Gépi tanulás gráfokkal

Bodó Zalán
Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar

1. Bevezetés

A gráfelmélet a matematika és számítástudomány egyik fontos ága. Az első, 1736-ban publikált – a königsbergi hidak problémájával foglalkozó – gráfelméleti tanulmány Leonhard Euler nevéhez fűződik. Azóta a gráfelmélet jelentős tudományággá nőtte ki magát, melynek alkalmazásaival szinte minden tudományterületen találkozhatunk. A sok fontos alkalmazás közül itt mindössze egyet emelnénk ki: a számítógépes hálózatokban a routerek gráfelméleti algoritmusokat használnak annak eldöntésére, hogy milyen *optimális* útvonalon továbbítsák a fogadott csomagot.

Gráfokkal a gépi tanulásban is találkozhatunk. A gráf alapú tanuló algoritmusok bemenetei nem maguk a tanulási adatok lesznek, hanem az adatok felett értelmezett *hasonlósági gráf*, amely alapján sok fontos következtetés vonható le.[1] A tanulmányban bemutatásra kerül a gráf alapú tanuló módszerek egyik központi fogalma, a Laplace-mátrix, melynek röviden ismertetjük néhány fontos tulajonságát. Ezután három konkrét, gráfokat használó algoritmust mutatunk be: a spektrális klaszterezést, a Laplace típusú regularizált legkisebb négyzetek módszerét és a címkepropagálást. Az algoritmusok ismertetése után azok működését szemléltetjük kisebb adathalmazokon. A tanulmányban bemutatottak megértéséhez mindössze alapvető matematikai és a gépi tanulással kapcsolatos fogalmak ismerete szükséges.

2. Jelölésrendszer

Jelen tanulmányban az alábbiakban bemutatott jelölésrendszert használjuk. A skalárokat egyszerű római vagy görög betűkkel jelöljük, például a, b, β, λ. A félkövérrel szedett kisbetűs entitások vektorokat ($\boldsymbol{u}, \boldsymbol{x}, \boldsymbol{y}, \ldots$), a nagybetűsek pedig mátrixokat jelölnek ($\boldsymbol{A}, \boldsymbol{L}, \boldsymbol{X}, \ldots$). Az $\boldsymbol{A}$ mátrix transzponáltját $\boldsymbol{A}'$-vel jelöljük. Az $\mathbf{1}$ vektor az 1-eseket tartalmazó vektort jelöli, $\boldsymbol{I}$ pedig az egységmátrixot. Ezek méretét nem jelöljük, a kontextus alapján az mindig kiolvasható lesz. A tanulmányban használt $\|\cdot\|$ norma az euklideszi normát jelenti.

A bemutatásra kerülő algortimusokban címkézetlen és címkézett adatokkal fogunk dolgozni. Címkézetlen adataink klaszterezés esetén vannak, címkézettek pedig osztályozási feladatokban jelennek meg. Adataink d-dimenziós valós vektorok, és ezeket általában $\boldsymbol{x}$-szel, illetve adott sorrend esetén $\boldsymbol{x}_i$-vel jelöljük, $\boldsymbol{x}, \boldsymbol{x}_i \in \mathbb{R}^d$, $i \in \{1, 2, \ldots, N\}$. Az adatainkat sorba rendezve, majd egy mátrix oszlopaiba helyezve megkapjuk a $d \times N$ méretű $\boldsymbol{X}$ adatmátrixot. Címkézett adatok esetén az adatok számát ℓ-lel jelöljük, viszont a bemutatott osztályozó algoritmusok félig-felügyelt módszerek, ami azt jelenti, hogy címkézett adataink mellett címkézetlen adatokat is használunk tanuláskor. Az adathalmaz címkézetlen részhalmazának méretét u-val jelöljük, a teljes tanulási adathalmaz mérete tehát $N = \ell + u$. A tanulmányban csak bináris (azaz kétosztályos)

[1] Habár jelen tanulmány csak irányítatlan gráfokkal foglalkozik, léteznek irányított gráfokon alapuló módszerek is. Egy példa erre a [10] cikkben bemutatott irányított páros gráfokon alapuló félig-felügyelt algoritmus.

osztályozási feladatokról lesz szó, ezért a címkézett adatokhoz bináris címkék társulnak – ezeket y_i-vel jelöljük, $y_i \in \{-1, 1\}, i = 1, 2, \ldots, \ell$.

3. Adatgráfok

Az adatok felett egy $G = (V, E, W)$ irányítatlan gráfot definiálunk: az adatpontok alkotják a V halmazt, az E élek pedig az adatokat kötik össze. Az adatokhoz egy hasonlósági mértéket választunk: ez szabja meg, hogy van-e él két pont között, illetve ez adja meg a kötés *erősségét*. A kötés erőssége, vagy az él súlya ($W(\boldsymbol{x}, \boldsymbol{z})$) a két adat hasonlóságát, közelségét fejezi ki. A hasonlósági mértéktől megköveteljük, hogy pozitív és szimmetrikus legyen: $W(\boldsymbol{x}, \boldsymbol{z}) \geq 0, W(\boldsymbol{x}, \boldsymbol{z}) = W(\boldsymbol{z}, \boldsymbol{x}), \forall \boldsymbol{x}, \boldsymbol{z} \in V$.

Két pont hasonlóságát jelölhetjük a pontok vagy, ha sorba rendezzük a pontokat, akkor a pontok indexeinek segítségével:

$$W(\boldsymbol{x}_i, \boldsymbol{x}_j) =: w_{ij}.$$

A $\boldsymbol{W}$ mátrix a súlyozott szomszédsági mátrixot jelöli, melyet hasonlósági mátrixnak vagy egyszerűen *súlymátrixnak* is nevezünk,

$$\boldsymbol{W} = (w_{ij})_{i,j=1,\ldots,N}.$$

Egy pont fokszámát a szomszédos élek súlyainak összege adja meg, és az $\boldsymbol{x}_i$ pont fokszámát d_i-vel jelöljük:

$$d_i = \sum_{j=1}^{N} w_{ij}.$$

A *fokszámmátrix* egy olyan diagonálmátrix, melynek főátlóján a pontok fokszámai találhatóak:

$$\boldsymbol{D} = \operatorname{diag}(\boldsymbol{W}\boldsymbol{1}).$$

3.1. Gráfok szerkesztése

Az adatok felett értelmezett gráf megszerkesztésének módja legalább annyira fontos, mint maga a tanuló algoritmus, amit alkalmazni fogunk. A gráf felépítéséhez szükségünk lesz egy hasonlósági mértékre (vagy távolságfüggvényre), illetve egy *vágási* mechanizmusra, amely megmondja, hogy milyen esetekben kössünk, és milyen esetekben ne kössünk össze éllel két pontot. A legtöbbet használt hasonlósági függvény a Gauss-féle hasonlóság, amely a következő módon értelmezett:

$$k_{\mathrm{G}}(\boldsymbol{x}, \boldsymbol{z}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{z}\|^2}{2\sigma^2}\right),$$

ahol a σ paraméter egyfajta *szomszédsági határt* szab meg: minél nagyobb ez az érték, annál nagyobb hasonlóságot eredményez a függvény távolabbi pontok esetén. Látható, hogy a Gauss-féle hasonlósági függvény pozitív és szimmetrikus, illetve még *normalizált* is, azaz minden hasonlósági érték a $[0, 1]$ intervallumba esik. Minél nagyobb a fügvény által visszatérített érték, az annál erősebb hasonlóságot jelent.

