**Teste de hipótese usando distância L1 e Bootstrap**

Rafael Bertola, 993163

Rodrigo Hoinkis Mazza, 982073

José Marcos Medeiros Vendramini 991942

Cassiano Ricardo Garcia de Souza, 991523

17 de Junho de 2002

**RESUMO**

O objetivo do trabalho é apresentar um procedimento prático, eficiente e de fácil aplicação para testar as hipóteses http://bertola.br.tripod.com/estat/me523/l1/Image1.gifcontra http://bertola.br.tripod.com/estat/me523/l1/Image2.gifsendo *F* a função de distribuição de *n* variáveis aleatórias independentes *X* e *G* a função de distribuição de *n* variáveis aleatórias independentes *Y*.

Usou-se para isto, um teste baseado na distância entre as densidades estimadas de *F* e *G* que foram obtidas pela forma mais elementar da estimação de densidades de Kernel, chamou-se este teste de distância *L1*.

Uma simulação de experimentos foi usada para comparar a distancia *L1* com outras estatísticas, como a t e a estatística de Kolmogorov-Smirnov. Essa simulação é feita por Bootstrap.

**INTRODUÇÃO**

Assuma *X1, X2, ..., Xn* variáveis aleatórias independentes com distribuição *F* e assuma a mesma condição para *Y1, Y2, ..., Yn* e *G*. As funções de distribuição *F* e *G* são contínuas e independentes.

Queremos testar a hipótese nula http://bertola.br.tripod.com/estat/me523/l1/Image1.gifcontra http://bertola.br.tripod.com/estat/me523/l1/Image2.gif. Existe uma variedade bastante grande de testes comparativos entre distribuições, mas aqui vamos trabalhar com a distância entre duas distribuições de densidade estimadas. Um teste baseado somente nos parâmetros da distribuição, como o teste t, não parece ser tão eficiente quando comparado com a distância de duas funções de densidade. No primeiro estamos comparando médias da distribuição, já no segundo toda a distribuição é comparada (ponto por ponto) pelos seus valores estimados. As densidades, *f* e *g*, das amostras são desconhecidas. Para estima-las usaremos a estimação de densidades por kernel do modo mais elementar possível.

Para as estatísticas t e L1, foi usado a técnica de reamostragem bootstrap da seguinte forma:

Para testar a estatística t foi construído um intervalo de confiança para t de 95% para cada rodada de bootstrap. Com isso, para cada rodada onde este intervalo não continha o valor zero era adicionado um ponto a um contador. Sendo assim B ensaios de Bernoulli eram realizados para cada repetição do experimento, e, considerando a fração do número de rejeições (contador) dividido por B, e, sendo superior a 0.05 esta repetição rejeitava a hipótese de que as distribuições eram iguais, e caso fosse menor que 0.05 era aceito que não tínhamos evidência para rejeitar a igualdade das distribuições.

Com R repetições deste experimento, pode-se chegar a uma taxa de rejeição deste método com a simples fração entre número de rejeições totais das repetições sobre R.

A estatística t falha quando a média das distribuições é igual, e com dispersão diferente. Ele é usado nesse caso somente como efeito comparativo.

Já o método de bootstrap para o L1 foi elaborado de forma um pouco diferente. Treinamos o computador para estabelecer um ponto de corte a partir de simulações baseada na distribuição normal.

