6. óra

Adatelemzési platformok, BME, 2018. Február 28., III. elméleti óra.

Üzleti probléma

Mikroszegmentáció: mintha az adott ügyfélre szóló ajánlatot tudna adni. Ügyfélcsoportok vagy szegmensek képzésén alapszik.

Klaszterezés

- Viszonylag ritkábban használt
- Az sorok felett olyan csoporotokat (klasztereket) alkossunk, amelyek viszonylag közel vannak egymáshoz, míg a csoportok távol vannak egymástól.
- Lehet persze az oszolopokon is klaszterezni.

Főbb típusok

1. Hierarchikus
2. Partícionáló

Mind a kettő 'mohó' algoritmus, elsősorban lokális optimumokat fog elérni.

A képzett klaszterek leginkább gömbszerűek.

1. Hierarchikus

Itt annyi klaszter lesz a végén, ahány sor van, és ezekből mi fogjuk kiválasztani a végső csoportosítást.

Agglomeratív (egyesítő)

- Kiinudlópont: Minden egyes sor vagy elem külön klaszterbe tartozik.
- Iteratíve elkezdi csoportosítani őket amíg minden egyes elem egy klasztrbe tartozik.

Divizív (felosztó)

- Kiinudlópont: Minden egyes sor vagy elem egy klaszterbe tartozik.

- Iteratíve elkezdi szétszedni őket amíg mind külön sorba tartozik.

2. Partícionáló

Meg kell adni neki, hogy hány darab klasztert szeretnénk létrehozni, és paraméterek alapján meg megpróbálja ezeket ráoptimalizálni.

- K-means (k-közép): Átlag alapján alkot központokat (fiktív értékek) és ezek köré szervez egy klasztert.
- K-medoid: adatpontokat használ a klaszterek központjaként és a Manhattan-szabályt alkalmazza a távolság számításhoz (abszolút érték alapján).

Távolságfüggvények

Ezek segítségével határozzák meg, hogy mennyire messzire vannak egymástól.

1. Euklideszi távolság

Négyzetösszegek gyöke: 'átló'.

$$d(i,j) = \sqrt{\sum_{i=1}^{n} (x_i - x_j)^2}$$

2. Manhattan távolság

Abszolut távolság: A vektorok értékének összege.

$$d(i,j) = \sum_{i=1}^{n} |(x_i - x_j)|$$

3. Csebisev távolság

A legnagyobb távolság.

$$d(i,j) = \max_{i} |(x_i - x_j)|$$

Minkovszki távolság

Általánosan ezeket **Minkovszki távolságnak** tekintjük.

$$d(i,j) = (\sum_{i=1}^{n} |x_i - x_j|^p)^{1/p}$$

Adatelőkészítési lépések

1. **Hiányzó értékek**: Ezekre nem tudunk távolságfüggvényt számolni.
2. Mérési szint, a **változók értékkészlete**: Nagyobb értékkészletű változók túldominálhatják a távolságfüggvényeket.
    - Standardizálásal vagy normalizálással lehet ezt megoldani.
3. **Kategória változók**: Nem lehet értelmezni a távolságot a különböző dimenziókban.
    1. **Bináris** változókkal nincs probléma.
    2. **Dummy** változó képzés:
        1. Túl sok új változót kreálhat
        2. A sok értékkel bíró változók túldominálhatják
    3. A **távolságfüggvényt** kell módosítani
        - Kategória változókat csak összehasonlítani tudjuk.
        - Olyan függvényt használunk, ami egyenlőséget vizsgál.
4. **Egyértékű változók**
   Ezeket kihagyjuk.
5. **Véletlenszerű változók**
   Ezeket is kihagyjuk.
6. **Kiugró értékek**
   A Hierarchikusnál nem okoznak nagy problémát, de a Partícionálóknál igen.

K-means

A klaszterekben vett távolságokat akarja minimalizálni
K darab klasztert akarunk létrehozni.

$$SSW = \sum_{i=1}^{k} \sum_{j=1}^{n_j} (x_{ij} - \bar{x}_i)^2$$

- SSW = Sum of squares within
- k = Klaszterek száma
- n_i = Klaszter tagjainak száma
- x_i = Klaszter tagjainak átlaga
- x_{ij} = Klaszterre vonatkozó mérés

Olyan klasztereket próbálunk csinálni, amelyekben a klaszterek elemei közötti távolság minimális.

Iterációs lépések

1. Meghatározuuk a k értékét.
2. Véletlenszerűen kialakítunk k darab pontot, és elnevezzük őket a klaszterek középpontjának.
3. Minden egyes pontot besorolunk egy középponthoz, amelyhez a távolsága a legkisebb. Ez kialakítja a klasztereinket.
4. Új klaszterközéppontot hozunk létre.
5. Iteráljuk a 3. és 4. pontokat.