A vágási mechanizmus szempontjából a következő típusú gráfokat különböztetjük meg:

- k-legközelebbi szomszéd gráfok: Akkor kötjük össze az $\boldsymbol{x}_i$ és $\boldsymbol{x}_j$ pontokat, hogyha $\boldsymbol{x}_j$ az $\boldsymbol{x}_i$ k legközelebbi szomszédai között van. Ez viszont irányított gráfot eredményez, vagyis a súlymátrix nem lesz szimmetrikus. Ennek szimmetrizálására a következő két módszer közül választhatunk:

(i) az $\boldsymbol{x}_i$ és $\boldsymbol{x}_j$ pontot akkor kötjük össze, ha $\boldsymbol{x}_j$ $\boldsymbol{x}_i$ k legközelebbi szomszédja között van, *vagy* fordítva ($\boldsymbol{x}_i$ $\boldsymbol{x}_j$ k legközelebbi szomszédja között van);

(ii) az $\boldsymbol{x}_i$ és $\boldsymbol{x}_j$ pontot akkor kötjük össze, ha $\boldsymbol{x}_j$ $\boldsymbol{x}_i$ k legközelebbi szomszédja között van, *és* fordítva.

E két gráf közül az elsőt egyszerűen *k-legközelebbi szomszéd gráfnak* nevezzük, míg a másodikra a *kölcsönös k-legközelebbi szomszéd gráf* elnevezéssel hivatkozunk.

- ε-szomszédsági gráfok: Egy ε-szomszédsági gráfban akkor kötünk össze két pontot, hogyha azok távolsága kisebb egy előre meghatározott ε küszöbértéknél, vagy hasonlóan, ha azok hasonlósága nagyobb egy előre meghatározott ε küszöbértéknél.[2]

- teljes gráfok: Ez a legegyszerűbb eset, amikor is nem használunk semmilyen vágást, minden, a hasonlósági mérték által meghatározott súlyt meghagyunk.

A gráfok szerkesztésének módszere nagyban befolyásolja a tanuló algoritmus kimenetét. El kell tehát döntenünk, hogy a fent említettek közül melyik gráfot használjuk: valamelyik k-legközelebbi szomszéd gráfot, ε-szomszédsági vagy a teljes gráfot? Ugyanakkor azt is el kell döntetünk, hogy súlyozzuk-e az éleket vagy sem, súlyozás esetén pedig meg kell választanunk a hasonlósági mértéket, illetve annak paramétereit. Ezek tárgyalására itt nem térünk ki, a paraméterek megválasztásának szempontjaihoz ajánljuk a [9] munkát.

4. A Laplace-mátrix

A Laplace-mátrix – amint azt az elkövetkezendő részekben látni fogjuk – fontos szerepet tölt be a gráf alapú tanuló algoritmusokban. Tulajdonképpen három algoritmust fogunk a későbbiekben részletesen bemutani, és mindhárom algoritmusban fel fog bukkanni ez a mátrix. Ez természetesen nem jelenti azt, hogy minden gráf alapú módszer a Laplace-mátrixot használja, de sok ezen alapszik, még akkor is, hogyha ez első pillantásra nem látható.

A Laplace-mátrix definíció szerint:

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W},$$

ahol $\boldsymbol{W}$ és $\boldsymbol{D}$ a már ismert súly-, illetve fokszámmátrix. A Laplace-mátrix fontosabb tulajdonságai:

1. Minden $\boldsymbol{f} \in \mathbb{R}^N$ vektorra:
$$\boldsymbol{f}'\boldsymbol{L}\boldsymbol{f} = \frac{1}{2}\sum_{i,j=1}^{N} w_{ij}(f_i - f_j)^2.$$

2. $\boldsymbol{L}$ szimmetrikus és pozitív szemidefinit.

3. $\boldsymbol{L}$ legkisebb sajátértéke 0, és a hozzá tartozó sajátvektor az $\mathbf{1} = [1, 1, \ldots, 1]'$.

4. $\boldsymbol{L}$ N darab nemnegatív valós sajátértékkel rendelkezik, $0 = \lambda_1 \leq \ldots \leq \lambda_N$.

[2]Ebben az esetben a vágások után sokszor elhagyjuk a mérték által meghatározott hasonlóságokat, azaz a súlyozatlan gráfot tekintjük.

A mátrix egy másik érdekes és fontos tulajdonsága, hogy pontosan annyi zérus sajátértéke van, ahány összefüggő komponensből áll a gráf.[3] Ha a gráf több összefüggő komponensből áll, a Laplace-mátrix felírható blokkdiagonális alakban, ahol minden blokk az illető komponens Laplace-mátrixa lesz.

A Laplace-mátrixnak léteznek más változatai is, nevezetesen a véletlen bolyongás típusú (*random walk*) és a szimmetrikus normalizált (*symmetric*) Laplace-mátrix:

$$\begin{aligned} \boldsymbol{L}_{\text{rw}} &= \boldsymbol{D}^{-1}\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{W}, \\ \boldsymbol{L}_{\text{sym}} &= \boldsymbol{D}^{-1/2}\boldsymbol{L}\boldsymbol{D}^{-1/2} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}. \end{aligned}$$

E változatok is a Laplace-mátrixéhoz hasonló tulajdonságokkal rendelkeznek, éspedig:

1. Minden $\boldsymbol{f} \in \mathbb{R}^N$ vektorra:

$$\boldsymbol{f}'\boldsymbol{L}_{\text{sym}}\boldsymbol{f} = \frac{1}{2}\sum_{i,j=1}^{N} w_{ij}\left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}}\right)^2.$$

2. λ $\boldsymbol{L}_{\text{rw}}$ sajátértéke $\boldsymbol{u}$ sajátvektorral akkor és csak akkor, ha λ $\boldsymbol{L}_{\text{sym}}$ sajátértéke $\boldsymbol{w} = \boldsymbol{D}^{1/2}\boldsymbol{u}$ sajátvektorral.