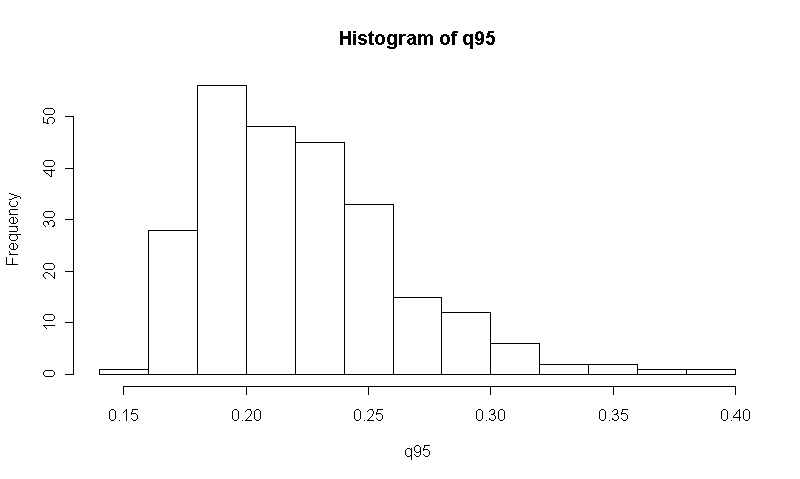
Para comparação usamos o teste Kolmogorov-Smirnov, que é bastante conhecido para este tipo de aplicação.

Esse estudo é baseado no artigo de David L. Allen na The American Statistician (veja referência), o código para a distância entre as estimativas da distribuição foi retirado do site do autor.

**TESTES BOOTSTRAP E ESTIMAÇÃO POR KERNEL**

O método de Bootstrap consiste em fazer uma reamostragem com reposição das amostras originais resultando em novas amostras que irão produzir uma estatística  \*. Esse processo é repetido B vezes produzindo um grande vetor de valores da estatística  \* . Inferências sobre a estatística  ^, que vem da amostra, são feitas sobre  \*. (Ver Efron e Tibshirani)

Para o teste de distância-L1 usamos um teste empírico baseado nos resultados de  \* (as distâncias das reamostragens) contra  ^. Para estabelecer o ponto de corte foram feitas 1000 simulações de F contra G (com B=499) ambas iid ~ N(0,1). Usamos o percentil 95 para estabelecer o ponto de corte da amostra. Abaixo o gráfico das simulações que resultou no ponto de corte escolhido:



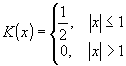
O ponto de corte empírico utilizado foi 0.2960.

O teste Kolmogorov-Smirnov, incluído para comparação, tem uma estatística conhecida e portanto não precisa de um método de reamostragem como o bootstrap para o cálculo de sua estatística. Na simulação usamos uma função do programa R (ks.test), passamos para ela as densidades estimadas e guardamos o p-valor por ela retornado. Comparamos esse valor com o nosso valor de significancia *alpha = 0.05*.

Como *f* e *g* são desconhecidas usamos a amostra para estimar as distribuições. Entrando no mundo da estatística não-paramétrica usamos a estimação de densidade por kernel, nossa função foi definida como:

http://bertola.br.tripod.com/estat/me523/l1/Image4.gife http://bertola.br.tripod.com/estat/me523/l1/Image5.gif

onde



http://bertola.br.tripod.com/estat/me523/l1/Image7.gifé a estimativa de http://bertola.br.tripod.com/estat/me523/l1/Image8.gifbaseado na amostra *Xi*. A função *K* do kernel, é uma função degrau, que facilita as contas do algoritmo de distância e é facilmente guardada em vetores. O parâmetro de suavização utilizado foi o sugerido por Allen em seu artigo, que de acordo com ele este valor para *C* é de fácil implementação.

A distância-L1 é uma estatística que mede a distância entre duas densidades. Ela é definida como:

http://bertola.br.tripod.com/estat/me523/l1/Image9.gif

O mesmo é válido quando temos as estimativas http://bertola.br.tripod.com/estat/me523/l1/Image10.gife http://bertola.br.tripod.com/estat/me523/l1/Image11.gif*.*

**SIMULAÇÕES E RESULTADOS**

Duas amostras de tamanho 80 são geradas, uma da distribuição *F* e outra da *G*.

