A tananyag előfeldolgozásai a legközelebbi társ módszerhez

doktori (Ph.D.) értekezés

Ketskeméty László

Debreceni Egyetem
Debrecen, 2003

Ezen értekezést a Debreceni Egyetem Matematika doktori program Valószínűségelmélet és matematikai statisztika alprogramja keretében készítettem 2000–2003 között és ezúton benyújtom a Debreceni Egyetem doktori Ph.D. fokozatának elnyerése céljából.

Debrecen, 2003.

.....
Ketskeméty László
jelölt

Tanúsítom, hogy Ketskeméty László doktorjelölt 2000–2003 között a fent megnevezett doktori alprogram keretében irányításommal végezte munkáját. Az értekezésben foglaltak a jelölt önálló munkáján alapulnak, az eredményekhez önálló alkotó tevékenységével meghatározóan hozzájárult. Az értekezés elfogadását javaslom.

Debrecen, 2003.

.....
Dr. Fazekas István
témavezető

Tartalomjegyzék

Köszönetnyílvánítás

Itt szeretnék köszönetet mondani azoknak, akik segítették disszertációm elkészülését. Hálával tartozom Györfi Lászlónak, aki a témaválasztástól kezdve a dolgozat elkészüléséig figyelemmel kísérte munkámat, hasznos tanácsokat adott és végig bíztatott. Megköszönöm Fazekas Istvánnak, témavezetőmnek az anyag igen alapos, gondos átolvasását, javító megjegyzéseit és a termékeny konzultációkat.

Jelölések

$\mathcal{X}$	Az alakzattér
$d, d_E, d_{Cs}, ...$	Az alkalmazott metrika
$\mathcal{Y}$	Az osztályok halmaza
$\phi, \phi_{\underline{A}}, g : \mathcal{X} \to \mathcal{Y}$	Döntésfüggvények
$\mathcal{D}_n$	A tananyag. Összesen n db (alakzatpont, osztály) elempárt tartalmaz
$\mathcal{T}_n$	A tanulópont halmaz, $\mathcal{D}_n$ alakzatpontjaiból (tanulópontok) álló n elemű halmaz
t_i	Az i-edik tanulópont $\mathcal{T}_n$-ből
$class(t)$	A t tanulópont tanítása (osztálya)
x	Egy metrikus pont $\mathcal{X}$-ből, az osztályozandó pont
$t_{NN}(x)$	Az x legközelebbi társa $\mathcal{T}_n$-ben
$t_{FN}(x)$	Az x legtávolabbi társa $\mathcal{T}_n$-ben
$d_{\min}$	Az x és a legközelebbi társának a távolsága
$d_{\max}$	Az x és a legtávolabbi társának a távolsága
$d_{\min}^{(k)}$	A legkisebb távolság a k-adik keresési lépés után
$d_{\max}^{(k)}$	A legnagyobb távolság a k-adik keresési lépés után
$t_{NN}^{(k)}(x)$	Az x legközelebbi társa a k-adik keresési lépés után
$\mathcal{K}, \mathcal{K}_1, \mathcal{K}_2, ...$	Kizárási feltételek
$\mathcal{S}, \mathcal{S}_{mm}, \mathcal{S}_e, ...$	Keresési stratégiák
$\#A$	Az A halmaz elemszáma
$\mathbf{P}$	Valószínűség
$A \setminus B$	Az A és B halmazok különbséghalmaza
$I_{\{A\}}$	Az A esemény indikátor változója
$\underline{x} \in \mathbb{R}^p$	p-dimenziós vektor
$\underline{\underline{A}} \in \mathbb{R}^{p \times q}$	$p \times q$-as mátrix
$\arg\max(\min)$	Az az elem, ahol a maximum (minimum) felvétetik
$G_A(c_A, R_A)$	Az $A \subset \mathcal{X}$ halmazt lefedő minimális gömb
c_A	Az A centrumeleme ($c_A \in A$)
R_A	Az A takaró sugara

Bevezetés

Célkitűzések, személyes motivációk

Az alakfelismerés a matematikai kutatás egyik olyan népszerű területe, melynek eredményeit széleskörűen alkalmazzák napjainkban is. A tematikus folyóiratok nagy száma -pl. IEEE Transactions on Pattern Recognition, Man, Systems and Cybernetics, IEEE Transactions on Information Theory, Pattern Recognition, Pattern Recognition Letters stb.-, konferenciák és könyvek sokasága tanúsítják ennek a témakörnek a fontosságát. Legújabban az internetes adatbányászat és az ujjlenyomatok felismerése a legnépszerűbb alkalmazás, amelyek kapcsán születnek a legértékesebb eredmények. Bár az első alakfelismerési dolgozatok már a hatvanas évek közepén megjelentek, a gyakorlat mindmáig vet fel megoldandó elméleti problémákat. A dolgozatban tárgyalt szűkebb területen is több tucat az utóbbi néhány évben megjelent publikációk száma. Ez egyrészt azt mutatja, hogy van érdeklődés a témakör iránt, másrészt pedig arra is utal, hogy az eddig bemutatott megoldások nem minden alkalmazásnál kielégítőek. A publikációkban bemutatott algoritmusok többsége szorosan kötődik egy-egy konkrét alkalmazáshoz, a megoldásnál kihasználják a probléma speciális adottságait. Ennélfogva más jellegű gyakorlati feladatok megoldásakor nem jelentkezik az előny. Jelen dolgozat módszereinek kidolgozásánál ezért az egyik szempont az volt, lehetőleg általános feltételek mellett lehessen azokat alkalmazni. Másszóval az volt a célkitűzés, hogy ,,absztrakt" alakfelismerési problémák megoldásához legyenek alkalmasak az algoritmusok, mert így a jelentkező előnyök minden konkrét alkalmazásnál meg fognak mutatkozni. A disszertációban egységes jelölésrendszert használva megkíséreljük összefoglani, áttekinteni az utóbbi kb. négy évtizedben publikált tananyagot előkészítő módszereket. Az általunk kidolgozott és javasolt algoritmusok lényege így jobban összevethető lesz a már meglévőkével.

Alakfelismerési problémákkal 1980-ban ismerkedtem meg az Országos Meteorológiai Szolgálatnál, ahol digitális műholdképek elemzéséhez kellett alkalmas módszereket implementálni [42], [43], [44], [45], [46], [47]. A kutatások

eredményeképpen készült el 1984-ben egyetemi doktori értekezésem [41], melynek címe „Digitális műholdképek számítógépes clusterezése" volt. Később, 1986 és 1988 között, már a műegyetemi dolgozóként vettem részt az OMFB által támogatott kutatásban, melynek keretében a minőségbiztosításban alkalmazható alakfelismerési módszereket dolgoztunk ki és implementáltunk számítógépre. Az eredményeket két tanulmányban foglaltuk össze [48], [49].

Az alakfelismerés témaköre

A dolgozat az alakfelismerés témaköréhez tartozó speciális problémákkal foglalkozik. Először röviden felvázoljuk az alakfelismerés problematikáját.

Adott *objektumoknak* egy Ω halmaza, melynek minden eleme egyértelműen véges számú ismert *osztály* valamelyikéhez tartozik. Ha az osztályok számát L-lel jelöljük, az osztályok halmazát megfeleltethetjük az $\mathcal{Y} = \{1, 2, ..., L\}$ halmaznak. Azt, hogy az objektumok melyik osztályba tartoznak, az $Y : \Omega \to \mathcal{Y}$ függvény írja le, melyet *osztálynak* nevezünk. Az Y-t nem ismerjük. Az objektumokat egy X függvénnyel egyértelműen le lehet képezni egy $\mathcal{X}$ halmazra. Az $X : \Omega \to \mathcal{X}$ függvényt *alakzatnak* nevezzük, és ismertnek tételezzük fel. Az $\mathcal{X}$ az *alakzattér*, amely matematikai szempontból Ω-nál jobban modellezhető halmaz. $\mathcal{X}$ egy adott d metrikával lehet metrikus tér, lehet lineáris tér, akár véges dimenziós vektortér, de speciális esetben az $\mathbb{R}^p$ Euklideszi tér is. Általános esetben $\mathcal{X}$ elemeit *pontoknak* nevezzük, de a megfelelő speciális esetekben használjuk majd a vektor, függvény stb. elnevezéseket is.

Az alakfelismerés alapfeladata az, hogy egy ismeretlen $\omega \in \Omega$ objektumról megállapítsuk, hogy melyik osztályhoz tartozik. Ehhez meg fogunk adni egy $\phi : \mathcal{X} \to \mathcal{Y}$, ún. *döntésfüggvényt*, amivel ω-hoz a $\phi(X(\omega))$ osztályt fogjuk rendelni. Az alapfeladat tehát regressziós: az ismert X egy függvényével akarjuk jól megbecsülni Y-t.

Mielőtt a matematikai modellt tárgyalnánk, nézünk néhány alkalmazási példát!

Rendszám-felismerés. Különböző ellenőrzési pontokon kamerák vannak elhelyezve, melyek digitális felvételeket sugároznak a központba az utakon közlekedő autókról. A központi számítógépben egy program a képekről a gépjárművek rendszámait leválasztja, kódolja, majd az archívumban tárolt körözött rendszámok listájában megkísérli azt azonosítani. Ebben a példában a közlekedő gépjárművek az objektumok, az összes rendszámok halmaza az alakzattér, az osztályok száma most $L = 2$. Az egyik osztály a körözött rendszámok osztálya, a másik a nem körözötteké.
megjegyzés: *Néhány további hatósági alkalmazás a digitális útlevélellenőrzés a határállomásokon, az ujjlenyomatkeresés a bűnügyi archívumban, graphologiai elemzés adott kézirat szerzőjének azonosításához, stb.*

Postai küldemények szortírozása. Az objektumok most a postai küldemények (levelek, képeslapok, pénzes utalványok stb.), melyeken kézzel írott vagy gépelt négyjegyű irányítószám található. Egy digitalizáló berendezés az irányítószámokat egy mátrixra képezi le. Ilyen mátrixok halmaza most az $\mathcal{X}$ alakzattér. Az osztályok azok a helységek, ahová a küldeményt továbbítani kell.

