MODELIMI I TESTIMEVE LOGJIKE E PROBABILITARE PËR TË DHËNAT NË SERITË E VAZHDUESHME

***SAATÇIU D., BIMBARI B.**

Universiteti i Tiranës, Fakulteti i Shkencave të Natyrës, Departamenti i Informatikës

e-mail: denis.saatciu@fshn.edu.al

Përmbledhje

Hedhja e informacionit në aplikime të ndryshme shoqërohet gjithmonë me gabime. Shumë prej aplikimeve kanë forma kontrolli për shmangien e gabimeve, por gjithsesi është e pamundur eliminimi total i tyre, sidomos në rastet e një proçesi masiv dixhitalizimi në një kohë të përshpejtuar. Në këtë artikull trajtohet modelimi informatik i proçesit të testimeve që lidhen me cilësinë e hedhjes së të dhënave për seri të vazhdueshme kohore. Do të paraqitet një strukturë e bazës së të dhënave e përshtatshme për ruajtjen e serisë së të dhënave për variabla të ndryshëm. Mbi këto të dhëna do të aplikohen teste të ndryshëm tregues e korrigjues të gabimeve.

Abstract

Data entry in different applications is always finished with errors. Most applications have controls to avoid errors, but nevertheless their total elimination is impossible, especially in a massive digitalization process in a short time. In this article we will present testing process modeling regarding quality of data entry continuous series. An adapt database structure will be designed to store data series of different variables. On those data, will be applied different indicator and corrective tests.

Fjalëkyçe: testim, seri të vazhdueshme, test Man-it, interval besimi, modelim.

Hyrje

Zhvillimi i teknologjisë ka çuar në informatizimin e shumë proceseve. Këto procese shoqërohen me operacione ditore hedhje të dhënash si dhe me një migrim masim të të dhënave të mëparshme të cilat ndodhen në letër. Në të shumtën e rasteve, migrimet kryhen para se aplikimi të vendoset në punë. Ato kërkojnë një proces testimi, në mënyrë që të prodhojnë raporte të besueshme.

Për gjetjen e këtyre gabimeve janë zhvilluar metoda të ndryshme. Në këtë artikull trajtohet modelimi informatik i procesit të testimeve që lidhen me cilësinë e hedhjes së të dhënave për seri të vazhdueshme kohore. Do të paraqitet një strukturë e bazës së të dhënave e përshtatshme për ruajtjen e serisë së të dhënave për variabla të ndryshëm. Mbi këto të dhëna do të aplikohen teste të ndryshëm tregues e korrigjues të gabimeve.

Metoda që ofron saktësinë më të madhe është metoda e quajtur "hedhje dopjo"(Paulsen, *et al*, 2012), ku secila prej të dhënave hidhet nga dy operatorë të ndryshëm në dy kopje të sistemit. Në përfundim të hedhjes së të dhënave bëhet një kontroll vetëm mbi të dhënat që nuk korrespondojnë me

njëra-tjetrën. Niveli besimit për saktësinë mund të rritet duke shtuar operatorët që hedhin të njëjtat të dhëna. Kjo metodë e ul ndjeshëm numrin e gabimeve, por ka një kosto të lartë. Në këtë rast, kosto rritet në përpjesëtim të drejtë me numrin e operatorëve. Ajo mund të përdoret kur numri i të dhënave është i vogël, ose në rastin kur toleranca ndaj gabimeve është e papranueshme.

Metodë tjetër për kontrollin e të dhënave, është përdorimi i kontrolluesve të të dhënave të hedhura nga operatorët. Në këtë rast, kosto është më e ulët se në rastin e parë, pasi numri i kontrolluesve mund të jetë më i ulët se numri i operatorëve, p.sh. mund të kemi një kontrollues për çdo katër operatorë. Por numri i gabime mund të jetë më i madh.

Metodat e mësipërme mundësojnë eliminimin e gabimeve gjatë hedhjes së të dhënave, por nuk na japin informacion për cilësinë e matjes së këtyre të dhënave. Për të kontrolluar cilësinë e të dhënave, si dhe në rastin kur nuk mund të rikontrollojmë të dhënat me operatorë shtesë, sygjerohet përdorimi i metodave statistikore, të cilat tregojnë ato të dhëna që potencialisht mund të jenë të gabuara(Ramírez, & Ramírez, 2008).