Uma quantidade de 499 reamostragens por Bootstrap é feita a partir dessas duas amostras gerando valores para o cálculo de  \*. Isso conclui um teste, serão feitos 500 testes para novas amostras das mesmas distribuições. Para cada uma das 499 reamostragens são calculadas as estimativas das densidades, para que os cálculos sejam feitos como se as distribuições *F* e *G* fossem desconhecidas.

|  |  |  |  |
| --- | --- | --- | --- |
| **Taxa de rejeição do teste de igualdade H0: F = G, F ~ N (0,1)** | | | |
|  | | **Bootstrap** | |
| **Distribuições (G)** | **K-S (%)** | **t (%)** | **L1 (%)** |
| N (0,1) | 3.0 | 6.0 | 5.0 |
| N(0.25, 1) | 19.8 | 33.6 | 31.5 |
| N (0.5, 1) | 69.8 | 88.2 | 81.0 |
| N (0, 2^2) | 100.0 | 5.8 | 100.0 |
| N (0, 1,5^2) | 80.4 | 5.6 | 100.0 |
| Cauchy(0,1) | 16.6 | 7.8 | 100.0 |
| Uniforme(-1.68,1.68) | 7.8 | 6.4 | 3.0 |
| NC (0,1,.5,0.01) | 7.2 | 8.8 | 85.2 |

Primeiramente foi testado a sensibilidade da alteração da média na distribuição normal. Todos os testes se apresentaram eficientes e sensíveis a esse tipo de mudança.

Depois foi testado a sensibilidade da alteração da variância na distribuição normal. Apenas a estatística t não foi sensível a essa mudança, o que já era esperado pois a estatística não capta a dispersão dos dados.

Testando se existe sensibilidade dos testes quando comparado com distribuições diferentes foi citada a comparação com a Cauchy padrão. Somente a estatística L1 conseguiu diferenciar as distribuições. Já quando comparado com uma distribuição uniforme [-1.68, 1.68], nenhuma estatística conseguiu diferenciar as distribuições.

Por fim comparamos uma distribuição normal contaminada, e novamente, onde só a estatística L1 conseguiu um resultado satisfatório.

**CONCLUSÃO**

Embora a estatística L1 se apresentou muito eficaz, devemos lembrar que ela foi "treinada" a partir de uma normal padrão, portanto se fossemos usa-la com outras distribuições seu desempenho provavelmente não seria tão satisfatório. Mas isso não impede de se elaborar um método que simule um ponto de corte apropriado para qualquer distribuição.

O teste Kolmogorov-Smirnov também não foi tão satisfatório quando foram comparadas distribuições diferentes, onde a L1 mostrou-se eficiente com exceção somente no caso da uniforme.

Também deve-se levar em conta o esforço computacional que o cálculo que a estatística L1 necessita.

**REFERÊNCIAS**

Allen, David L., "Hypothesis Testing Using an L1-Distance Bootstrap", The American Statistician, vol. 51, No 2, 145-150.

\_\_\_\_\_\_\_\_\_\_\_\_, http://www.eou.edu/~dallen/lp\_dist\_code.html

Efron, B. and Tibshirani, R., "An Introduction to the Bootstrap", New York , Chapman and Hall

**CÓDIGOS**

**Distância L1**

rm(list=ls())

### inicio da funcao que calcula dist L1 ###

f <- function(x,y,k){

cx<-(k^(1/5))/(2\*sd(x))

cy<-(k^(1/5))/(2\*sd(y))

kF <- density(x,cx ,kernel="r", n=k) # kernel density F^

kG <- density(y,cy, kernel="r", n=k) # kernel density G^

F <- kF$y # os valores da funçao estimada

G <- kG$y

S <- kF$x # pontos onde a densidade F é estimada

T <- kG$x

NF <- kF$n # a quant de pontos amostrados; tamanho do array

NG <- kG$n

S[1] <- min(S[1], T[1])

T[1] <- S[1]

S[NF] <- max(S[NF-1], T[NG-1])

T[NG] <- S[NF]

End <- S[NF]