Hosszútávú meteorológiai előrejelzés. Az alakfelismerés témaköréhez tartozik a meteorológusok analógiás előrejelzési módszere is. Tegyük fel, hogy most a kiindulási kérdés az, hogy a következő hat hónapban a csapadékösszeg meghalad-e egy adott küszöbértéket, vagy sem? Az osztályok száma tehát most is kettő: a csapadékmennyiség meghaladja a küszöböt (1) illetve nem haladja meg (2). A vizsgált objektumok most az elmúlt évek hasonló időszakaszainak időjárási történései. Az időjárás lefolyását mérések segítségével jellemzik, és azt archívumokban tárolják. Az $\mathcal{X}$ alakzattér egy pontja most egy előző év hőmérséklet, csapadék, légnyomás stb. adatsorainak összessége. A jelen év időjárásának előző hat hónapos lefolyása szintén kiolvasható ebből az archívumból. A meteorológusok összevetik ezt az adattömeget az előző évek hasonló időszakhoz tartozó adattömegeivel és megkeresik azt az évet, melynél a legjobb az egyezés a jelen évivel. Ezek után a kiindulási kérdésre az alapján adják meg a választ, hogy a leginkább hasonló lefolyású múltbeli időszakban később hogyan alakult a csapadékmennyiség. Az analógiás módszer az alakfelismerés legközelebbi társ módszerének felel meg.

Műholdképek pixeleinek azonosítása. A földfelszín pontjai (ezek most az objektumok) különböző osztályokhoz tartoznak: víz, település, erdő, lucerna, búza, kukorica stb. A műhold a világűrből nagyfelbontású digitális fényképeket készít a földfelszínről. A felszín minden pontjának megfelel egy képpont (pixel) ezen a felvételen. Mivel a fényképezőgép több frekvenciatartományban is mér, ezek a pixelek többdimenziós vektorok lesznek. Tehát most az alakzattér vektortér lesz. Különböző erőforráskutatási alkalmazásoknál fontos a pixelek pontos osztálybasorolása. Ez alapján pl. jól megbecsülhető a búza termőterülete (ez lehet alapja a búza termésbecslésnek), egy folyó árvíz-borítása, egy tó nádas-borítottsága, stb.

Az alakfelismerés matematikai modellje

Adott az $(\Omega, \mathcal{F}, \mathbf{P})$ Kolmogorov-féle valószínűségi mező. Ω elemeit *objektumoknak* nevezzük. Az objektumok mindegyike L különböző osztály valamelyikéhez tartozik. Jelölje $\mathcal{Y} = \{1, 2, ..., L\}$ az osztályok halmazát! Legyen $Y : \Omega \to \mathcal{Y}$ egy olyan diszkrét valószínűségi változó, melynek eloszlása: $p_i = \mathbf{P}(Y = i), i =$

$1, 2, ..., L$ (*a'priori eloszlás*). Az Y valószínűségi változó az objektumok *osztálya.* Adott az $(\mathcal{X}, d)$ metrikus tér, melynek alapterét *alakzattérnek* nevezzük. Jelölje az $\mathcal{X}$ nyílt halmazai által generált σ-algebrát $\mathcal{B}$! Legyen az $X : \Omega \to \mathcal{X}$ mérhető leképezés. X-et *alakzatnak* nevezzük. Tételezzük fel, hogy adott egy μ mérték az $(\mathcal{X}, \mathcal{B})$ mérhető téren, melyre abszolút folytonos X eloszlása, vagyis létezik az $f_X : \mathcal{X} \to \mathbb{R}$ Radon-Nykodim derivált: $\mathbf{P}(X \in B) = \int\limits_B f_X(x)\, d\mu(x)$, minden $B \in \mathcal{B}$-re.

Tekintsük most az $Y = i$ eseménynek az X által generált $\mathcal{G}_X = \sigma(X)$ σ-algebrára vett feltételes valószínűségét:

$$\eta_i(X) = \mathbf{P}(Y = i \mid \mathcal{G}_X).$$

Ekkor az (X, Y) pár együttes eloszlása kifejezhető η_i-vel és f_X-szel:

$$\mathbf{P}(X \in B, Y = i) = \int\limits_{\{X \in B\}} \eta_i(X) d\mathbf{P} = \int\limits_B \eta_i(x) \cdot f_X(x)\, d\mu(x).$$

A képletben szereplő $\eta_i(x)$, $i = 1, 2, ..., L$ függvények alkotját az *a'poszteriori eloszlást*, hiszen $\sum\limits_{i=1}^{L} \eta_i(x) = 1$ minden $x \in \mathcal{X}$ esetén.

A $\phi : \mathcal{X} \to \mathcal{Y}$ függvényt *döntésfüggvénynek* nevezzük. Y-t $\phi(X)$-szel szeretnénk minél jobban megbecsülni. Adott egy $\underline{\underline{W}} = (w_{ij})$ $L \times L$-es mátrix, a *veszteségmátrix*, ahol w_{ij} annak a rossz döntésnek a veszteségét jellemzi, amit akkor követünk el, ha $\phi(X) = i$, de $Y = j$. ($\underline{\underline{W}}$ nem feltétlenül szimmetrikus, de általában $w_{ii} = 0$. Pl. nagyobb a keletkezett kár - illetőleg kisebb az elmaradt nyereség-, ha egy gépjárművet tévedésből körözöttnek osztályozunk, mintha egy lopott kocsit a nem körözöttek közé sorolunk.)

A ϕ döntésfüggvény *rizikóján* az

$$R\left(\underline{\underline{W}}, \phi\right) = \sum_{i=1}^{L} \sum_{j=1}^{L} w_{ij} \cdot \mathbf{P}(\phi(X) = i, Y = j) =$$

$$= \sum_{i=1}^{L} \sum_{j=1}^{L} w_{ij} \cdot \int\limits_{\{\phi(X)=i\}} \eta_j(X) d\mathbf{P} = \sum_{i=1}^{L} \sum_{j=1}^{L} w_{ij} \cdot \int\limits_{\{\phi(x)=i\}} \eta_j(x) \cdot f_X(x)\, d\mu(x) =$$

$$= \sum_{i=1}^{L} \int\limits_{\{\phi(x)=i\}} f_X(x) \cdot \sum_{j=1}^{L} w_{ij} \cdot \eta_j(x) d\mu(x)$$

értéket értjük. Amennyiben $\underline{\underline{W}} = \underline{1} \cdot \underline{1}^T - \underline{\underline{E}}$ az anti-egységmátrix (azaz $w_{ij} = 1 - \delta_{ij}$, ahol δ_{ij} a Kronecker szimbólum), $R\left(\underline{1} \cdot \underline{1}^T - \underline{\underline{E}}, \phi\right) = L(\phi) = \mathbf{P}(\phi(X) \neq Y)$ a hibás döntés valószínűségét adja. A rizikót minimalizáló döntésfüggvényt *Bayes-döntésfüggvénynek* nevezzük, és ϕ^*-gal jelöljük. A minimalizált rizikó (illetve hibavalószínűség) az R^* *Bayes-rizikó* (illetve L^* *Bayes-hiba*).

Belátható, hogy

$$\phi^*(x) = i, \text{ ha } \sum_{j=1}^{L} w_{ij} \cdot \eta_j(x) \leq \sum_{j=1}^{L} w_{kj} \cdot \eta_j(x) \text{ minden } k \neq i\text{-re.}$$

Ha $w_{ij} = 1 - \delta_{ij}$, akkor a Bayes döntésfüggvény egyszerűbben is megadható:

$$\phi^*(x) = i, \text{ ha } \eta_i(x) \geq \eta_k(x) \text{ minden } k \neq i\text{-re.}$$

Az (X, Y) pár együttes eloszlásának ismeretében ϕ^* megszerkeszthető. Az alkalmazások többségében ezt nem ismerjük, viszont adott egy

$$(X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n), ...$$

független, (X, Y)-nal azonos eloszlású párokból álló sorozat, melyet felhasználhatunk a döntésfüggvény (rekurzív) megadásához. A

$$\mathcal{D}_n = \{(X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n)\}$$

halmazt *tananyagnak* nevezzük. X_i az i-edik *tanulópont*, Y_i a megfelelő *tanítás*. Gyakran használjuk a $\mathcal{T}_n = \{X_1, X_2, ..., X_n\}$ *tanulópont-halmaz* elnevezést is, ilyenkor az X_i tanulópont tanítására a $class(X_i)$ jelöléssel fogunk hivatkozni. A tanulóalgoritmusok a $\mathcal{D}_n$ tananyag függvényeként állítják elő a $g_n : \mathcal{X} \to \mathcal{Y}$ döntésfüggvényt. Ekkor értelmezhető a

$$\mathbf{P}(g_n(X) = i, Y = j \mid \mathcal{D}_n) = \mathbf{P}(g_n(X) = i, Y = j \mid \mathcal{G})$$

feltételes valószínűség, ahol a feltételben szereplő $\mathcal{G}$ a $\mathcal{D}_n$ tananyag által generált σ-algebra. A g_n döntésfüggvény feltételes rizikóján az

$$R_n = R(\underline{\underline{W}}, g_n) = \sum_{i=1}^{L} \sum_{j=1}^{L} w_{ij} \cdot \mathbf{P}(g_n(X) = i, Y = j \mid \mathcal{D}_n)$$

valószínűségi változót értjük.

