Fontos epizódok keresése idősorokban

Felkészülés

A mérés során az idősorokban található fontos epizódok keresését fogjuk vizsgálni egy konkrét módszer mentén. A mérésre fel kell készülni. A szükséges ismeretek megtalálhatók E. O. Heierman és tsai.: „Mining Temporal Sequences To Discover Interesting Patterns” című cikkében, amely a mérési útmutató mellett a tárgy honlapjáról letölthető.

A mérésre hozni kell az elkészített házi feladatot (lásd később)!

Néhány további irodalom annak, akit érdekel a téma, de az interneten rengeteg publikáció található „temporal data mining” témában:

- C. M. Antunes and A. L. Oliveira Temporal Data Mining: an overview, Lecture Notes in Computer Science
- V. R. Jakkula, D. J.Cook, and Aaron S. Crandall, "Temporal pattern discovery for anomaly detection in smart homes", Proc. of the the 3rd IET Int. Conf. on Intelligent Environments (IE 07), Germany, September 2007
- Z. Stoecker-Sylvia: Mining for Frequent Events in Time Series, MSc Thesis,. Worcester Polytechnic Institute, 2004
- V. R. Jakkula, and D. J.Cook, "Mining Sensor Data in Smart Environment for Temporal Activity Prediction", ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining Workshop on Knowledge Discovery from Sensor Data (sensor-KDD 2007), San Jose, August 2007

Ellenőrző kérdések:

- Minek alapján javasolták megítélni az epizódok fontosságát?
- Ha az epizódkeresés két maximális epizódot adott, akkor mit javasolták a jelöltek generálására?
- A szimbólumsorban két epizód váltakozik, egyiknek hossza 3, a másiké 8. Tudjuk, hogy a kezdetük mindig legalább 12 időegységre van. Lesz-e különbség, ha az epizódkeresést 5 ablakkapacitással és 10 ablakhosszal végezzük, illetve 10 ablakkapacitással és 5 ablakhosszal?

A mérés

A mérés során az idézett cikkben ismertetett algoritmusokat fogjuk használni. Ezek Matlab m-fájlok formájában állnak rendelkezésre, a BIRFE.zip fájlban tömörítve. Az m-fájlok nem mindenütt egy az egyben valósítják meg a közölt algoritmusokat, néhol egyszerűsítéseket tartalmaznak.

A mérés során nem csupán használni fogjuk a megírt m-fájlokat, hanem bizonyos algoritmusok önálló megírása is követelmény! (Lásd a mérési feladatokat.) Aki akarja természetesen előre megírhatja otthon ezeket, akkor annyival előbb végez. Ugyanakkor az idő elég lesz arra is, hogy helyben megírásra kerüljenek.

Az idősorok **szimbólumsorozatok** formájában állnak rendelkezése. Minden szimbólum 3-karakteres. Ennek megfelelően egy idősor a Matlabban egy N*3-as char mátrixként jelenik meg. Kitüntetett szerepet játszik az „üres” (nem történt semmi) szimbólum: ezt ’XXX’ jelzi az idősorban. Tipikus szimbólumsorozat részlet:

• • •
LOW
XXX
XXX
HIG
HIG
XXX
XXX
• • •

_ _

A szimbólumsor megjelenítésére szolgál a

function NumSor=ShowSzimbSor1(SzimbSor,SzimbKeszlet)

függvény. Ez a SzimbKeszletben található (természetesen szintén 3-karakteres) szimbólumoknak a készletbeli sorszámára cseréli a szimbólumsor elemeit. (Amit nem talál a készletben, arra 0-át ad vissza.) Tehát ha:

SzimbKeszlet = [MED
LOW
HIG]

akkor a fenti szimbólumsor részlet a következő numerikus sorozatba (NumSor) megy át, ami egy egyszerű *plot* paranccsal már könnyen megjeníthető:

• • •
2

0
0
3
3
0
0
• • •

_ _

A tömörítés első lépéseként az üres karaktereket törölhetjük (persze ez csak akkor jelent tömörítést, ha sok ilyen van a szimbólumsorban), ekkor az időinformáció megőrzése érdekében egy SzimbIndex vektorban a „hányadik minta volt az idősorban” információ is visszaadásra kerül:

function [SzimbSor1,SzimbIndex]=SzimbSor2UresNelkul(SzimbSor,UresSzimb)

_ _

Az epizódkeresés cikkben leírt módját valósítja meg a következő függvény:

**function [Epizodok,EpizodHosszak,EpizodokSzlo,EpizodIdok] =
EpizodKereses1(SzimbSor1, SzimbInd1,AblakHossz,AblakKapac);**

Az AblakHossz az eredeti szimbólumsor minta sorszámaival értendő. Tehát a fenti példa szimbólumsorozat esetén nem fér be egyszerre az ablakba a két LOW és a HIG szimbólum, ha az AblakHossz=3, hiába vettük ki az üres szimbólumot közülük.