Leállási feltételek

1. Nem változik a pontok klaszterbesorolása.
2. Az előző kritérium puhítása: leáll, ha már csak egy csekély mennyiségű klaszter változik.
3. SSW távolságfüggvény minimális.
4. Általunk megadott iteráció után.

Tulajdonságok

1. Probléma lehet, ha az elején **rossz kezdőpontokat** választunk ki
    1. Brute force: Lefuttatjuk az összes lehetőségre
    2. Lefuttatjuk több kezdőpont-halmazra
    3. A legtávolabbi k darab pontból indulunk ki
2. **Kiugró értékek**: Ha az elején nincs beválasztva, a 4. iterációs lépésben eltolja a klaszterközéppontot. A 4-es lépésben ne az átlag alapján számoljuk a középpontot, helyette:
    1. K-**medián** érték
    2. Az **átlaghoz legközelebbi valós pont** (bár a kiugró értékeket nem tudja kezelni)
    3. K-medoidhoz hasonló: azt a pontot választjuk ki, **amitől vett klaszteren belüli távolság minimális**.
        1. Ez viszont felerősíti a kezdőpontok hatását.
        2. Számításigénye nagyon nagy lesz

K értéke

1. Hüvejkujjszabály: A sorok számának a felénél a gyökéből induljunk ki $k = \sqrt{\frac{n}{2}}$. Ez nagy adatnál azonban ugyancsak nagyon nagy lenne, ezért nem érdemes használni.
2. Próbálgatással találjuk ki, az SSW-t próbáljuk minimalizálni (ez a javasolt).
3. A partícionáló algoritmusokat meg lehet előzni egy hierarchikussal, ami már ad egy k értéket.

Próbálgatás

Klaszterezés nem felügyelt tanulás: itt nem tudjuk a célváltozót, ezért nem tudunk meghatározott választ adni erre.

SSW: Klaszteren belüli távolság

$$SSW = \sum_{i=1}^{k} \sum_{j=1}^{n_j} (x_{ij} - \bar{x}_i)^2$$

- SSW = Sum of squares within
- k = Klaszterek száma
- n_i = Klaszter tagjainak száma
- x_i = Klaszter tagjainak átlaga
- x_{ij} = Klaszterre vonatkozó mérés

SSB: klaszterek közötti távolság

$$SSB = \sum_{i=1}^{k} n_i (\bar{x}_i - \bar{\bar{x}})^2$$

- SSB = Sum of squares between
- $\bar{\bar{x}}$ = Nagy átlag (az értékek átlaga)

SST: teljes eltérés négyzet

$$SST = SSB + SSW$$

$$\sum_{i=1}^{k} \sum_{i=j}^{n_j} (x_{ij} - \bar{\bar{x}})^2$$

Az SSB/SST könyökpontjánál érdemes a k értékét meghatározni (inflexiós pont).

Elemszám

- A klaszterezés nagyon eltérő elemszámú klasztereket tud alkotni

Davies Boulding index

Visszamérési függvényt

$$R_{i,j} = \frac{S_i + S_j}{M_{i,j}}$$

Where
$R_{i,j}$ = A klaszterezés 'jósága'
S_i = A klaszter belső 'szóródása'
$M_{i,j}$ = A klaszterek közötti távolság

$$DB = \frac{\sum_{i=1}^{k} \max_{i \neq j} R_{i,j}}{k}$$

Minél kisebb, annál jobb.

Hierarchikus klaszterezés

A felosztó nagyon hasonló mint az egyesítő, ezért csak az utóbbit vesszük.

Távolságok definiálása

Hogyan nézzük meg két többelemű klaszter számolását?

1. Single Linkage: legközelebbi pontok távolsága
2. Complete linkage: legtávolabbi pontok távolsága
3. Average linkage: Átlagos távolság
4. Centroidok távolsága

Példa

Minden elem külön klaszterbe tartozik
Egy dimnezió

0. Példa, 10 elemű adatsor: 2, 5, 9, 15, 16, 18, 25, 33, 33, 45
1. Két legközelebbit összevonva (33-ak), már csak kilenc elem.
2. Távolságok számolás, példa lépések (single linkage):
    - (15, 16)
        - (15, 16, 18)
            - (2, 5)
                - (2, 5, 9)
                    - (2, 5, 9, 15, 16, 18)
                        - stb...

- A fenti teljesítményfüggvényeket lehet használni
- Lánc alakú klasztereket képez

- Ha egy lépést megtettünk, az a lépés már nem fog változni. Nem biztos, hogy értelmesek lesznek a klaszterek.

Klaszterezésnek és felügyelet nélküli tanításnak ez egy nagy problémája, ezért is ritka. Hüvejkujjszabályként, az a jó klaszterezés, amely eredményeként a klaszteréknek maximum egy mondatos nevet tudunk adni. Ez azt sugallja, hogy valami erősen jellemző rájuk, tehát valószínüleg van értelmük.