Ha $\lim_{n \to \infty} \mathbf{E} R_n = R^*$, akkor a $\{g_n\}$ döntésfüggvény-sorozat, illetve a kapcsolatos tanulóalgoritmus *gyengén konzisztens.*

A legközelebbi társ módszer

Az alkalmazásokban az egyik legnépszerűbb tanulóalgoritmus a legközelebbi szomszéd, vagy legközelebbi társ (angolul Nearest Neighbor Decision Rule) módszer, amelynél a döntésfüggvény-sorozat definíciója:

$$g_n(x) = Y_j, \text{ ha } d(x, X_j) = \min_{\alpha=1,2,\ldots,n} d(x, X_\alpha).$$

Erre az algoritmusra általános feltételek mellett megmutatható, hogyha létezik $\lim_{n\to\infty} \mathbf{E}\left(R\left(\underline{\underline{W}}, g_n\right)\right) = R_{NN}$, akkor $R^* \leq R_{NN} \leq 2R^*$, vagyis $R^* = 0$ esetben a legközelebbi társ tanulóalgoritmus gyengén konzisztens lesz.

A legközelebbi társ módszereknek igen nagy irodalma van, itt most Devroye, Györfi, Lugosi [8], Cover és Hart [4], Stone [32], Duda és Hart [9], Devijever, P.A., Kittler, J. [6] valamint Fukunaga [16] összefoglaló művére hivatkozunk. Cover és Hart [4] foglalkoztak először a legközelebbi társ módszer aszimptotikus tulajdonságaival. Bebizonyították, hogyha az $(\mathcal{X}, d)$ metrikus térben az x osztályozandó pont 1 valószínűséggel az osztályok feltételes sűrűségfüggvényének folytonossági pontja, akkor az osztályozási hiba nagysága aszimptotikusan a Bayes hiba egyszerese és kétszerese közé fog esni függetlenül a d metrika megválasztásától. Ezt az eredményt többen általánosították arra az esetre is, amikor az X alakzat eloszlására semmilyen kikötés sincsen. (L. pl. Devroye [7].) Szeparábilis metrikus térben a legközelebbi szomszéd módszer konzisztens döntésfüggvények sorozatát fogja előállítani, ha a Bayes-hiba 0.

A dolgozatban kizárólag a legközelebbi társ módszerrel, illetve ennek módosításaival fogunk foglalkozni speciális szempontból. Legtöbbször $\mathcal{D}_n$-et (illetve $\mathcal{T}_n$-et) adottnak tekintjük, ami nem függ a véletlentől. Ezért ilyenkor elemeiket kisbetükkel fogjuk jelölni. Fent említettük, hogy a módszer aszimptotikusan - amikor a tananyag elemszáma végtelenhez tart - jó tulajdonságot mutat azzal, hogy rizikója a Bayes rizikó kétszeresénél kisebb. Gyakorlati alkalmazásoknál a mintaelemszám azonban véges. Minél nagyobb elemszámú tananyagra volna szükség a pontosság biztosításához, azonban ezáltal az osztályozás lassul, a költsége pedig növekszik. Hogyan lehet a $\mathcal{T}_n = \{t_1, t_2, ..., t_n\}$ tanulóponthalmazt (tananyagot) úgy átalakítani, hogy a pontosság ne romoljon, miközben a költségek csökkennek? Ennek a kérdésnek a lehetséges válaszait keressük és adjuk meg ebben a dolgozatban.

Tanulóalgoritmusoknál az osztályozás folyamata három lépésből áll.

1^o A tananyag átszerkesztése (előfeldolgozása).

2^o A döntésfüggvény előállítása.

3^o Ismeretlen alakzatvektorok osztályozása.

Az első két lépést csak egyszer kell végrehajtani, míg a harmadik lépésben az osztályozandó alakzatok száma igen nagy lehet. Pl. digitális műholdképek esetében az osztályozandó pixelek száma milliós nagyságrendű, de a rendszám felismerés, vagy a postai szortírozás során is több százezer osztályozandó objektum lehet naponta. Tehát a tananyag átszerkesztésének és a döntésfüggvény előállításának költségei -mivel ezeket csak egyszer kell elvégezni-, eltörpülnek az osztályozás költségeihez képest. Ezért érdemes a tananyagot előfeldolgozás alá vonni, hiszen az osztályozás során a futtatási idő-, számítási művelet- vagy tárolási kapacitás-csökkenés formájában jelentkező megtakarítás az előfeldolgozás költségeit bőségesen fedezni fogja.

Alább a $\mathcal{T}_n$ tanulópont-halmaz lehetséges előfeldolgozásait hat pontban foglaljuk össze. Az 1., 2. pontban vázolt előfeldolgozás típusok az osztályozás költségét hivatottak csökkenteni. A 3., 4. és az 5. pontban említett módszerek az osztályozási pontosság növelését célozzák. Az előfeldolgozások harmadik típusa -amit a 6. pontban részletezünk- a legközelebbi szomszéd gyorsított megtalálását szolgálja.

1. ***A $\mathcal{T}_n$ ritkítása.*** Olyan $\mathcal{T}^* \subset \mathcal{T}_n$ részhalmaz előállítása a cél, amely rendelkezik az alábbi két tulajdonsággal:
   a.) $\mathcal{T}^*$ pontjai pontosan osztályozzák a $\mathcal{T}_n \setminus \mathcal{T}^*$ tanulópontokat, azaz $\mathcal{T}^*$ *konzisztens.*
   b.) $\mathcal{T}^*$ minimális elemszámú konzisztens részhalmaza $\mathcal{T}_n$-nek.

2. ***A $\mathcal{T}_n$ tömörítése.*** A $\tilde{\mathcal{T}} = \{z_1, z_2, ..., z_k\} \subset \mathcal{X}$ a $\mathcal{T}_n$ prototípusa, ha tananyag, $k << n$ és pontosan osztályozza $\mathcal{T}_n$-t. Általában $\tilde{\mathcal{T}} \nsubseteq \mathcal{T}_n$, sőt $\tilde{\mathcal{T}} \cap \mathcal{T}_n = \emptyset$ is elképzelhető.

3. ***A $\mathcal{T}_n$ szűrése.*** Olyan $\mathcal{T}^* \subset \mathcal{T}_n$ részhalmaz előállítása a cél, amelynél $\mathcal{T}^*$ pontjait az legközelebbi szomszéd módszerrel pontosabban lehet osztályozni. A $\mathcal{T}_n \setminus \mathcal{T}^*$ halmaz elemei az adott metrikával osztályaik rosszul osztályozható (deviáns vagy outlier) tanulópontjai. $\mathcal{T}^*$-ot a fenti 1. pont szerint vagy 2. pont szerint további feldolgozásnak vetjük alá, $\mathcal{T}_n \setminus \mathcal{T}^*$ pontjait vagy külön kezeljük, vagy pedig mérési hibákból eredőeknek tekintjük, és elhagyjuk a további számolásból őket.

4. ***A $\mathcal{D}_n$ tananyag átdefiniálása.*** Ezen egy $f : \{1, 2, ..., L\} \to \{1, 2, ..., K\}$ transzformáció megadását értjük, amikoris $t \in \mathcal{T}_n$ esetén $class(t)$ helyett $f(class(t))$-t tekintjük a t tanulópont új tanításának. Ha a tanulópont eddig az i osztályhoz tartozott, ezután az $f(i)$ osztályhoz fog. Az új tananyaggal javul az osztályozás pontossága.

5. ***Metrikaválasztás.*** Olyan $f : \mathcal{X}^n \to \Delta$, $\Delta = \{d;\ d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ metrika$\}$ leképezés megadása a feladat, hogy az $f(\mathcal{T}_n)$ metrikával osztályozva $\mathcal{T}_n$-t a pontosság javuljon. Δ lehet véges sok metrikát tartalmazó halmaz, és lehet valamilyen súlyvektortól függő kontinuum számosságú halmaz is.

6. ***A legközelebbi társ gyorsított megkeresésének stratégiái.*** Itt az a cél, hogy n db szekvenciális metrikaszámítás helyett átlagosan sokkal kevesebb $d(x, t_i)$ távolság kiszámításával pontosan meg tudjuk állapítani az $x \in \mathcal{X}$ osztályozandó pont osztályát. (Még a minimális távolságot sem kell ehhez feltétlenül kiszámolni.) A keresési stratégia a $\mathcal{T}_n$ előfeldolgozásából nyert adatok, statisztikák segítségével adható meg.

A dolgozat első fejezetében a tananyag 1.-4. átszerkesztési lehetőségeit foglaljuk össze. A második fejezet a legközelebbi társ megkeresésének gyorsított algoritmusaival foglalkozik. A harmadik fejezet tárgya az alakzattér metrikájának megszerkesztése.

A disszertációban elért eredmények

A dolgozat eredményeinek egy része megtalálható a [38], [39], [40] publikációkban. Az alábbi listában tételesen is felsoroljuk az önálló téziseket.