3. λ $\boldsymbol{L}_{\text{rw}}$ sajátértéke $\boldsymbol{u}$ sajátvektorral akkor és csak akkor, ha λ és $\boldsymbol{u}$ az $\boldsymbol{L}\boldsymbol{u} = \lambda\boldsymbol{D}\boldsymbol{u}$ általánosított sajátfeladat megoldása.

4. 0 az $\boldsymbol{L}_{\text{rw}}$ sajátértéke az $\mathbf{1}$ sajátvektorral, és 0 az $\boldsymbol{L}_{\text{sym}}$ sajátértéke $\boldsymbol{D}^{1/2}\mathbf{1}$ sajátvektorral.

5. $\boldsymbol{L}_{\text{rw}}$ és $\boldsymbol{L}_{\text{sym}}$ pozitív szemidefinit mátrixok, melyek d nemnegatív valós sajátértékkel rendelkeznek, $0 = \lambda_1 \leq \ldots \leq \lambda_N$.

Megjegyezzük, hogy $\boldsymbol{L}_{\text{rw}}$ és $\boldsymbol{L}_{\text{sym}}$ ugyanannyi zérus sajátértékkel rendelkezik, mint $\boldsymbol{L}$.

5. Klaszterezés

5.1. Minimális vágatok és spektrális klaszterezés

A klaszterezés felügyelet nélküli tanulást jelent, azaz nincsenek tanulási példáink, melyek megmondják, adott bemenetre mi legyen a kimenet. A klaszterezés adott pontok egymástól minél jobban elkülöníthető, *homogén* csoportokba való szervezését jelenti, melyen belül az adatok *jobban hasonlítanak* egymáshoz – ezeket a csoportokat nevezzük klasztereknek. Ha adott egy összefüggő irányítatlan súlyozott gráf, akkor a legkézenfekvőbb ötlet megkeresni azokat a részeket, melyek a *leglazább* kapcsolatban állnak a gráf más részeivel. Bináris esetben ez a gráf két részre való bontását/vágását jelenti: ha A és $\overline{A}$ jelöli a V halmaz egy ilyen diszjunkt felbontását, akkor ez megfogalmazható az A és $\overline{A}$ halmazok közötti hasonlóságok összegének minimalizálásával:

$$\operatorname*{argmin}_{A} \sum_{\boldsymbol{x}\in A, \boldsymbol{z}\in\overline{A}} W(\boldsymbol{x}, \boldsymbol{z}). \tag{1}$$

Ha az A és $\overline{A}$ halmazokat, vagyis a V felbontását az $\boldsymbol{y} \in \{-1, +1\}^N$ vektorral jelöljük, ahol a -1 *címke* az egyik, a $+1$ pedig a másik klaszterbe való tartozást jelenti, akkor ezt felhasználva (1)

[3] A tulajdonságok bizonyításához lásd a [9] munkát.

minimalizálandó kifejezése felírható a következőképpen:[4]

$$\begin{aligned}\frac{(\boldsymbol{y}+1)'}{2}\boldsymbol{W}\frac{(-\boldsymbol{y}+1)}{2} &= -\frac{1}{4}\boldsymbol{y}'\boldsymbol{W}\boldsymbol{y}+\frac{1}{4}\mathbf{1}'\boldsymbol{W}\mathbf{1}\\ &= \frac{1}{4}\left(\boldsymbol{y}'\boldsymbol{D}\boldsymbol{y}-\boldsymbol{y}'\boldsymbol{W}\boldsymbol{y}\right)=\frac{1}{4}\boldsymbol{y}'(\boldsymbol{D}-\boldsymbol{W})\boldsymbol{y}\\ &= \frac{1}{4}\boldsymbol{y}'\boldsymbol{L}\boldsymbol{y}.\end{aligned}$$

Látható, hogy a feladat felírható diszkrét optimalizálási feladatként, amely polinomiális időben megoldható az Edmonds–Karp algoritmussal [4], viszont ez a megoldás az esetek többségében az egyik halmazban nagyon kevés pontot fog tartalmazni – akár egyetlen pontot, hogyha például létezik olyan csúcs a gráfban, mely csak egyetlen csúccsal van összekötve egy kisebb súlyú éllel –, azaz a halmazok mérete nem lesz kiegyenlített. Emiatt behozzuk azt a megkötést, hogy a halmazok mérete legyen egyenlő, azaz teljesüljön az $\boldsymbol{y}'\mathbf{1}=1$ feltétel. Ezáltal viszont a feladat NP-nehéz feladattá válik [5], amit relaxációval oldunk meg: nem követeljük meg a diszkrét, $\{-1,1\}$ feletti megoldást, hanem áthelyezzük a feladatot a valós térbe, majd a végén zérusnál *küszöböljük* a kapott értékeket, így alakítva vissza az eredményt diszkrét megoldássá. A feladat így a következő folytonos optimalizálási feladattá válik:

$$\begin{aligned}\underset{\boldsymbol{y}\in\mathbb{R}^N}{\operatorname{argmin}} \quad & \boldsymbol{y}'\boldsymbol{L}\boldsymbol{y}\\ & \text{ú.h. } \boldsymbol{y}'\mathbf{1}=0 \text{ és } \boldsymbol{y}'\boldsymbol{y}=1,\end{aligned} \tag{2}$$

ahol a második megkötés a $\mathbf{0}$ megoldás elkerülése miatt jelent meg. Ez átírható az

$$\begin{aligned}\underset{\boldsymbol{y}\in\mathbb{R}^N}{\operatorname{argmin}} \quad & \frac{\boldsymbol{y}'\boldsymbol{L}\boldsymbol{y}}{\boldsymbol{y}'\boldsymbol{y}}\\ & \text{ú.h. } \boldsymbol{y}'\mathbf{1}=0\end{aligned} \tag{3}$$

alakra, melynek megoldása az $\boldsymbol{L}$ második legkisebb sajátértékéhez tartozó sajátvektor lesz[5] [9]. A kapott valós vektort 0-nál küszöböljük: zérusnál kisebb érték az egyik, nálánál nagyobb pedig a másik klaszterbe való tartozást fogja jelenteni.