Materiali dhe metodat

Si sistem për menaxhimin e bazës së të dhënave është përdorur SQL Server 2008 R2.

Gjuha për krijimin e objekteve është T-SQL.

Krijohet tabela “seriteBruto” e të dhënave bruto

```
CREATE TABLE seriteBruto(
    [id] [int] IDENTITY(1,1) NOT NULL,
    [Variabel] [nvarchar](50) NULL,
    [Vendndodhje] [nvarchar](50) NULL,
    [Data] [datetime] NULL,
    [Vlera] [nvarchar](50) NULL
)
```

Në këtë tabelë mund të regjistrohen të dhëna bazë. Sipas kërkesës, tabela mund të zgjerohet duke shtuar fusha të reja, p.sh. matje të ndryshme për të njëjtën datë, etj.

Vihet re që të në këtë tabelë regjistrojmë dy atribute kryesore për të dhënat, variablin dhe vendndodhjen. Zakonisht, seritë e të dhënave janë për çdo dyshe të këtyre atributeve.

Të dhënat duhet të merren nga aplikimet dhe pas transformimeve përkatëse të ngarkohen në tabelën e mësipërme. Procesi i transformimit mund të automatizohet nëpërmjet ndërtimit të proçedurave T-SQL.

Kalimi i të dhënave në një vend të vetëm na krijon mundësinë të ndërtojmë vetëm një proçedurë për secilin test të pavarur nga variabli, i cili do të testohet. Në këtë rast ky variabël mund të kalojë si parametër.

Para se të kryhen testimet, duhet të bëhet unifikimi i vlerave. Të dhënat që nuk janë plotësuar markohen me një vlerë të caktuar, p.sh. -55555.

Update `seriteBruto set vlera=-55555 where vlera is null`

Kategoria e parë (testi i limiteve)

Testi i limiteve kontrollon që vlerat e një variabli të jenë brenda limiteve të caktuara, p.sh. çmimi i bukës vendoset nga 0 në 120 lekë, pasi dihet që gjatë kohës nuk ka pasur çmim jashtë këtij limit. Çdo vlerë jashtë këtij limiti do quhet vlerë e gabuar dhe do kalohet për rishikim.

Automatizimi i testit kryhet si me poshtë:

Krijojmë tabelën ku vendosim testet

```
CREATE TABLE [testet](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [testues] [nvarchar](50) NULL,
    [kategori] [nvarchar](200) NULL,
    [tabela] [nvarchar](200) NULL,
    [kushti] [nvarchar](200) NULL,
    [pershkrim] [nvarchar](200) NULL,
    [fusha1] [nvarchar](200) NULL,
    [fusha2] [nvarchar](200) NULL,
    [limiti_poshtem] [float] NULL,
    [limiti_siperm] [float] NULL
)
```

Komanda për listën e testeve është:

select id, kategori, tabela, kushti, pershkrim, fusha1, fusha2, limiti_poshtem, limiti_siperm from testet where kategori=@kategori order by id

Duke krijuar një kursor mbi setin e të dhënave të marra më sipër, mund të nxjerrim listën e vlerave që nuk plotësojnë kushtin për secilin test me anë të komandës:

*Exec("select * from "+@tabela+" where "+ @kushti)*

Kategoria e dytë (testet spacialë)

Këto teste kanë qëllim kontrollues – korrigjues, bazuar në konceptin e fqinjësisë gjeografike të çdo vendndodhjeje me vendndodhjet përreth.

(Getis 2009) duke iu referuar (Bailey & Gatrell.1995) propozon zbatimin e testeve numër 1 dhe 2 si më poshtë:

Shënim: t(i) i referohet vlerës së variablit në ditën e i-të.