1.1.7. Tomek második ritkítási eljárásának általánosítása, az 1.2. ellenpélda.
1.2.3. Prototípushalmaz szerkesztése segéd-ponthalmazzal.
1.3. A tananyag osztályainak átdefiniálása.
1.4. A tananyag szűrése.
2.1. A $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_4, \mathcal{K}_5$ kizárási feltételek, 2.1.-2.5. tételek.
2.3. A tananyag szeparálása, 2.1., 2.2. definíciók, 2.6.-2.8. tételek.
2.4. Algoritmusok szeparálható részhalmazok keresésére.
2.6. A kizárásos keresőalgoritmusok alaposssága.
2.8. A keresés hatékonyságának jellemzése, számítógéppel elvégzett kísérletek.
3.2. A skálázott metrikás keresés konvergencia-sebessége, 3.1. tétel, 3.3. segédtétel
3.3. A legjobban skálázott metrika kiválasztásának előkészítő fázisa.
3.5. A metrikák keverése.

1. fejezet

A tananyag szerkesztése

Bevezetés

A *tananyag szerkesztésén* egy $(\mathcal{X} \times \mathcal{Y})^n \to (\mathcal{X} \times \mathcal{Y})^m$ leképezés definiálását értjük, melynek az a célja, hogy az eredeti $\mathcal{D}_n$ tananyagból az osztályozáshoz jobb input $\mathcal{D}_m^*$ tananyagot állítsunk elő.

Az egyik ok az előfeldolgozásra az, hogy a kiindulási $\mathcal{D}_n$ tananyagban felléphet információ-redundancia. Ez alatt azt értjük, hogy egyes tanulópontok igazából nem vesznek részt az osztályozásban mert az általuk hordozott információt más tanulópontok is tartalmazzák. Az ilyen „felesleges" tanulópontokat célszerű ignorálni, hiszen csak a feldolgozás költségét növelik, a pontosságát nem. A redundancia megszüntetését szolgálják a *tananyag ritkítási eljárásai*, melyek feltárják és eltávolítják a felesleges tanulópontokat. Ezeket a módszereket az 1.1. pontban tárgyaljuk. Hasonló célt valósítanak meg a *tananyag-tömörítő* eljárások is. Ekkor a $\mathcal{D}_n$-nál kisebb számosságú, ún. *prototípus-halmazt* állítunk elő, amely jól helyettesíti majd $\mathcal{D}_n$-t az osztályozásnál. A tömörítéssel az 1.2. pontban foglalkozunk.

A tananyag egy másik jellemzője lehet, hogy elemei között hibás, vagy legalábbis kis valószínűséggel előforduló ún. *deviáns* tanulópont is vannak. Ezek pl. a legközelebbi szomszéd osztályozásnál ugyanolyan súllyal vesznek részt, mint „normális" társaik, és az osztályozás pontosságát rontják. Azokat az eljárásokat, melyek a deviáns tanulópontok detektálását, és a tananyagból eltávolítását végzik, *szűrésnek* nevezzük. A témakört az 1.3. pontban részletezzük.

Végül, egy harmadik jelenség is megfigyelhető az eredeti tananyagban, ami az átszerkesztést indokolhatja. Sokszor lehet tapasztalni, hogy az alakzatpontok nem mindig felelnek meg markánsan az osztályoknak. Egyrészt különböző osztályok az alakzattérben teljesen összekeveredhetnek, másrészt viszont lehet

olyan osztály is, amelynek pontjai jól szeparálható klaszterekbe tömörödve helyezkednek el. Ezeket az anomáliákat az *osztályok átdefiniálásával* lehet kezelni. A kapcsolódó módszereket az 1.4. pontban foglaljuk össze.

1.1. A tanulópont-halmaz ritkítása

A $\mathcal{T}_n$ tanulópont-halmaz ritkításán egy olyan $m < n$ elemszámú $\mathcal{T}^* \subset \mathcal{T}_n$ részhalmaz megadását értjük, mellyel $\mathcal{T}_n$ pontjai hiba nélkül, vagy minimális hiba mellett osztályozhatók a legközelebbi szomszéd módszerrel. A problémakörnek több lehetséges megoldása ismert. Ennek a fejezetnek az 1.1.1.-1.1.6. ponjaiban ismertetjük az irodalomban javasolt megoldásokat. Az 1.1.7. pontban egy új ritkító algoritmust adunk meg.

A leírásokban a $\mathcal{T}_n$ tanulópont-halmaz osztályait C_i jelöli:

$$C_i = \{t \in \mathcal{T}_n, class(t) = i\}, i = 1, 2, ..., L. \; \mathcal{T}_n = \bigcup_{i=1}^{L} C_i, C_i \cap C_j = \emptyset, i \neq j.$$

1.1.1. A kondenzált NN-módszer (CNN)

HART [19] az alábbi megoldást adta meg.

1^o Véletlenszerűen kiválasztjuk $\mathcal{T}_n$ egy tanulópontját, és $\mathcal{T}^*$-ba töltjük.

2^o Osztályozzuk a $\mathcal{T}_n \setminus \mathcal{T}^*$ tanulópontokat sorra. Legyen $t \in \mathcal{T}_n \setminus \mathcal{T}^*$ a következő pont. Két eset lehetséges.

- $t \in C_i \cap (\mathcal{T}_n \setminus \mathcal{T}^*)$ olyan, hogy $\exists t^* \in C_i \cap \mathcal{T}^*$, melyre $\min_{z \in \mathcal{T}^*} d(t, z) = d(t, t^*)$. Ilyenkor vesszük a következő tanulópontot $\mathcal{T}_n \setminus \mathcal{T}^*$-ból.
- $t \in C_i \cap (\mathcal{T}_n \setminus \mathcal{T}^*)$ olyan, hogy $\min_{z \in \mathcal{T}^*} d(t, z) = d(t, t^*)$ és $t^* \in \mathcal{T}^*$, de $t^* \notin C_i$, azaz t-t $\mathcal{T}^*$ rosszul osztályozza. Ilyenkor $\mathcal{T}^* := \mathcal{T}^* \cup \{t\}$, vagyis t-vel bővítjük $\mathcal{T}^*$-ot.

3^o Ha egyszer már $\mathcal{T}_n$ mindegyik tanulópontját megvizsgáltuk, ismételten végigmegyünk a $\mathcal{T}_n \setminus \mathcal{T}^*$ pontokon. Ha valamelyiket $\mathcal{T}^*$ rosszul osztályozza, azzal kibővítjük $\mathcal{T}^*$-ot. A 3^o lépés kétféleképpen fejeződhet be. Vagy mindegyik $\mathcal{T}_n \setminus \mathcal{T}^*$ pont belekerül $\mathcal{T}^*$-ba (azaz $\mathcal{T}_n \equiv \mathcal{T}^*$), vagy nem volt rosszul osztályozott tanulópont $\mathcal{T}_n \setminus \mathcal{T}^*$-ban. Ilyenkor megállunk, és a továbbiakban $\mathcal{T}^*$-gal osztályozunk.

1.1.2. A redukált NN-módszer (RNN)

G.W.Gates [18] a CNN módszerrel ritkított $\mathcal{T}^*$ tananyagot további feldolgozásnak veti alá kiküszöbölendő azt a hibát, hogy $\mathcal{T}^*$ tartalma függhet a feldolgozás sorrendjétől.

1^o Kiindulunk a CNN algoritmussal kapott $\mathcal{T}^*$-ból és betöltjük azt $\mathcal{T}^{**}$-ba.

2^o Egyenként sorravesszük $\mathcal{T}^{**}$ elemeit. Vizsgáljuk most $t \in \mathcal{T}^{**}$-ot. Két eset lehetséges:

- $\mathcal{T}^{**} \setminus \{t\}$-vel osztályozva $\mathcal{T}_n$ pontjait, hibátlan osztályozást kapunk. Ilyenkor t-t végleg eltávolítjuk $\mathcal{T}^{**}$-ból: $\mathcal{T}^{**} := \mathcal{T}^{**} \setminus \{t\}$.
- $\mathcal{T}^{**} \setminus \{t\}$-vel osztályozva $\mathcal{T}_n$ pontjait, kapunk hibás osztályozást $\mathcal{T}_n$-en. Ilyenkor t-t nem távolítjuk el $\mathcal{T}^{**}$-ból.

1.1.3. A δ -zónás CNN módszer

J.R.Ullman [34] -tól származik ez a ritkítási algoritmus, amely $\delta = 0$ esetben éppen a CNN módszer. „Jó” $\delta > 0$ paraméter választása esetén kisebb elemszámú ritkított elemszámú tananyag állítható elő a módszerrel. A δ értékének belövéséhez célszerű többször elindítani az algoritmust.

0^o Induláskor mindegyik C_i^* halmaz üres.

1^o $\forall i = 1, 2, ..., L$ -re legyen $t \in C_i$ tetszőleges: $C_i^* := C_i^* \cup \{t\}, \mathcal{T}^* := \bigcup_{i=1}^{L} C_i^*, \delta > 0$.

2^o Véletlenszerűen sorra véve a $t \in \mathcal{T}_n$ tanulópontokat, ha $t \in C_r \setminus C_r^*$ olyan, hogy $\nexists z \in C_r^*$ amivel

$$(*) \qquad d(t,z) + \delta < \min_{y \in T^* \setminus C_r^*} d(y,t)$$

lenne, akkor $C_r^* := C_r^* \cup \{t\}$ és $T^* := T^* \cup \{t\}$. Ellenkező esetben, tehát ha $\exists z \in C_r^*$ amivel (*) fennáll, nem bővítjük C_r^*, T^*-ot.

3^o A 2^o lépést addig ismételjük, amíg van olyan elem a $C_i \setminus C_i^*$ halmazokban, mellyel bővíteni lehetne $\mathcal{T}^*$-ot.