Fcur <- F[1] # valor corrente do F^ no loop

Gcur <- G[1]

InextF <- 2 # índice do prox valor de F^

InextG <- 2

Low <- S[1]

L1 <- 0

while (Low < End) {

if(T[InextG] < S[InextF]) {

High <- T[InextG]

L1 <- L1 + (abs(Fcur - Gcur))\*(High-Low)

Gcur <- G[InextG]

InextG <- InextG + 1

}

else{

High <- S[InextF]

L1 <- L1 + (abs(Fcur - Gcur))\*(High-Low)

Fcur <- F[InextF]

InextF <- InextF + 1

} # fim do if

Low <- High

} # fim do while

return(L1)

}

# inicio da simulação

repli <- 250

rej <- 0

for (r in 1:repli){

n <- 80

x <- rnorm(80,0,1)

y <- rnorm(80,0.5,1)

y2 <- rnorm(80, 0.5, 0.01)

z <- runif(1,0,1)

if (z <= 0.05) y <- y2

cont <- 0

nboot <- 500

dist <- rep(0,c(nboot))

# bootstrap

bsL1x <- matrix(sample(x, size = n \* nboot, replace = TRUE), nrow = nboot)

bsL1y <- matrix(sample(y, size = n \* nboot, replace = TRUE), nrow = nboot)

# calculo das estatisticas do bootstrap

for (i in 1:nboot){

bsL1x[i,] <- sort(bsL1x[i,])

bsL1y[i,] <- sort(bsL1y[i,])

dist[i] <- f(bsL1x[i,],bsL1y[i,],n)}

# teste de hipotese

for (j in 1:nboot){if (dist[j] >= 0.2960) cont<-cont+1}

if (cont/nboot >= 0.05) rej <- rej + 1

print(r)

}

rej <- rej/repli

print(rej)

**Teste KS**

rm(list=ls())

repli <- 2

ac <- 0

pks <- 0

act <- 0

nboot <- 499

sqx <-array(0,c(nboot))

sqy <-array(0,c(nboot))

den <-array(0,c(nboot))

n <- 22

####################### Replicação (for principal) ################

for (i in 1:repli){

x <- rnorm(22)

y <- rnorm(22, 0, 1)

### Bootstrap

bootsamx <- matrix(sample(x, size = n \* nboot, replace = TRUE), nrow = nboot)

bootsamy <- matrix(sample(y, size = n \* nboot, replace = TRUE), nrow = nboot)

### Estatística t

xbar <- apply(bootsamx, 1, mean)

for(j in 1:nboot)

ybar <- apply(bootsamy, 1, mean)

for(j in 1:nboot)

{sqy[j] <- sum((bootsamy[j,]-ybar[j])^2)}

{sqx[j] <- sum((bootsamx[j,]-xbar[j])^2)}

t <-array(0,c(nboot))

den <- sqrt((sqx + sqy)/(2\*n - 2))\*sqrt(2/n)

t <- (xbar-ybar)/den

t1 <- t.test(x,y)

tchap <- t1$s

qi <- quantile(t, 0.025)

qs <- quantile(t, 0.975)

print(i)

#print(qi)

#print(qs)

if (qi < 0 & qs > 0) ac <- ac + 1

sdteta <- sd(t) # fazer conta com teta

vet <- (t-tchap)/sdteta

qi <- quantile(t, 0.025)

qs <- quantile(t, 0.975)

if (qi < 0 & qs > 0) act <- act + 1

### Kolmogorov - Smirnov

ks <- rep(0,repli)

res <- ks.test(x,y) # guarda a lista de resultados

ks[i]<- res$p.value # guarda os p-valores de cada teste

if (ks[i]<=0.05) pks<- pks+1

}

####################### Fim da Replicação (for principal) ################

prop <- ac/repli #

rejt <- 1-prop # rejeição da t

rejteta <- 1 - act/repli

rejks <- pks/repli