Testi nr.1

Nëse |t(i)-t(i+1)|> limit_1

Nëse për çdo k €{1,2,3}: |[t(i,k)-t(i+1,k)]-[t(i)-t(i+1)]|>limit_2

Testi dështon, vlera t(i) duhet të kontrollohet

Shënim: k €{1,2,3} i referohet tre vendndodhjeve të ndryshme fqinje.

t(i,k)-jep vlerën e serisë në ditën e i-të në vendndodhjen e k-të

Vlerat limit_1 dhe limit_2 duhen të vendosen sipas rastit të serisë së të dhënave në shqyrtim. P.sh. në rastin e një serie të temperaturave ditore, mund të themi që lëvizja prej 10 °c, midis dy vlerave të njëpasnjëshme, është e dyshimtë, duke vendosur kështu limit_1=10. Gjithashtu mund të themi që ndryshimet e diferencave të vlerave të njëpasnjëshme midis dy vendndodhjeve fqinje nuk duhet të jenë më shumë se 3 °c, duke vendosur kështu limit_2=3.

Nëse ndodh që diferenca e vlerave të njëpasnjëshme të jetë më e madhe se limit_1, atëherë kjo mund të ketë ndodhur ose si rrjedhojë e ndonjë anomalie të sistemit, ose si rrjedhojë e hedhjes gabim së të dhënës. Shikohet diferenca midis vlerave të njëpasnjëshme në vendndodhjet fqinjë, përgjithësisht tre janë të mjaftueshëm. Nëse në secilën vendndodhje, diferenca, midis vlerave të njëpasnjëshme, ndryshon më shumë se një vlerë e caktuar (limit_2) nga diferenca t(i)-t(i+1), atëherë vlera t(i) markohet si e dyshimtë, përndryshe mendohet si e saktë.

Testi nr.2

Nëse për çdo k € {2,3,4}: |t(i,l)-t(i,k)|>limit

Ath t(i, l) quhet matje e matur keq.

Apliko interpolimin gjeografik si më poshtë:

t(i,l)=[t(i,2)+t(i,3)+t(i,4)]/3

Shënim: k €{1,2,3} i referohet tre vendndodhjeve të ndryshme fqinje.

t(i,k)-jep vlerën e serisë në ditën e i-të në vendndodhjen e k-të

Nëse ndodh që për një datë të caktuar, diferenca e vlerës së variablit me secilën prej vlerave të variablit në vendndodhjet fqinje të jetë më e madhe se *limit*, atëherë kjo mund të ketë ndodhur ose si rrjedhojë e ndonjë anomalie të

sistemit, ose si rrjedhojë e hedhjes gabim së të dhënës. Në këtë rast vlera t(i) markohet si e dyshimtë. Në qoftë se kërkohet korrigjimi i kësaj vlere ajo interpelohet duke u zëvendësuar me mesataren e vlerave të njëjtit variabël në të njëjtën datë në vendndodhjet fqinje.

Vlera limit duhet të vendoset sipas rastit të serisë së të dhënave në shqyrtim.

Testi dispersionit (http://www.sascommunity.org/mwiki**)**

Llogariten vlerësimet e mesatares dhe dispersionit të serisë për 30 ditë para ditës i, sipas formulave:

$$\overline{t(i-1)} = \frac{1}{30}\sum_{k=i-1}^{i-30} t(k);$$

$$\sigma(\overline{i-1})^2 = \frac{1}{30}\sum_{k=i-1}^{i-30}\left[t(k) - \overline{t(i-1)}\right]^2$$

Vlera është në rregull nëse vlera e matur pasardhëse t(i) është

$$\overline{t(i-1)} \pm 2.05 * \sigma(i-1)\sqrt{\frac{30+1}{30}}$$

Metoda më sipër nënkupton që shpërndarja e serisë t(i) është normale.

Niveli i besimit 95 % mund te ndryshohet.

Më poshtë paraqesim pjesën transact SQL të procedurës që gjen vlerat e dyshuara për një variabël të caktuar.