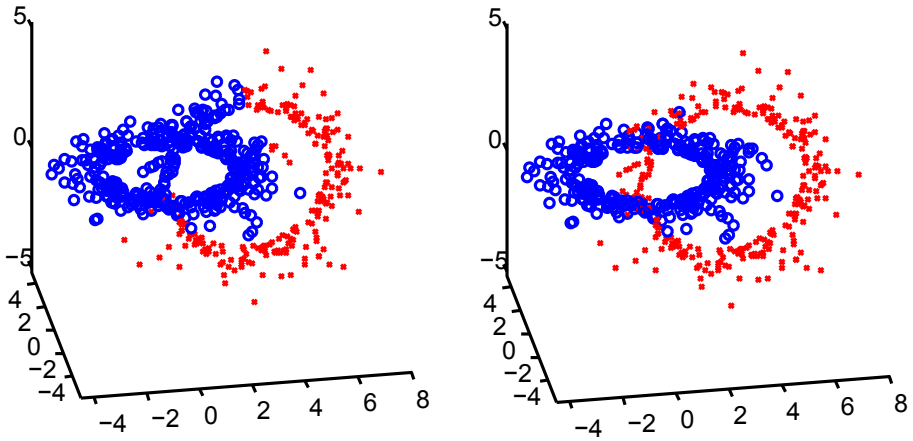
A minimális vágat helyett sokszor *normalizált* minimális vágatot [7] használunk, amely előnyösebb tulajdonságokkal rendelkezik:

$$\underset{A}{\operatorname{argmin}}\frac{\sum_{\boldsymbol{x}\in A,\boldsymbol{z}\in\overline{A}}W(\boldsymbol{x},\boldsymbol{z})}{\sum_{\boldsymbol{x}\in A,\boldsymbol{v}\in V}W(\boldsymbol{x},\boldsymbol{v})}+\frac{\sum_{\boldsymbol{x}\in A,\boldsymbol{z}\in\overline{A}}W(\boldsymbol{x},\boldsymbol{z})}{\sum_{\boldsymbol{z}\in\overline{A},\boldsymbol{v}\in V}W(\boldsymbol{z},\boldsymbol{v})}.$$

Az optimalizálási feladatot ebben az esetben is – az előbbihez hasonló módon – folytonos térbe helyezzük és ott oldjuk meg, mivel a diszkrét optimalizálási feladat ez esetben NP-teljes [7]. Ebben az esetben a megoldás az $\boldsymbol{L}_{\text{sym}}$ második legkisebb sajátértékéhez tartozó sajátvektor lesz, pontosabban $\boldsymbol{D}^{-1/2}\boldsymbol{v}_2$, ahol $\boldsymbol{v}_2$ az illető sajátvektort jelöli – ezt fogjuk majd zérusnál küszöbölni [9].

[4] A levezetésben felhasználtuk az alábbi azonosságokat: $\mathbf{1}'\boldsymbol{W}\mathbf{1}=\mathbf{1}'\boldsymbol{D}\mathbf{1}=\boldsymbol{y}'\boldsymbol{D}\boldsymbol{y}$.

[5] Ha az $\frac{\boldsymbol{y}'\boldsymbol{L}\boldsymbol{y}}{\boldsymbol{y}'\boldsymbol{y}}$ kifejezést (Rayleigh-együttható) akarjuk minimalizálni, akkor ennek optimumpontja az $\boldsymbol{L}$ legkisebb sajátértékéhez tartozó sajátvektorban lesz [6]. Az $\boldsymbol{y}'\mathbf{1}=0$ megkötés viszont ekkor nem teljesül, mivel láttuk, hogy $\boldsymbol{L}$ legkisebb sajátvektora az $\mathbf{1}$ vektor 0 sajátértékkel. Tekintsük a Rayleigh-együttható következő tulajdonságát [6, 7]: ha $\boldsymbol{M}$ valós szimmetrikus mátrix és ha megköveteljük, hogy $\boldsymbol{y}$ ortogonális legyen a $j-1$ legkisebb $\boldsymbol{u}_1,\boldsymbol{u}_2,\ldots,\boldsymbol{u}_{j-1}$ sajátvektorra, akkor $\frac{\boldsymbol{y}'\boldsymbol{M}\boldsymbol{y}}{\boldsymbol{y}'\boldsymbol{y}}$ a következő legkisebb $\boldsymbol{u}_j$ sajátvektorban veszi fel minimumát.



(a) (b)

1. ábra. Spektrális klaszterezés szemléltetése egy kis adathalmazon. Teljes gráfot használtunk Gauss-féle hasonlósági mértékkel és (a) $1/(2\sigma^2) = 0,5$ (88, 17%-os helyes hozzárendelés), illetve (b) $1/(2\sigma^2) = 2$ paraméterrel (100%-os helyes hozzárendelés).

Mivel a megoldást a Laplace-mátrix egyik sajátvektora szolgáltatja, ezért ezt a típusú klaszterezést *spektrális klaszterezésnek* nevezzük. A spektrális klaszterezésnek természetesen létezik több klaszterre kiterjesztett változata is – ezen változatokról például a [9] munkában olvashatunk. Itt nem térünk ki részletekbe menően ezek tárgyalására, mindössze annyit jegyzünk meg, hogy a többklaszteres esetben szintén a Laplace-mátrix sajátvektorai szolgáltatják a megoldást, ekkor viszont több sajátvektor is a megoldás része lesz. A sajátvektorokat a pontok egy kisebb dimenziós térbe való leképezésére használjuk, majd ebben a térben a k-közép klaszterező algoritmust használjuk a *célklaszterek* meghatározásához.

Az 1. ábrán a spektrális klaszterezés kimenetét láthatjuk egy kis adathalmazon. Az adathalmaz összesen 600 pontot tartalmaz, melyből 322 pont az egyik, 278 pedig a másik klaszterben található. A két klaszter a két láncszerűen összekapcsolt pontfelhőt jelenti; a pontok hovatartozását (azaz az algoritmus kimenetét) kék körökkel és piros x-ekkel jelöltük. Az adatgráf mindkét esetben teljes, a súlyokat a Gauss-féle hasonlósági függvénnyel adtuk meg különböző paraméterrel, ezek eredményei láthatók az (a) és (b) rajzokon.

6. Osztályozás

6.1. Laplace típusú regularizált legkisebb négyzetek módszere

Az osztályozás felügyelt tanulást jelent, vagyis a rendszer (adat, címke) tanulási példákon keresztül tanulja meg, hogy adott bemenetre (adat) mi legyen a kimenet (címke). A klaszterezéssel ellentétben – ahol sokszor a klaszterek számát sem ismerjük, ennek meghatározása is a feladat része – a csoportok száma véges. Ezeket a csoportokat osztályoknak nevezzük.