Az Epizodok kimeneti változóban a talált maximális epizódok ’###’ szimbólummal elválasztva jelennek meg. Tehát a példa szimbólumsorozat részletet feldolgozva (AblakHossz=3, AblakKapac=3) a következő lesz az Epizodok tartalma:

###
LOW
###
HIG
HIG
###

Az EpizodHosszak a talált epizódok szimbólumszámát, az EpizodokSzlo az adott epizód előfordulási számát, az EpizodIdok pedig az egyes előforulások kezdeti időpontját adják meg.

_ _

A következő függvény arra szolgál, hogy az Epizodok közül megmutassa a *K*-dikat.

function [AktEpizod,NE]=ShowEpizodK(Epizodok,K)

Az Epizodkereses1 függvény által visszaadott char-mátrixban tárolt K-dik epizódot mutatja meg. Visszaadja az Epizodok tömbben tárolt K-dik epizódot és az epizódok számát (NE).

_ _

Az Epizodkereses függvény által visszaadott char-mátrixban tárolt *K*-dik epizódot adja vissza.

function AktEpizod=EpizodK(Epizodok,KK)

_ _

A következő függvény az Epizodok K-dik epizódja szerint tömörít. Az Epizodok egy N*3 "character" mátrix, ahol az egyes epizódokat '###' választja el.

function [TomSzimbSor,TomIndex, PeriodikusStrukt, KetperiodusuStrukt, ...
AperiodikusStrukt]= TomoritKdikEpizod1(SzimbSor,SzimbInd,Epizodok, ...
EpizodStartIdok,K);

_ _

A tényleges tömörítést a következő függvény végzi:

function [TomSzimbSor,TomInd,PeriodikusStr,KetperiodusuStr,AperiodikusStr]=
Tomorites1(SzimbSor1,SzimbInd1,AktEpizod,AktEpizodStartIdok);

A kimeneten visszaadja a tömörített szimbólumsort, a megfelelő index (idő) sorral együtt, valamint 3 struktúrában a lehetséges tömörítési megoldásokat. A tömörített szimbólumsorból kihagyja az aktuális epizódot, ez által lesz rövidebb (tömörebb). Ez természetesen csak akkor jelent tényleges nyereséget, ha valamelyik a három leíró struktúrából rövidebb, mint amennyit nyertünk a szimbólumsoron. A három struktúra rendre periodikusan fellépő epizódokra (PeriodikusStr), két váltakozó periódusidővel fellépő epizódokra (KetperiodusuStr) és aperiodikusan fellépő epizódokra (AperiodikusStr) vonatkozik. A függvény csak akkor hozza létre a periodikus (illetve kétperiódusú) struktúrákat, ha dominánsan felfedezhető ez a jelleg. Az aperiodikus struktúra (amely egyszerűen felsorolja a kezdőidőpontjait az aktuális epizódnak) ez

értelemszerűen csak akkor hatékony, ha az epizód elég hosszú. Ezért ez a struktúra csak akkor jön létre, ha az aktuális epizód legalább 9 hosszú (legalább 3db 3 karakteres szimbólumból áll). Az időpontokat ugyanis fixen 8 karakteren ábrázoljuk a struktúrákban.

A függvényben részletesen megtalálható a három – potenciálisan tömörítésre használható – struktúra felépítése.

_ _

A következő függvény két jelölt szimbólumsor legnagyobb közös halmazát adja vissza! (Mind a bemeneten, mind a kimeneten a szimbólumsor lezáró szimbólummal – 'XXX' – keretezve vannak a szimbólumok.)

function Kozos=JeloltHasonlit(Jelolt1,Jelolt2);

_ _

A következő függvény a SzimbSor-ban megkeresi az AktEpizod összes előfordulását, és visszaadja az epizód startidőket. (A bemeneten a SzimbInd-ben adtuk meg szimbólumonként az időpontokat.)

function AktStartIdok=FindStartIdok(AktEpizod,SzimbSor,SzimbInd);

_ _

A mérés során jegyzőkönyvet kell készíteni. A jegyzőkönyv lényegre törő, világos kell legyen. Minden feladnál tartalmaznia kell a feladat megfogalmazását, a megoldás menetének főbb részleteit, az eredményeket és az *eredmények értékelését*.