Proçedura përdor dy tabela të përkohshme, vlera Temp, e cila përdoret për llogaritje brenda procedurës, si dhe vleraTest 2, në të cilën regjistrohen vlerat e dyshuara të serisë. Procedura do të marrë si parametër emrin e variablit.

create table vleraTemp(id int, vlera float)

create table vleraTest(
variabli nvarchar(50),
stacioni nvarchar(50),
data date,
t float,
s float,
vlera float
)

Në këtë proçedurë, variabli merret si parametër *@variabli*. Atributi i dytë i nevojshëm për tu përcaktuar është vendndodhja. Ndërtojmë një kursor që bredh listën e vendndodhjeve:

declare kursoriSt cursor FOR select vendndodhje from vendndodhjet

Kursori do e kalojë listën e vendndodhjeve te @vendndodhje.

Për secilën vendndodhje, pastrojmë tabelën e përkoshshme vleraTemp dhe inicializojmë @i=1:

set @i=1

delete vleraTemp

Ndërtojmë një kursor që bredh datat një e nga një:

declare kursoridate cursor FOR select data,vlera from seritebruto where vendndodhje=@vendndodhje and variabli=@variabli order by data

Kursori do e kalojë listën te *@data* dhe *@vl.*

Në ciklin e parë, kemi *@i=1,* prandaj do kemi incializim te *@data1* dhe jo *@data2*.

Kjo sjell që diferenca e *@data1* dhe *@data2* nuk do të jetë një, pasi @data2 ka një vlerë të mëparshme. Vini re që kjo ndodh vetëm në ciklin e parë të kursorit të datës për secilën vendndodhje, prandaj ne duhet të pastrojmë tabelën e përkohshme vlera Temp.

```
if @i=1
    set @data1=@data
else
    set @data2=@data
if dateadd(day,1,@data1)!=@data2
begin
    delete vleraTemp
    set @data1=@data
    set @i=1;
end
```

Për 30 ditët e para nuk mund të kryejmë veprime, pasi nuk kemi se c'mesatare të gjejmë, prandaj për 30 vlerat e para vetëm i vendosim te *vlera Temp*.

Nëse kemi kaluar 30 ditëshin e parë, gjejmë mesataren e tabelës dhe e vendosim te @t. Gjejmë dispersionin dhe e vendosim te @s. Kontrollojmë vlerën nëse është në rregull sipas formulës më sipër. Nëse jo, regjistrojmë në tabelën vleraTest datën, vlerën, mesataren dhe shangien e dispersionit.

Vini re që në tabelën *vlera Temp*, mbajmë vetëm vlerat e 30 ditëve të fundit. Kjo arrihet duke fshirë vlerën më të hershme të regjistruar në tabelë.

Inicializimi *@data1=@data* në fund të ciklit dhe *@data2=@data,*pasi *@i*!=1, në fillim të ciklit pasardhës mundëson *@data2=@data1+1* duke mos pastruar tabelën vleraTemp dhe vendosur *@i*=1.

```
if @i<31
begin
  insert into vleraTemp values (@i,@vl);
  set @i=@i+1;
end
else
begin
  select @t=round(sum(vlera)/30,2) from vleraTemp
  select @s=round(sum(power(vlera-@t,2))/30,2) from vleraTemp
  if abs(@t-@vl)>2.05*round(sqrt(@s),2)*round(sqrt(31/30),2)
    insert into vleraTest values (@seri, @stacioni, @data, @t,
2.05*round(sqrt(@s),2)*round(sqrt(31/30),2), @vl)
  insert into vleraTemp values (@i,@vl)
  delete vleraTemp where id=@i-30
  set @i=@i+1;
end
set @data1=@data
```

Testi Mann-it)

Shënim:

Testi i Manit (Nagarajan & Keich 2009) kryhet mbi seritë vjetore të të dhënave, pasi kërkon një numër të caktuar të dhënash për të pasur rezultate me kuptim.