Egy elterjedt osztályozási, illetve *regressziós* módszer[6] a legkisebb négyzetek módszere [2]. A legkisebb négyzetek módszere – bináris osztályozási esetben – a pontokat úgy próbálja meg

[6]Felügyelt tanuláskor lehetnek valós címkéink is, ekkor regresszióról beszélünk. A legkisebb négyzetek módszere valójában egy regressziós metódus, viszont osztályozásra is könnyedén alkalmazható.

szétválasztani egy hipersíkkal[7], hogy az a legkisebb négyzetes hibát eredményezze az ismert címkékre:

$$\operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{\ell} \sum_{i=1}^{\ell} \left(\boldsymbol{w}'\boldsymbol{x}_i - y_i\right)^2 = \frac{1}{\ell} \left\|\boldsymbol{X}'\boldsymbol{w} - \boldsymbol{y}\right\|^2. \tag{4}$$

Ehhez az objektív függvényhez általában hozzátoldunk egy *regularizációs* tagot[8] is, mert az $\boldsymbol{X}\boldsymbol{X}'$ mátrixtól nem tudjuk megkövetelni, hogy mindig invertálható legyen. Így a (4) függvényéhez hozzáadva a $\lambda\|\boldsymbol{w}\|^2$ tagot, majd ezt $\boldsymbol{w}$ szerint deriválva és egyenlővé téve zéróval kapjuk, hogy

$$\boldsymbol{w} = \left(\boldsymbol{X}\boldsymbol{X}' + \lambda\ell\boldsymbol{I}\right)^{-1} \boldsymbol{X}\boldsymbol{y}.$$

A döntési függvényünk, vagyis a pontokhoz címkét rendelő függvényünk ez esetben

$$f(\boldsymbol{x}) = \operatorname{sgn}(\boldsymbol{w}'\boldsymbol{x})$$

lesz. Ez egy *induktív* tanuló rendszer, azaz a döntési függvény *általánosan* alkalmazható bármilyen pontra. Ezzel szemben az ez után bemutatásra kerülő, címkepropagálás nevet viselő algoritmus egy másfajta, ún. *transzduktív* tanulási módszert ír le.

A gráf alapú vagy Laplace típusú legkisebb négyzetek módszerében egy *jobban* szétválasztó hipersík elérése érdekében címkézetlen pontokat is bevonunk az optimalizálási feladatba. Az így kapott módszert félig-felügyelt tanuló metódusnak nevezzük, mert címkézett és címkézetlen pontokat egyaránt felhasznál. A félig-felügyelt tanuló rendszerek egyik alapfeltevése az úgynevezett *simasági* feltevés (*smoothness assumption*): ha két pont közel áll egymáshoz, azaz hasonlóságuk nagy, az osztályozó kimenete nagy valószínűséggel ugyanaz lesz a két pontra [3]. Ezt a következőképpen vihetjük be a feladatba. Tekintsünk először egy hasonlósági mértéket. Az ismert címkék függvényében felírt négyzetes hibához hasonlóan most az osztályozó kimenetei közötti négyzetes hibát vesszük minden pontpárra, majd ezt a hasonlósággal skálázzuk – ebből adódik hibafüggvényünk második része:

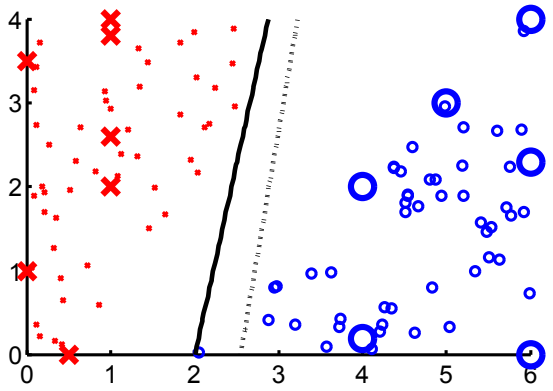
$$\operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{\ell} \left\|\boldsymbol{X}'_{\ell}\boldsymbol{w} - \boldsymbol{y}\right\|^2 + \lambda\boldsymbol{w}'\boldsymbol{w} + \frac{\mu}{2N^2} \sum_{i,j=1}^{N} a_{ij} \left(\boldsymbol{w}'\boldsymbol{x}_i - \boldsymbol{w}'\boldsymbol{x}_j\right)^2,$$

ahol N a címkézett és a címkézetlen pontok együttes számát jelöli, $N = \ell + u$. Ezt a Laplace típusú regularizált legkisebb négyzetek módszerének nevezzük [1]. Az egyszerűbb és kompakt jelölés érdekében osszuk fel a teljes adatmátrixot két részre, a címkézett és címkézetlen pontok vektoraira, melyeket jelöljünk rendre $\boldsymbol{X}_{\ell}$, illetve $\boldsymbol{X}_u$-val. A teljes adatmátrix tehát ezek konkatenációjából áll elő, $\boldsymbol{X} = [\boldsymbol{X}_{\ell}\, \boldsymbol{X}_u]$. Ha az új, utolsó tagban – az egyszerűbb jelölés érdekében – elvégezzük az $\boldsymbol{f}_i := \boldsymbol{w}'\boldsymbol{x}_i$ és $\boldsymbol{f} := \boldsymbol{X}'\boldsymbol{w}$ helyettesítéseket, akkor a következőket vehetjük észre:

$$\begin{aligned}
\sum_{i,j=1}^{N} a_{ij} \left(\boldsymbol{f}_i - \boldsymbol{f}_j\right)^2 &= \sum_{i,j=1}^{N} a_{ij} \left(\boldsymbol{f}_i^2 - 2\boldsymbol{f}_i\boldsymbol{f}_j + \boldsymbol{f}_j^2\right) \\
&= 2\sum_{i,j=1}^{N} a_{ij}\boldsymbol{f}_i^2 - 2\sum_{i,j=1}^{N} a_{i,j}\boldsymbol{f}_i\boldsymbol{f}_j \\
&= 2\sum_{i,j=1}^{N} \boldsymbol{f}_i^2 d_i - 2\boldsymbol{f}'\boldsymbol{A}\boldsymbol{f} \\
&= 2\boldsymbol{f}'\boldsymbol{D}\boldsymbol{f} - 2\boldsymbol{f}'\boldsymbol{A}\boldsymbol{f} = 2\boldsymbol{f}'\boldsymbol{L}\boldsymbol{f},
\end{aligned} \tag{5}$$

[7] A hipersík egy n-dimenziós tér $(n-1)$-dimenziós altere, két dimenzióban például egy egyenes, három dimenzióban egy sík.

