {
lista_celgam[[i]]=prop.test(table(teste_celgam[,i],teste_celga
 m$perfil)[,c(2,3)],
          alternative='two.sided',
          conf.level=0.95,
          correct=FALSE)
}
```