Otthon megoldandó feladat a laborra való felkészüléshez

A házi feladatot MATLAB segítségével kell megoldani. Töltse be a BIR8.zip állományban található HFSzimbSor.mat állományt! Ebben egy szimbólumsor, az UresSzimbolum és a SzimbolumKeszlet (az állományban SzK néven) található.

HF1. Próbálja végig ezen az állományon a fent ismertetett m-fájlokat! Melyik módszer ad optimális tömörítést?

HF2. Írja meg azt az m-fájlt, amivel ebből a tömörített szimbólumsorból, a tömörített indexsorból és a megfelelő – optimálisan tömörítő, epizód előfordulást megadó – struktúrából vissza tudja állítani az eredeti szimbólumsort! (NE az aperiodikust használja, hanem a legrövidebb leíró struktúrát!) Nem kell mind a háromra megírni, csak arra, amely erre az adott szimbólumsorra a legjobban tömörít.

HF3. Írjon olyan függvényt, amely egy adott szimbólumsorozat (epizód) előfordulásainak kezdeti időpontjait megtalálja a szimbólumsorban!

A mérés során megoldandó feladatok

1. Szimbólumsor legfontosabb epizódjának keresése

Töltse be a BIR8_FE_X.mat állományból a szimbólumsort (SzimbSor, UresSzimb, SzK=SzimbKeszlet)!

1.1 Vizsgálja meg a létrejövő maximálisan hosszú epizódokat az ablak kapacitás és az ablakhossz 5, 10 és 15 értékénél! Hány epizódot talál az egyes esetekben? Mire lehet ebből következtetni a szimbólumsor felépítésére nézve? Lehet-e olyan beállítást használni, amikor csak egyetlen epizódot talál a keresés?

1.2 Amikor csak két epizódot talált, vizsgálja meg a tömörítést az első megtalált epizód, a másik megtalált epizód, illetve a két epizód maximális közös részét használva! Mikor kapjuk a legjobb eredményt?

1.3 Ha csak az 'AAA' szimbólumot felhasználva tömörít, akkor milyen arányt ér el? Milyen jellegű struktúrával lehetne optimálisan tömöríteni ez esetben? (Nem a három felkínált közül, hanem az 'AAA' előfordulásához legjobban illeszkedőt adja meg!

2. Szimbólumsor konstruálás

Készítsen három olyan olyan szimbólumsort (a szokásos 3 karakterből álló szimbólumokból), amelyeknél rendre a periódikus, a kétperiódusú és az aperiodikus struktúra adja a legjobb tömörítést!

3. Szimbólumsor legfontosabb epizódjának keresése

Töltse be a BIR8_FE_Y.mat állományból a szimbólumsort (SzimbSor, UresSzimb, SzK=SzimbKeszlet)!

3.1 Vizsgálja meg a létrejövő maximálisan hosszú epizódokat az ablak kapacitás és az ablakhossz 5, 10 és 15 értékénél! Hány epizódot talál az egyes esetekben? Mire lehet ebből következtetni a szimbólumsor felépítésére nézve? Lehet-e olyan beállítást használni, amikor csak egyetlen epizódot talál a keresés?

3.2 Amikor csak két epizódot talált, vizsgálja meg a tömörítést az első megtalált epizód, a másik megtalált epizód, illetve a két epizód maximális közös részét használva! Mikor kapjuk a legjobb eredményt?

4. Jelöltkiválasztó függvény készítése

Készítsen olyan függvényt, amely a cikkben ismertetett jelöltkiválasztás folyamatát valósítja meg, az alábbi egyszerűsített formában:

a. bemenetként megkapja a talált maximális hosszú epizódokat,
b. kimenetként a jelölteket adja vissza egy, az Epizodok-kal megegyező felépítésű karaktermátrixban,
c. a jelöltek a bemenetként kapott epizódokat (a négy leghosszabbat, lásd következő pont), illetve ezek maximális hosszúságú közös részeit tartalmazzák,
d. hogy a folyamat ne tartson túl sokáig, és ne jöjjön létre nagyon sok jelölt, csak a négy leghosszabb bemeneti epizódot használja fel.

A cikkben a jelöltek közt szerepel a maximális hosszú szimbólumsor és a közös halmaz különbsége is, ezt itt – az egyszerűsítés kedvéért – elhagytuk.