[8] A regularizáció valamilyen többletinformáció, követelmény bevezetését jelenti egy adott problémába, a feladat megoldhatóvá tételének érdekében.



2. ábra. A regularizált legkisebb négyzetek módszerének szemléltetése egy kis adathalmazon. A szaggatott, illetve a folytonos vonal a kapott szétválasztó hipersíkot jelöli a regularizált legkisebb négyzetek módszerével, illetve annak Laplace típusú kiterjesztésével. A megcímkézés szempontjából a rajz a félig-felügyelt eset kimenetét mutatja. Ebben az esetben hasonlóságként skalárszorzatot használtunk szimmetrikus normalizált Laplace-mátrixszal és $\mu = 200$ paraméterrel. A λ együttható értékét mindkét esetben $0,001$-re állítottuk.

ahol újra megjelent a hasonlósági gráf Laplace-mátrixa. Visszahelyettesítve $\boldsymbol{f}$-et, minimalizálandó függvényünk a következőképpen alakul:

$$\underset{\boldsymbol{w}}{\operatorname{argmin}} \frac{1}{\ell} \left\| \boldsymbol{X}'_\ell \boldsymbol{w} - \boldsymbol{y} \right\|^2 + \lambda \boldsymbol{w}'\boldsymbol{w} + \frac{\mu}{N^2} \boldsymbol{w}'\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}'\boldsymbol{w}.$$

Innen – az előbbi összefüggést $\boldsymbol{w}$ szerint deriválva majd egyenlő téve zérussal – kapjuk, hogy

$$\boldsymbol{w} = \left(\boldsymbol{X}_\ell \boldsymbol{X}'_\ell + \lambda \ell \boldsymbol{I} + \frac{\mu \ell}{N^2} \boldsymbol{X}\boldsymbol{L}\boldsymbol{X}' \right)^{-1} \boldsymbol{X}_\ell \boldsymbol{y}.$$

A legkisebb négyzetek módszere, amint az a fentiekben látható volt, egy szétválasztó hipersíkot keres az adatokhoz úgy, hogy a négyzetes hiba minimális legyen. A Laplace típusú regularizált legkisebb négyzetek módszere pedig ezt az alapötletet terjeszti ki úgy, hogy a szétválasztó hipersíkot a pontok közötti hasonlóságok is befolyásolják. Ha a hipersíkot csak a normálvektorral definiáljuk, akkor mindig egy az origón átmenő hipersíkot kapunk. Viszont jelen esetben nem csak ilyen hipersíkok jöhetnek számításba, ezért az általános egyenlet minden paraméterét meg kell határoznunk, vagyis döntési függvényünk $\boldsymbol{w}'\boldsymbol{x} + b$ alakú. Hogy ne bonyolítsuk el az optimalizálási feladatot egy új b paraméter bevezetésével, az adatainkat terjesszük ki egy új *konstans* dimenzióval: $\begin{bmatrix} \boldsymbol{X} \\ \boldsymbol{1}' \end{bmatrix}$, így az objektív függvényen nem kell változtatnunk.

A 2. ábrán a regularizált legkisebb négyzetek módszerének és annak Laplace-típusú kiterjesztésének kimenetét láthatjuk egy kis adathalmazon. A tanuló halmaz összesen 100 pontot tartalmaz, melyből 13-at (7 pozitív, 6 negatív példa) tartalmaz a címkézett és 97-et (49 pozitív, 48 negatív példa) a címkézetlen halmaz. Habár mindkét hipersíkot jelöltük az ábrán, a rajz a Laplace típusú regularizált legkisebb négyzetek módszerének (100%-osan pontos) kimenetét mutatja: a piros x-ek a pozitív, a kék körök a negatív pontokat jelölik, ahol a nagyobb méretű jelek a címkézett pontokat jelentik.

6.2. Címkepropagálás

A félig-felügyelt tanulás egy tipikus példája a címkepropagálás [12]. Az adatokon a már látott módon egy gráfot építünk, majd a címkéket a tanulási adatoktól a címkézetlen adatok felé *propagáljuk* a kapcsolatok erősségétől függően.

A címkék propagálásának megvalósítása érdekében egy átmenet-valószínűség mátrixot építünk a hasonlóságok segítségével. Ha a hasonlósági mátrixot a $\boldsymbol{W}$ szimbólummal jelöljük, az átmenet-valószínűség mátrixot pedig $\boldsymbol{P} = (p_{ij})_{i,j=1,\ldots,N}$-vel, akkor a valószínűségeket a következő módon számítjuk ki:

$$p_{ij} = \frac{w_{ij}}{\sum_{k=1}^{N} w_{ik}}.$$

Ez röviden a $\boldsymbol{P} = \boldsymbol{D}^{-1}\boldsymbol{W}$ összefüggéssel is felírható, ahol $\boldsymbol{D}$ a már ismert fokszámmátrix.

Az algoritmust most is csak bináris osztályozásra adjuk meg, viszont a feladat nagyon egyszerűen átírható többosztályos esetre [12, 11]. Jelölje a címkék vektorát $\boldsymbol{y} \in \{-1, 1\}^N$, és bontsuk ezt fel két részre: jelölje a felső ℓ elem az ismert címkéket, az alsó rész pedig a címkézetlen adatokét:

$$\boldsymbol{y} =: \begin{bmatrix} \boldsymbol{y}_\ell \\ \boldsymbol{y}_u \end{bmatrix}.$$

Célunk a címkézetlen adatok $\boldsymbol{y}_u$ címkéinek meghatározása. A módszer alapötlete: az i-edik pont címkéje legyen egyenlő az illető pont *bemenő* szomszédainak az átmenet-valószínűségek szerint súlyozott címkéjével. Azaz, minden bemenő szomszédja propagálja a címkéjét az i-edik pontnak az átmenet-valószínűség szerint. Természetesen, kezdetben a címkézetlen pontoknak nincs címkéjük, ellenben ezek is lehetnek szomszédai az i-edik címkézetlen pontnak. A címkézetlen pontoknak választhatunk tetszőleges címkét – akár mindegyiknek 1-et vagy -1-et –, a későbbiekben látni fogjuk, hogy ez nem befolyásolja a végső eredményt – az iterációk során az eredményvektor egy stabil konfigurációhoz konvergál. Tehát legyen

$$y_i = p_{1i}y_1 + p_{2i}y_2 + \ldots + p_{Ni}y_N, \quad i = 1, \ldots, N.$$

Ezt a címkepropagálást mátrix alakban a következőképpen írhatjuk fel az összes pontra:

$$\boldsymbol{y} = \boldsymbol{P}'\boldsymbol{y}. \tag{6}$$

Az algoritmus a következő lépésekből áll:

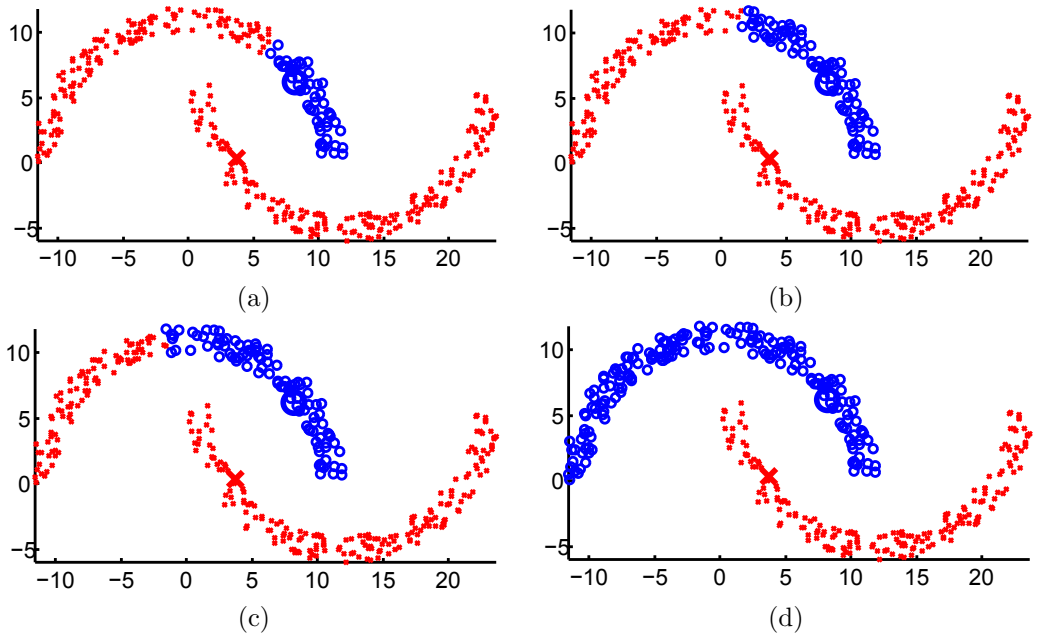
1. $\boldsymbol{y} = \boldsymbol{P}'\boldsymbol{y}$
2. Helyettesítsük vissza az *eredeti*, ismert címkéket $\boldsymbol{y}_\ell$-be.
3. Vissza az 1. lépésre.

A fenti lépéseket addig kell ismételnünk, amíg az $\boldsymbol{y}_u$ vektor konvergálni fog egy stabil megoldáshoz. A konvergencia ellenőrzését például úgy végezhetjük el, hogy megnézzük, mennyit változott az $\boldsymbol{y}_u$ vektor az előző lépésben kapott vektorhoz képest[9], és amint ez egy előre meghatározott kis érték alá esik, megállunk.

Könnyen megmutatható, hogy az algoritmus kimenete nem függ a kezdeti $\boldsymbol{y}_u$ címkék megválasztásától. Ha a címkepropagálást megvalósító (6) rekurzív kifejezést a következőképpen írjuk fel,

$$\begin{bmatrix} \boldsymbol{y}_\ell \\ \boldsymbol{y}_u \end{bmatrix} = \begin{bmatrix} \boldsymbol{T}_{\ell\ell} & \boldsymbol{T}_{\ell u} \\ \boldsymbol{T}_{u\ell} & \boldsymbol{T}_{uu} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_\ell \\ \boldsymbol{y}_u \end{bmatrix},$$

[9] A változást mérhetjük a vektorok közötti euklideszi távolsággal.



3. ábra. A címkepropagálás iteratív változatának szemléltetése egy kis adathalmazon. Az adatgráf ebben az esetben is teljes, a hasonlóságokat a Gauss-féle hasonlósági függvénnyel adtuk meg, $1/(2\sigma^2) = 0,2$ paraméterrel. A négy rajz a címkepropagálás kimenetét mutatja az (a) 50-edik, (b) 100-adik, (c) 200-adik és (d) 300-adik iterációban.

ahol $\boldsymbol{T}$ a $\boldsymbol{P}$ mátrix transzponáltját jelöli, akkor innen kifejezhető az $\boldsymbol{y}_u$,

$$\begin{aligned} \boldsymbol{y}_u &= (\boldsymbol{I} - \boldsymbol{T}_{uu})^{-1} \boldsymbol{T}_{u\ell} \boldsymbol{y}_\ell \\ &= \left(\boldsymbol{I} - \boldsymbol{W}_{uu} \boldsymbol{D}_u^{-1}\right)^{-1} \boldsymbol{W}_{u\ell} \boldsymbol{D}_\ell^{-1} \boldsymbol{y}_\ell, \end{aligned} \tag{7}$$

ahol a $\boldsymbol{W}$ mátrixot a fenti $\boldsymbol{T}$-hez hasonlóan bontottuk fel, a $\boldsymbol{D}$ diagonálmátrixot pedig a $\begin{bmatrix} \boldsymbol{D}_\ell & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{D}_u \end{bmatrix}$ módon. Ha a Laplace-mátrixokat is felbontjuk hasonlóképpen, akkor az előbbi kifejezés felírható ezek függvényében is:

$$\begin{aligned} \boldsymbol{y}_u &= -\boldsymbol{D}_u \boldsymbol{L}_{uu}^{-1} (\boldsymbol{L}_{\mathrm{rw}})'_{\ell u} \boldsymbol{y}_\ell \\ &= -\boldsymbol{D}_u (\boldsymbol{L}_{\mathrm{rw}})_{uu}^{-1} \boldsymbol{D}_u^{-1} (\boldsymbol{L}_{\mathrm{rw}})'_{\ell u} \boldsymbol{y}_\ell. \end{aligned}$$

Ez tulajdonképpen azt jelenti, hogy a címkepropagálás megvalósítható iteratívan a bemutatott háromlépéses algoritmussal, de kiszámíthatjuk a címkéket a (7) összefüggés segítségével is. Mivel (7) mátrixinverziót is tartalmaz, amely köbös bonyolultságú, nagy adathalmazok esetén hatékonyabb lehet az iteratív változat használata.[10]

A 3. ábrán a címkepropagálás iteratív változatának működését szemléltettük egy kis adathalmazon. Az adathalmaz összesen 385 pontot tartalmaz, melyből mindössze kettő címkézett, a maradék 383 pont címkéje ismeretlen. A címkézetlen pontok két különálló felhője 191, illetve 192 pontot tartalmaz. A négy rajzon az algoritmus kimenete látható az iterációszám függvényében.