1.1.4. A δ -zónás RNN módszer

J.R.Ullman [34] egy másik ritkítási eljárást is leír, amely egyrészt $\delta = 0$ esetben az RNN módszer, másrészt az algoritmus igyekszik kiküszöbölni a CNN módszer azon további hibáját is, hogy jobb híján a C_r^*-ba töltjük azokat a $t \in C_r$ tanulópontokat, melyeket a $\mathcal{T}^*$ pontjaival nem tudunk pontosan osztályozni.

1^o Mindegyik $C_i \subset \mathcal{T}_n$ osztályhoz meg fogunk szerkeszteni egy $C_{i,0}^*, C_{i,1}^*, ..., C_{i,g}^* \subset C_i$ halmazsorozatot, ahol g az iterációk száma. Legyen $\delta > 0$.

2^o A $C_{i,0}^*$ elkészítése. A $t \in C_i$ tanulópontot akkor gyűjtjük bele $C_{i,0}^*$-ba, ha $t^*, t^{**} \in C_i \setminus \{t\}$ jelöli a t két legközelebbi szomszédját az osztályban, és $d(t, t^*) < d(t, t^{**})$, amikkel

$$(**) \qquad d(t, t^*) + \delta < \min_{z \in \mathcal{T}_n \setminus C_i} d(t, z), \text{ de } d(t, t^{**}) + \delta \geq \min_{z \in \mathcal{T}_n \setminus C_i} d(t, z).$$

3^o A $C_{i,0}^*$ halmazok segítségével készítjük el a $C_{i,1}^*$ halmazokat. A $t \in C_i$ eleme lesz $C_{i,1}^*$-nek, ha

$$(***) \qquad d(t, t^*) + \delta < \min_{z \in \mathcal{T}^* \setminus C_{i,0}^*} d(t, z), \text{ de } d(t, t^{**}) + \delta \geq \min_{z \in \mathcal{T}^* \setminus C_{i,0}^*} d(t, z).$$

4^o A 3^o lépést g iteráción keresztül végezzük.
A ritkított tananyag $\mathcal{T}^* = \bigcup_{i=1}^{L} C_{i,g}^*$ lesz.

1.1.5. A rendezett CNN módszer (OCNN)

A CNN módszer végeredményeként az iterációval kapott ritkított tananyag pontjai a két osztályt elválasztó sáv tanulópontjai lesznek. I. Tomek [33] módosításának az a lényege, hogy a ritkított tananyagba eleve az osztályokat elválasztó határsávból választ tanulópontokat, így a Hart-féle CNN algoritmus kevesebb iterációt igényel.

A CNN leírásában a 2^o pont második részét kell módosítani.

Legyen $t \in C_i \cap (\mathcal{T}_n \setminus \mathcal{T}^*)$ olyan, hogy $\min_{z \in \mathcal{T}^*} d(t, z) = d(t, t^*)$ és $t^* \in \mathcal{T}^*$, de $t^* \notin C_i$, azaz t-t $\mathcal{T}^*$ rosszul osztályozza. Ilyenkor megkeressük azt az $u \in \mathcal{T}_n \setminus \mathcal{T}^*$ pontot, ami egy osztályban van t-vel, és $d(u, t^*) < d(t, t^*)$. Ha van ilyen, ezzel az u tanulóponttal bővítjük $\mathcal{T}^*$-ot t helyett: $\mathcal{T}^* := \mathcal{T}^* \cup \{u\}$.

1.1.6. A szelektív NN módszer (SNN)

G.L.RITTER ET.AL. [28] eredménye a legélesebb ebben a témakörben, amennyiben a javasolt algoritmus minimális konzisztens tananyagot állít elő. Nevezetesen az SNN algoritmus olyan tanulópont halmazt konstruál, amely teljesíti az alábbiakat:

- $\mathcal{T}^*$ konzisztens részhalmaz, azaz $\forall t \in C_i \setminus \mathcal{T}^*$ tanulópontra teljesül, hogy $\exists t^* \in \mathcal{T}^* : d(t,t^*) < \min\limits_{z \notin C_i} d(t,z)$.
- $\mathcal{T}^*$ a fenti tulajdonságú halmazok között minimális elemszámú.

Legyen $t_\alpha \in C_i$, akkor $Y_\alpha = \left\{ z; z \in C_i, d(z,t_\alpha) < \min\limits_{y \notin C_i} d(t_\alpha, y) \right\}$. Legyen $a_{ij}^{(0)} = I_{\{t_j \in Y_i\}}$. Nyilván $a_{ii}^{(0)} = 1$ mindig fennáll. Az $a_{ij}^{(0)} = 0$, ha $class(t_i) \neq class(t_j)$, vagy bár $class(t_i) = class(t_j)$, de létezik olyan $z, class(z) \neq class(t_i)$, hogy $d(t_i,z) < d(t_i,t_j)$. Ekkor $\underline{\underline{A}}^{(0)} = \left(a_{ij}^{(0)}\right)_{i,j=1,\ldots,n}$ az algoritmus kiindulási mátrixa. Az $\underline{\underline{A}}^{(0)}$ mátrixot minden lépésben oly módon módosítjuk, hogy sorokat és oszlopokat törlünk belőle. A végső állapot az lesz, amikor mindegyik sort és oszlopot töröltük már. A k-adik lépés után redukált mátrixot $\underline{\underline{A}}^{(k)}$ fogja jelölni. Az $\underline{\underline{A}}^{(k)}$-ban a még nem törölt sorok sorszáma azon tanulópontok indexeinek felelnek meg, amelyeket még be lehet majd vonni a $\mathcal{T}^*$ halmazba, míg az oszlopok sorszámai azon tanulópontok indexeit mutatják, amelyek egyelőre még nem jól osztályozódnak az aktuális $\underline{\underline{A}}^{(k)}$-hoz tartozó $\mathcal{T}^*$ tartalma segítségével. Tegyük fel, hogy már $k \geq 0$ lépést végrehajtottunk, megkapva $\underline{\underline{A}}^{(k)}$-t.

1^o - Töröljük $\underline{\underline{A}}^{(k)}$-ban azon i oszlopokat (és t_i-t besoroljuk $\mathcal{T}^*$-ba), ha az i-edik oszlopvektor egységvektor. Ha például az i-edik oszlopnak csak a j-edik komponense nem nulla akkor megvizsgáljuk, hogy a j-edik sorvektorban van-e még 1-es valahol. Ha például az α-adik oszlopban van, akkor az α-adik oszlopot is töröljük (de t_α-t nem soroljuk be $\mathcal{T}^*$-ba). A törlések nyomán kapjuk az $\underline{\underline{A}}^{(k+1)}$ mátrixot.

- Töröljük $\underline{\underline{A}}^{(k+1)}$-ban azokat a j sorokat is, amelyeknél van nagyobb vagy egyenlő másik sorvektor. Azaz, ha $\underline{s}_j^{(k+1)}$ jelöli a mátrix j-edik sorvektorát, akkor $\exists \alpha : \underline{s}_j^{(k+1)} \leq \underline{s}_\alpha^{(k+1)}$ kell a j-edik sor törléséhez. A redukált mátrix az $\underline{\underline{A}}^{(k+2)}$.
- Töröljük $\underline{\underline{A}}^{(k+2)}$-ben azokat a j oszlopokat is, melyeknél van kisebb. Azaz a j-edik oszlopot töröljük, ha $\exists \alpha : \underline{o}_j^{(k+2)} \geq \underline{o}_\alpha^{(k+2)}$. Az ilymódon redukált mátrixot jelölje $\underline{\underline{A}}^{(k+3)}$.

2^o Az algoritmus leáll, ha nincs már sor vagy oszlop a kapott redukált mátrixban. A keresett ritkított tananyag $\mathcal{T}^*$ tartalma. Ha még van sor vagy oszlopvektor, de az oszlopok között nincsen egységvektor, a 3^o lépésre térünk. Különben visszatérünk 1^o-re.

3^o Töröljük azt az i-edik sort, amelyben az egyesek száma minimális. Ilyenkor a megfelelő tanulópontot besoroljuk $\mathcal{T}^*$-ba is. Visszalépünk 1^o-re.

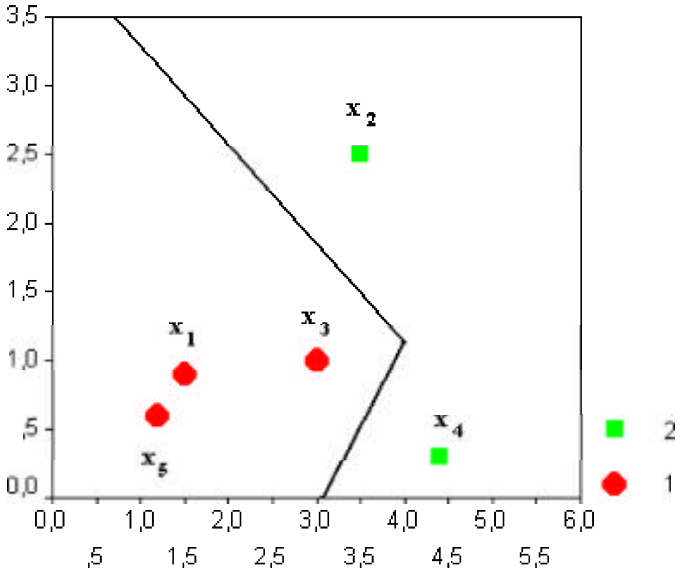
Az alábbi megoldást GÁBOR [17] publikálta. Az algoritmus ismertetése előtt szükségünk van néhány elnevezésre. A $\mathcal{D}_n$ tananyaghoz tartózó *rendezési táblázaton* azt az $n \times n$-es táblázatot értjük, melynek i-edik sorában az $(i, class\,(t_i))\,, (p_1, class\,(t_{p_1}))\,, (p_2, class(t_{p_2})\,, ..., \left(p_{n-1}, class\,\left(t_{p_{n-1}}\right)\right)$ elemek találhatók, ahol fennállnak a $d\,(t_i, t_{p_1}) \leq d\,(t_i, t_{p_2}) \leq \cdots \leq d\left(t_i, t_{p_{n-1}}\right)$ relációk $(i = 1, 2, ..., n)$.