Fillimi i Testit:

Për serinë që do të testohet, vjetore, (p.sh. N=10 vjet) jepet seria **t(i)**

i varion nga 1 deri në 10

1. S(1)=0
2. Fiksojmë i € {2, 10}.
3. Llogaritet statistika: S(i) = S(i-1) + {"Numrin e rasteve që termi
   t(i) > t(k) per k=**1** deri **i-1**"} + 0.5 * {"Numrin e rasteve që termi
   t(i) = t(k) per k=**1** deri **i-1"}**
4. Për çdo **i** llogariten pritja matematike dhe varianca
   E(i)=i*(i-1)/4 Var(i)=i*(i-1)*(2*i+5)/72
5. Llogaritet Statistika e Testit Mann sipas formulës:
   U(i) = [S(i)-E(i)]/SQRT[Var(i)]
6. Nqs U(i) > 1.96, atëherë seria ka heterogjenitet në vitin **i** me probabilitet 95%
7. Fund

Më poshtë paraqesim pjesën transact SQL të procedurës që gjen vlerat e dyshuara për një variabël të caktuar. Procedura do të marrë si parametër emrin e variablit.

Në këtë proçedurë, variabli merret si parametër @variabli. Atributi i dytë i nevojshëm për tu përcaktuar është vendndodhja. Ndërtojmë një kursor që bredh listën e vendndodhjeve:

declare kursoriSt cursor FOR select vendndodhje from vendndodhjet

Kursori do e kalojë listën e vendndodhjeve te @vendndodhje.

Bëjmë inicializimet e mëposhtme. Variabli @i do tregojë vitin. @s është shumë progresive gjatë viteve, prandaj inicializohet jashtë ciklit të viteve.

set @i=1
set @s=0

Testi i Mannit shërben për të treguar vitet ku seria ka heterogjenitet me probabilitet 95%. Ndërtojmë një kursor që bredh vitet me radhë:

declare kursori cursor FOR select viti,avg(vlera) from seritebruto where variabli=@variabli and vendndodhje=@vendndodhje group by viti order by viti

Kursori do e kalojë listën te @viti dhe @Sasia

set @s_re=0

select @s_re=sum(a.mes) from (select viti,case when avg(vlera)<@Sasia then 1 when avg(vlera)=@Sasia then 0.5 else 0 end mes from seritebruto where variabli=@variabli and *vendndodhje=@vendndodhje* and viti<@viti group by viti) a

```
set @s=@s+isnull(@s_re,0)
set @e=@i*(@i-1)/cast(4 as float)
set @v=@i*(@i-1)*(2*@i+5)/cast(72 as float)
if(@i>1)
   set @u=(@s-@e)/sqrt(@v)
if (abs(@u)>=2)
  insert into vleraTestMan values (@stacioni,@viti,@u)
set @i=@i+1
```

Cikli përsëritet vetëm për të gjithë vitet.

Përfundime

Në ditët e sotme, kompanitë hedhin të dhëna në aplikime gjithmonë e më tepër. Për ketë arsye, testimi i të dhënave të futura është tepër i rëndësishëm.

Transformimi paraprak i informacionit, duke u bazuar në një model të përgjithshëm, lehtëson automatizim dhe përgjithësimin e testeve.

Metodat e testimit mund të shtohen, për të përfshirë në të edhe testime shtesë që do të ndihmonin në korrigjimin edhe më të mirë të gabimeve.

Testimet duhet të kryhen periodikisht nga kompanitë, dhe jo të lihen si proces i fundit. Numri i gabimeve në fund mund të jetë shumë i madh dhe të detyrojë, në situata të caktuara, të bëhet kompromis midis kohës në dispozicion dhe cilësisë së të dhënave.

Literatura

Aksel Paulsen, Søren Overgaard,Jens Martin Lauritsen(2012): Quality of Data Entry Using Single Entry, Double Entry and Automated Forms Processing–An Example Based on a Study of Patient-Reported Outcomes. P L o S One, Vol. 7, No. 4, 2012

Ramírez, J. G., and Ramírez, B. S. (2008): Analyzing and Interpreting Continuous Data Using JMP: A Step-by-Step Guide. Cary, NC: SAS Institute Inc

Getis A. (2009): Spatial statistics

Bailey T.C & Gatrell A.C. (1995): Interactive Spatial Data Analisys. Longman, Harlow.

José G. Ramírez, W.L. Gore and Associates Inc: Statistical Intervals: Confidence, Prediction,Enclosure. http://www.sascommunity.org/mwiki

Nagarajan N., Keich U. (2009): Reliability and efficiency of algorithms for computing the significance of the Mann–Whitney test. Comput Stat (2009) 24:605–622