[10] A címkék csak akkor lesznek meghatározhatók, illetve az algoritmus csak akkor fog konvergálni, hogyha az $\boldsymbol{I} - \boldsymbol{T}_{uu}$ mátrix invertálható. Megjegyezzük, hogy a Gauss-féle hasonlóság használata esetén ez mindig teljesül.

A piros x-ek a poztív, a kék körök a negatív pontokat jelölik, ahol a nagyobb méretű jelek a címkézett pontok.

A címkepropagálás – mint azt már korábban említettük – egy transzduktív tanuló algoritmus. Az ilyen típusú algoritmusok, ellentétben az induktív módszerekkel, nem határoznak meg egy tetszőleges pontra alkalmazható általános függvényt, hanem csak a függvény értékeit adják meg a kérdéses pontokban [8, 3]. A címkepropagálásban tehát egy pont címkéje csak akkor határozható meg, hogyha azt hozzáadjuk a címkézetlen pontok halmazához, és újra kiszámítjuk az összes címkét. A következőkben röviden bemutatjuk a címkepropagálás egy másik változatát, amely jobb tulajdonságokkal rendelkezik. A különbség a már bemutatott módszer és e között mindössze az, hogy a propagálást most az $\boldsymbol{y} = \boldsymbol{P}\boldsymbol{y}$ egyenlettel írjuk le. Ezt azt jelenti, hogy egy pont címkéjét a pont *kimenő* szomszédai határozzák meg,

$$y_i = p_{i1}y_1 + p_{i2}y_2 + \ldots + p_{iN}y_N, \quad i = 1, \ldots, N.$$

Ezzel az egyszerű változtatással azt érjük el, hogy a keresett címkéket megadó explicit kifejezésünk a következőképpen módosul:

$$\boldsymbol{y}_u = (\boldsymbol{I} - \boldsymbol{P}_{uu})^{-1} \boldsymbol{P}_{u\ell}\boldsymbol{y}_\ell = -\boldsymbol{L}_{uu}^{-1}\boldsymbol{L}_{u\ell}\boldsymbol{y}_\ell. \tag{8}$$

Ebben az esetben megfigyelhetjük, hogy az optimalizálási problémát felírhatjuk a következő alakban:

$$\underset{y_i, i=\ell+1,\ldots,N}{\operatorname{argmin}} \frac{1}{2} \sum_{i,j=1}^{N} a_{ij}(y_i - y_j)^2, \tag{9}$$

ahol a_{ij} újfent az i és j-edik pont hasonlóságát jelöli. Az (5) alapján az objektív függvényt felírhatjuk az $\boldsymbol{y}'\boldsymbol{L}\boldsymbol{y}$ alakban, ahonnan a Laplace-mátrix felbontásával az

$$\boldsymbol{y}_u'\boldsymbol{L}_{uu}\boldsymbol{y}_u + 2\boldsymbol{y}_u'\boldsymbol{L}_{u\ell}\boldsymbol{y}_\ell + \boldsymbol{y}_\ell'\boldsymbol{L}_{\ell\ell}\boldsymbol{y}_\ell$$

kifejezéshez jutunk. Ha ennek a deriváltját egyenlővé tesszük zérussal és kifejezzük belőle az $\boldsymbol{y}_u$-t, a következőt kapjuk:

$$\boldsymbol{y}_u = -\boldsymbol{L}_{uu}^{-1}\boldsymbol{L}_{u\ell}\boldsymbol{y}_\ell,$$

amely megegyezik a (8) egyenlettel. A címkepropagálás ezen új változatával fel tudunk írni egy egyszerű induktív függvényt egy *új* pont címkéjének meghatározására. Tételezzük fel, hogy bizonyos címkézetlen pontokra már kiszámítottuk a címkéket. Ekkor egy új $\boldsymbol{x}$ pont a (9) objektív függvényt a következőképpen módosítja:

$$C + \sum_{i=1}^{N} W(\boldsymbol{x}, \boldsymbol{x}_i)(y - y_i)^2,$$

ahol C a (9) objektív függvény értékét jelöli, y pedig az új pont címkéje. Ennek deriváltját egyenlővé téve zérussal y-ra az

$$y = \frac{\sum_{i=1}^{N} W(\boldsymbol{x}, \boldsymbol{x}_i) y_i}{\sum_{i=1}^{N} W(\boldsymbol{x}, \boldsymbol{x}_i)}$$

egyenletet kapjuk, amely alkalmazható tetszőleges $\boldsymbol{x}$ pont címkéjének kiszámítására.

7. Összefoglalás

A tanulmányban bemutattuk a gráf alapú tanulás néhány módszerét, és láthattuk, hogy habár ezek egymástól eltérő, illetve különböző feladatokat megoldó algoritmusok, mindegyikben megjelenik a Laplace-mátrix. Ezért ezt a speciális mátrixot sokszor a gráf alapú tanuló módszerek egyik központi fogalmaként definiálják. Bemutatásra került egy klaszterező algoritmus, egy regressziós módszer, illetve egy transzduktív tanuló algoritmus. Mindhárom módszernél csak a bináris esetet tárgyaltuk, de az algoritmusok viszonylag egyszerűen kiterjeszthetők több klaszterre, illetve osztályra. A cél nem a módszerek részletekbe menő elemzése és vizsgálata volt, hanem inkább egy bevezető nyújtása a gráf alapú gépi tanulási módszerekhez. Ezen módszerek további tanulmányozásához a [9], [11] és [3] munkákat ajánljuk.

Hivatkozások

[1] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[2] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning*. MIT Press, 2006.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 3 edition, 2009.

[5] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Co., 1979.

[6] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 3rd edition, 1996.

[7] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Conf. Computer Vision and Pattern Recognition*, June 1997.

[8] V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[9] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[10] D. Zhou, B. Schölkopf, and T. Hofmann. Semi-supervised learning on directed graphs. In *In NIPS*, pages 1633–1640. MIT Press, 2005.

[11] X. Zhu. *Semi-supervised learning with graphs*. PhD thesis, 2005.

[12] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.