A tananyaghoz tartozó rendezési tábla felhasználható az alábbi n egyenletből álló, n ismeretlenes logikai egyenletrendszer felállításához. Az egyenletrendszer ismeretlenjeit α_i-vel $(i = 1, 2, ..., n)$ fogjuk jelölni. $\rceil$ a negáció, $\vee$ a diszjunkció, $\wedge$ a konjukció, $=$ pedig az ekvivalencia jele lesz a leírásban. A rendezési tábla i-edik sora egyértelműen definiálja az egyenletrendszer i-edik egyenletét. Tegyük fel, hogy a rendezési tábla i-edik sorában csak a $j_1, j_2, ..., j_k$-adik oszlopában állnak a t_i tanulóponttal egy osztályba tartozó tanulópontok. (Tehát $class(t_i) = class(t_{p_{j_1}}) = \cdots = class(t_{p_{j_k}})$.) Akkor az i-edik logikai egyenlet: $\rceil\alpha_i = \left(\rceil\alpha_2 \wedge \rceil\alpha_3 \wedge \cdots \wedge \rceil\alpha_{p_{j_1-1}}\right) \wedge \alpha_{p_{j_1}} \vee \left(\rceil\alpha_{p_{j_1+1}} \wedge \rceil\alpha_{p_{j_1+2}} \wedge \cdots \wedge \rceil\alpha_{p_{j_2-1}}\right) \wedge \alpha_{p_{j_2}} \vee \cdots \vee \left(\rceil\alpha_{p_{j_{k-1}+1}} \wedge \rceil\alpha_{p_{j_{k-1}+2}} \wedge \cdots \wedge \rceil\alpha_{p_{j_k-1}}\right) \wedge \alpha_{p_{j_k}}$. Ha egy $\underline{\alpha}$ bináris vektor kielégíti a fenti egyenletrendszert, akkor $\underline{\alpha}$-hoz hozzákapcsolható egy relatíve minimális konzisztens halmaz: $C_{\underline{\alpha}} = \{t_i,\ \text{ha}\ \alpha_i = 1\ \ (i = 1, ..., n)\}$. ($C_{\underline{\alpha}}$ bármely elemét elhagyva nem kapnánk konzisztens halmazt!) Az abszolút minimális konzisztens halmaz ezek közül a minimális számosságú lesz.

1.1. *példa.* Az alábbi példát az egyenletrendszer felállításának jobb megértése végett GÁBOR [17] publikációjából vettük át. Az 1.1/a. ábra mutatja a tanulópontok síkbeli elhelyezkedését, az 1.1/b. mutatja a megfelelő rendezési táblát, 1.1/c. tartalmazza az egyenletrendszert, 1.1/d. pedig az összes lehetséges relatíve minimális konzisztens részhalmazt.

(1,1)	(5,1)	(3,1)	**(4,2)**	**(2,2)**
(2,2)	**(3,1)**	**(1,1)**	(4,2)	**(5,1)**
(3,1)	(1,1)	**(4,2)**	(5,1)	**(2,2)**
(4,2)	**(3,1)**	(2,2)	**(1,1)**	**(5,1)**
(5,1)	(1,1)	(3,1)	**(4,2)**	**(2,2)**

1.1/b. ábra. A rendezési táblázat



1.1./a. ábra. A feldolgozott tanulópontok elhelyezkedése a síkon. Az öt tanulópont két osztályhoz tartozik. Az 1-es osztályt kör, a 2-es osztályt négyzet jelöli.

$$\begin{aligned}
&\rceil\alpha_1 = \alpha_5 \vee \alpha_3 \\
&\rceil\alpha_2 = (\rceil\alpha_3 \wedge \rceil\alpha_1) \wedge \alpha_4 \\
&\rceil\alpha_3 = \alpha_1 \vee [(\rceil\alpha_4) \wedge \alpha_5] \\
&\rceil\alpha_4 = (\rceil\alpha_3) \wedge \alpha_2 \\
&\rceil\alpha_5 = \alpha_1 \vee \alpha_3
\end{aligned}$$

1.1/c. ábra. A logikai egyenletrendszer. Megoldásai $(0,1,0,0,1)\,,(0,1,1,1,0)$ és $(1,1,0,0,0)$

$$S_1 = \{2,5\}\,, S_2 = \{2,3,4\}\,, S_3 = \{1,2\}$$

1.1.d. ábra Az eredményül kapott relatíve minimális konzisztens halmazok

Az alábbiakban adjuk meg a módszerhez tartozó algorimust.

1^o $i := 1, \Lambda := \{\underline{\alpha} = (\alpha_1, ..., \alpha_n)\,; \alpha_j \in \{0,1\}\,, j = 1,2,...,n\}\,.$

2^o Elkészítjük a rendezési táblázat i-edik sorát.

3^o Felállítjuk a logikai egyenletrendszer i-edik egyenletét.

4^o Ha Λ-nak nincsen eleme, 7^o-re ugrunk. Különben végigmegyünk Λ összes elemén. Ha valamely $\underline{\alpha} \in \Lambda$ esetén nem elégül ki az i-edik egyenlet, $\underline{\alpha}$-t töröljük: $\Lambda := \Lambda \setminus \{\underline{\alpha}\}$.

5^o $i := i + 1$. Ha $i \leq n$, visszaugrunk 2^o-re, különben 6^o-ra térünk.

6^o Λ tartalma alapján előállítjuk az összes relatíve konzisztens halmazt. STOP.

7^o $\mathcal{D}_n$ nem ritkítható! STOP.

ZHANG és SUN [37] a tananyag prototípushalmazának megkonstruálását „tabu-tiltással" oldották meg. Meghatározandó az az $S^* \subseteq \mathcal{T}_n$ részhalmaz, mellyel osztályozva a $\mathcal{T}_n \setminus S^*$ pontokat az osztályozási hiba

$$L(S^*) = \frac{1}{n} \sum_{t \in \mathcal{T}_n} I_{\{class(t) \neq class(NN(t, \mathcal{T}_n \setminus \{t\}))\}}$$

relatív gyakorisága egy előírt $\varepsilon_0 \geq 0$ határt nem halad meg, és az ilyen tulajdonságú S részhalmazok között S^* minimális elemszámú. Amikor $\varepsilon_0 = 0$, akkor az algoritmus minimális elemszámú konzisztens részhalmazt fog előállítani. Az algoritmus egy iterációs folyamatban, hol egy újabb elemmel bővítve, hol egy elemet elhagyva módosítja az S tanulópont-halmaz előző tartalmát, amíg S^*-ot meg nem találjuk. A lépések során elkészül egy tabu-lista, melybe azok a halmazok kerülnek feljegyzésre, amelyek nem lehetnek folytatásai a fenti keresési folyamatnak. A tabu listában egy S halmazt egy n-dimenziós bináris vektorral azonosítunk: $\delta_i = I_{\{t_i \in S\}}, \underline{\delta}_S = (\delta_1, ..., \delta_n)^T$. A javasolt algoritmus leírása:

1^o Beállítjuk a TL FIFO tabu-lista maximális hosszát k-ra. Ezzel szabályozzuk, hogy maximum mennyi halmaz lehet egyszerre tiltás alatt. Mindig a legrégebben eltárolt halmazt írjuk felül egy újabb tiltásnál. Beállítjuk ε_0 értékét.

2^o Véletlenszerűen kiválasztunk egy $t \in \mathcal{T}_n$ tanulópontot, és $S_{curr} := \{t\}, S_{best} := \mathcal{T}_n, TL := \varnothing, ind := 0$.

3^o Kiszámoljuk az $L(S_{curr})$ értéket.

- Ha $L(S_{curr}) > \varepsilon_0$, akkor minden $u \in \mathcal{T}_n \setminus S_{curr}$ esetén képezzük az $S_{next} := S_{curr} \cup \{u\}$ halmazokat. Ezek közül csak azokat tekintjük, amelyek minimalizálják az $L(S_{next})$ kifejezést. Jelöljük ezt a minimumot L^*-gal!

$3^o/a$– Ha $L^* < L(S_{curr})$, akkor a minimumot produkáló S_{next} halmazok közül azt választjuk ki, amelyik minimalizálja a $D(S_{next}, \mathcal{T}_n) = \sum_{t \in \mathcal{T}_n \setminus S_{next}} d(t, t_{NN}^{=})$ távolságösszeget, ahol $t_{NN}^{=}$ a t-vel azonos osztályba eső legközelebbi társat jelöli S_{next}-ből.

– Ha $L^* \geq L(S_{curr})$, akkor a minimumot produkáló S_{next} halmazok közül csak azokat választjuk ki, amelyek legalább egy olyan $t \in \mathcal{T}_n \setminus S_{next}$ pontot jól osztályoznak, amit S_{curr} még nem osztályozott jól. Az ilyenek

közül kiválasztjuk a $D(S_{next}, \mathcal{T}_n)$ távolságösszeget minimalizálót. A 4^o pontra ugrunk.

Ha viszont nincs olyan halmaz az S_{next} halmazok között, melynél az eddig rosszul osztályozott pontok közül legalább egyet jól osztályozna, azt amelyik minimalizálja $D(S_{next}, \mathcal{T}_n)$-et, felvesszük a TL-listába: $TL := TL \cup \{\underline{\delta}_{S_{next}}\}$. Az algoritmust ilyenkor egy olyan S_{next} halmazzal folytatjuk, amely nem minimalizálta az $D(S_{next}, \mathcal{T}_n)$-et, és visszatérünk a $3^o/a$ pontra.

- Ha $L(S_{curr}) \leq \varepsilon_0$, akkor egy pontot elhagyunk S_{curr} elemei közül, azaz minden $u \in S_{curr}$ esetén képezzük az $S_{next} := S_{curr} \setminus \{u\}$ halmazokat. Az ilyenek közül azt fogjuk kiválasztani, amely minimalizálja $L(S_{next})$-et, és a minimumot adók közül azt vesszük, ahol $D(S_{next}, \mathcal{T}_n)$ is minimális.

4^o $S_{curr} := S_{next}$. Ha $\#S_{curr} < \#S_{best}$ és $L(S_{curr}) \leq \varepsilon_o$ vagy $L(S_{curr}) < L(S_{best})$ és $\#S_{curr} = \#S_{best}$, akkor $S_{best} := S_{curr}$. S_{curr}-t eltároljuk a TL listában, $ind := ind + 1$. Ha $ind < k$, visszatérünk a 3^o lépésre, különben megállunk.

Az 1.1.1.-1.1.6. pontoknál tárgyalt algoritmusokban közös, hogy a $\mathcal{T}^*$ ritkított tanulópont-halmaz elemszáma csak az eljárás végére alakul ki. Az alábbi tételben a ritkított tanulópont-halmaz elemszáma (m) előre rögzített érték.

Legyen $m < n$ rögzített. Jelölje C_m^n az $\{1, 2, ..., n\}$ számok összes m-edosztályú kombinációinak halmazát! Legyen $c = (i_1, i_2, ..., i_m) \in C_m^n$ egy tetszőleges kombináció, amihez tartozik egy

$$\mathcal{D}_c = \{(x_{i_1}, y_{i_1}), (x_{i_2}, y_{i_2}), ..., (x_{i_m}, y_{i_m})\} (\subset \mathcal{D}_n)$$

ritkított tananyag. Jelölje $\mathcal{E}_{n-m} = \mathcal{D}_n \setminus \mathcal{D}_c$ a tananyag maradékát! Jelölje g_n a $\mathcal{D}_c$ tananyagot felhasználó legközelebbi szomszéd döntésfüggvényt!

$$\hat{L}_{n,m}(\mathcal{D}_c, \mathcal{E}_{n-m}) = \frac{1}{n-m} \sum_{(x,y) \in \mathcal{E}_{n-m}} I_{\{g_n(x) \neq y\}},$$

a g_n-osztályozás empirikus hibája az $\mathcal{E}_{n-m}$ tesztanyagon. Tekintsük most azt a $c^* \in C_m^n$ kombinációt, és a hozzátartozó $\mathcal{D}_{c^*}$ ritkított tananyagot, amely minimalizálja $\hat{L}_{n,m}$-et! A $\mathcal{D}_{c^*}$-hez tartozó döntésfüggvényt g_n^* jelöli. Legyen $L_n^* = \mathbf{P}(g_n^*(X) \neq Y \mid \mathcal{D}_n)$ az osztályozás elvi hibája. Akkor belátható (DEVROYE, GYÖRFI, LUGOSI [8], 19.3 tétel), hogyha $m = o(n/\log m)$ és $m \to \infty$, akkor $\lim_{n \to \infty} \mathbf{E} L_n^* = L^*$, vagyis a ritkítással kapott g_n^* döntésfüggvény-sorozat konzisztens.

1.1.7. Ritkítás idegen pontpárokkal

TOMEK [33] cikkében egy második ritkítási módszert is javasol lineáris metrikus térben és $L = 2$ esetre. Az elv az eddig ismertetendő módszerektől alap-

vetően eltér, amennyiben megpróbál közvetlen feltételt adni a konzisztens halmaz pontjaira. Ezért ez az algorimus igazából nem ritkít, hanem kigyűjt. A módszer hibája az, hogy -*Tomek* állításával ellentétben- nem állít elő konzisztens halmazt, hanem annak csak egy C részhalmazát. *Tomek* cikkében ugyan kimond egy tételt a C halmaz konzisztenciájáról, de a bizonyítás hibás. Az 1.2. példában egy síkbeli ellenpéldát mutatunk be a fentiek igazolására. Az algoritmus mégis használhatónak tűnik az alábbi értelemben. Mivel a CNN módszer hatékonysága függ a feldolgozás sorrendjétől, célszerű a *Tomek* kigyűjtő algoritmusával előállított C halmaz pontjaival kezdeni a CNN algoritmust. így kevesebb iterációval lehet a konzisztens halmazhoz jutni. Megadjuk a *Tomek* módszernek az általánosítását tetszőleges $(\mathcal{X}, d)$ metrikus térben $L \geq 2$ esetben. A $\mathcal{T}_n$ tanulópont-halmaz egy C részhalmazát fogjuk előállítani ún. *idegen pontpárt* alkotó tanulópontokkal.

Az $x, y \in \mathcal{T}_n$ *idegen pontpárt* alkotnak, ha $class(x) \neq class(y)$, és minden $z \in \mathcal{T}_n \setminus \{x, y\}$-ra fennáll

$$d^2(z, x) + d^2(z, y) > d^2(x, y).$$

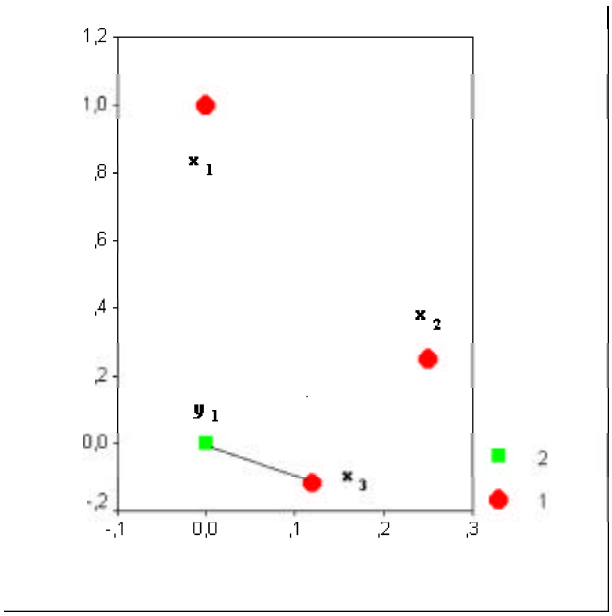
Az alábbi algoritmus előállítja az összes idegen pontpárból álló $C \subset \mathcal{T}_n$ részhalmazt.

Minden $x \in \mathcal{T}_n$ tanulópont esetén megvizsgáljuk, hogy van-e olyan $y \in \mathcal{T}_n, class(y) > class(x)$ tanulópont, hogy minden $z \in \mathcal{T}_n \setminus \{x, y\}$ esetén teljesül $d^2(z, x) + d^2(z, y) > d^2(x, y)$. Ha van ilyen x, y pár, tároljuk őket a C halmazban.

1.2. Példa: Az alábbi ellenpélda mutatja, hogy az idegen pontpárok halmaza nem mindig osztályozza korrektül a tananyagot. Álljon a tananyag az $\underline{x}_1 = (0, 1)$,$\underline{x}_2 = (0.25, 0.25)$, $\underline{x}_3 = (0.12, -0.12)$ és $\underline{y}_1 = (0, 0)$ pontokból, ahol $\underline{x}_1$,$\underline{x}_2$,$\underline{x}_3$ az 1 osztályhoz, $\underline{y}_1$ pedig a 2 osztályhoz tartozik. A pontok közül csak $\left(\underline{y}_1, \underline{x}_3\right)$ alkot idegen pontpárt, tehát $C = \left\{\underline{y}_1, \underline{x}_3\right\}$. Ez a halmaz viszont $\underline{x}_1$-et nem jól osztályozza. (Ld. a 1.2. ábrát.)

1.2. A tananyag tömörítése

Legyen $\mathcal{X}$ lineáris tér, $(\mathcal{X}, d)$ metrikus tér, $\mathcal{D}_n \subset \mathcal{X} \times \mathcal{Y}$ tananyag. Feladat olyan $\mathcal{P}_m \subset \mathcal{X} \times \mathcal{Y}$ *prototípus halmaz* megadása, hogy $m << n$ legyen és a $\mathcal{P}_m$-mel pontosan lehessen osztályozni $\mathcal{D}_n$ elemeit.



1.2. ábra. Az 1.2. példában szereplő pontok elhelyezkedése a síkon. Egyenes szakasszal kötöttük össze az egyetlen idegen pontpárt

1.2.1. Prototípushalmaz szerkesztése segítőkész tanítóval

Az $(\mathcal{X}, d)$ metrikus térben definiálható a *Voronoi-diagram* fogalma. Legyen $S \subset \mathcal{X}$ egy véges elemszámú ponthalmaz. Az S halmaz V_S Voronoi-diagramján azon $x \in \mathcal{X}$ pontok halmazát értjük, melyeknek van legalább két legközelebbi társa S-ben:

$$V_S = \left\{ x \in \mathcal{X} \; ; \exists a, b \in S, a \neq b : d(a, x) = d(b, x) = \min_{y \in S} d(y, x) \right\}.$$

Tehát, egy $\mathcal{T}_n$ tananyag egyértelműen definiálja a maga Voronoi-diagramját, melynek pontjai tulajdonképpen az $\mathcal{X}$ alakzattér határpontjainak halmazát jelentik. Ezek a határpontok a teret *Voronoi-cellákra* particionálják. Minden cella beazonosítható azzal a tanulóponttal, amely eleme a cellának (ő a cella *reprezentáns tanulópontja*), és azzal a tulajdonsággal rendelkezik, hogy a cella minden pontjához közelebb van, mint bármely más tanulópont. Az *NN*-algoritmusoknak az a célja, hogy minden $x \in \mathcal{X}$ pontról minél előbb eldönthető legyen, melyik Voronoi-cellához tartozik. A cellák közül azok, melyekneknél a „szomszédos" cellák reprezentáns tanulópontjai azonos osztályba tartoznak, feleslegesek. Az

ilyen cellába eső pontok akkor is jól osztályozhatóak lennének, ha a reprezentáló tanulópontot elhagyjuk a tananyagból.

Tehát egy $t \in \mathcal{T}_n$ *felesleges*, ha léteznek olyan $u_1, u_2, ..., u_k \in \mathcal{T}_n, class(u_i) = class(t)$ $(i = 1, ..., k)$ tanulópontok, hogy $C_t \subseteq C^*_{u_1} \cup C^*_{u_2} \cup \cdots \cup C^*_{u_k}$ teljesül, ahol C_t a t által reprezentált Voronoi-cella, ami a $\mathcal{T}_n$ halmaz Voronoi-diagramjához tartozik, $C^*_{u_i}$ pedig az u_i által reprezentált Voronoi-cellát jelöli, amely a $\mathcal{T}_n \setminus \{t\}$ halmaz Voronoi-diagramjának egy partíciója.

A tananyag-ritkító algoritmusok célja éppen a felesleges tanulópontok kiszűrése. Az $\mathcal{X} = \mathbb{R}^p$ speciális esetben érdekesen közelít a problémához Salzberg et al [29]. Abból indulnak ki, hogy ismerjük a Voronoi-diagram geometria szerkezetét. (Hasonlatos ez ahhoz a szituációhoz, amikor dolgozatírás közben egy segítőkész tanár megsúgja a végeredményt, és nekünk csak a megoldást kell kitalálnunk.) A dolgozatban azt vizsgálják, mennyi tanulópont szükséges minimálisan egy geometriailag jól körülírható térbeli alakzatot formázó Voronoi-diagram korrekt felismeréséhez. Ezzel bizonyos esetekben pontos alsó- és felsőkorlát adható a minimálisan szükséges (nem felesleges) tanulópontok számára.

1.2.2. Prototípus halmaz szerkesztése dinamikus klaszterezéssel

Az első algoritmus, amit ismertetünk Chin-Liang Chang [2] cikkén alapszik. Az algoritmus lényege az, hogy az egymáshoz közeli, azonos osztályhoz tartozó tanulópontokat összevonja és az átlagvektorukkal helyettesíti őket. Az ötletet nyilván a *McQueen* klaszterezési eljárás adta. Az alakzattér legyen most a p-dimenziós Euklideszi tér, azaz $\mathcal{X} = \mathbb{R}^p$!

0^o (*Inicializálás*) $\mathcal{P} := \mathcal{T}_n$,

$$w_i := \min_{\substack{u \in \mathcal{T}_n \setminus \{t_i\} \\ class(u) = class(t_i)}} d(t_i, u), \; b_i := \min_{\substack{u \in \mathcal{T}_n \\ class(u) \neq class(t_i)}} d(t_i, u), i = 1, 2, ..., n,$$

$A := \{t_1\}, B := \mathcal{T}_n \setminus \{t_1\}, count := 0, s(u) := 1, u \in \mathcal{P}.$

1^o Ha B-nek nincs eleme, az 5^o-re lépünk. Keressük meg azt a $p \in A, q \in B$ elempárt, mellyel $d(p, q) = \min_{u \in A, v \in B} d(u, v)$.

2^o

- Ha $class(p) = class(q)$, legyen $p^* := \frac{s(p)p + s(q)q}{s(p) + s(q)}$ és $s(p^*) := s(p) + s(q)$. Térjünk a 3^o lépésre.
- Ha $class(p) \neq class(q)$, akkor töröljük q-t B-ből, p-t és eltároljuk A-ban. Térjünk vissza az 1^o lépésre.

3^o (Módosítjuk $\underline{w}$ és $\underline{b}$ komponenseit)

- Ha $class(t_i) = class(p)$ és t_i-nek nem legközelebbi társa sem p sem q $A \cup B$-ben, akkor $w_i^* := \min\{w_i, d(t_i, p^*)\}$.

- Ha $class(t_i) = class(p)$ és t_i-nek p vagy q legközelebbi társa $A \cup B$-ben, akkor $w_i^* := \min\limits_{\substack{u \in A \cup B \setminus \{p,q,t_i\} \\ class(t_i)=class(u)}} d(t_i, u)$.

- Ha $class(t_i) \neq class(p)$ és t_i-nek nem legközelebbi társa sem p sem q $A \cup B$-ben, akkor $b_i^* := \min\{w_i, d(t_i, p^*)\}$.

- Ha $class(t_i) \neq class(p)$ és t_i-nek p vagy q legközelebbi társa $A \cup B$-ben, akkor $b_i^* := \min\limits_{\substack{u \in A \cup B \setminus \{p,q,t_i\} \\ class(t_i)\neq class(u)}} d(t_i, u)$.

4^o Megvizsgáljuk minden i-re, hogy ahol $w_i < b_i$ volt $w_i^* < b_i^*$ is fennáll-e.

- Ha igen, q-t töröljük B-ből, p-t töröljük A-ból és helyére p^*-ot tároljuk. Minden i-re legyen $w_i := w_i^*$ és $b_i := b_i^*$. $count := count + 1$. Visszatérünk az 1^o lépésre.

- Ha nem, azaz létezik j, hogy $w_j < b_j$, de $w_j^* \geq b_j^*$, akkor q-t töröljük B-ből és eltároljuk A-ban. Visszatérünk az 1^o lépésre.

5^o - Ha $count = 0$, azaz B úgy ürült ki, hogy egyetlen prototípus pontot sem módosítottunk, akkor az algoritmust leállítjuk. Ekkor az A halmaz tartalmazza a prototípus pontokat, azaz $\mathcal{P} := A$.

- Ha $count > 0$, akkor A tartalmát visszatöltjük B-be, B egy tetszőleges elemét átmozgatjuk A-ba, és egy újabb ciklust indítunk. $count := 0$ és visszatérünk az 1^o lépésre.

1.2.3. Prototípus halmaz szerkesztése segéd ponthalmazzal

Az ebben a pontban tárgyalt algoritmus általános metrikus térben alkalmazható, ha a $\mathcal{D}_n$ tananyagon kívül adott egy véges $\mathcal{A}_k \subset \mathcal{X}$ segéd ponthalmaz is, melynek elemei kezdetben nincsenek osztályozva, éppen az algoritmus fogja $\mathcal{A}_k$ elemeinek osztályait definiálni. Legyen tehát $\mathcal{D}_n$ egy tananyag, $\mathcal{T}_n \subset \mathcal{X}$ a hozzátartozó tanulópont-halmaz, $\mathcal{A}_k \subset \mathcal{X}$ pedig egy k elemszámú ponthalmaz. Cél olyan $\mathcal{P}_l$ prototípus halmaz szerkesztése a $\mathcal{T}_n \cup \mathcal{S}_k$ pontjaiból, ahol $l << n$ és $\hat{L}_n =$

$\frac{1}{n}\sum_{t\in\mathcal{T}_n} I_{\{\phi(t)\neq class(t)\}} = 0$. A képletben ϕ jelöli a $\mathcal{P}_l$ tananyaghoz tartozó NN-döntésfüggvényt.

Tekintsük az SNN-módszerrel sűrített $\mathcal{D}_m^*$ tananyagot és a megfelelő $\mathcal{T}_m^*$ tanulópont-halmazt! (L. 1.1.6. pontot!) Láttuk, hogy $\mathcal{D}_m^*$ minimális elemszámú olyan részhalmaz, mellyel hiba nélkül tudjuk az összes tanulópontot osztályozni. Legyen $y \in \mathcal{T}_m^*$ esetén

$$Group(y) = \left\{u; u \in \mathcal{T}_n \setminus \mathcal{T}_m^*, d(u,y) = \min_{w\in\mathcal{T}_m^*} d(u,w)\right\}$$

azon tanulópontok halmaza, melyet y ismer fel NN-osztályozáskor.

Célunk az, hogy az $\mathcal{A}_k$ halmaz elemeivel minél több pontot ki tudjunk váltani $\mathcal{T}_m^*$-ből. A tananyagba bevont $x \in \mathcal{A}_k$ pontok tanításait úgy fogjuk definiálni, hogy a fenti cél minél jobban megvalósulhasson. Jelölje $t_{NN}(x)$ azt a tanulópontot $\mathcal{T}_m^*$-ban, amely legközelebb van x-hez. Legyen

$$Sub(x) = \left\{y; y \in \mathcal{T}_m^*, d(y,x) < \min_{class(t_{NN}(x))\neq class(z)} d(z,x)\right\}$$

azon tanulópontok halmaza, melyeket potenciálisan ki lehet váltani $\mathcal{T}_m^*$-ből az x segítségével, ha belőle $class(t_{NN}(x))$ tanítású tanulópontot szerkesztünk.

Legyen most $y \in Sub(x)$ tetszőlegesen rögzített. Két eset lehetséges:

a. vagy minden $z \in Group(y)$ esetén $d(z,u) = \min_{w\in(\mathcal{T}_m^*\setminus\{y\})\cup\{x\}} d(z,w)$ esetén $class(u) = class(z)$;

b. vagy létezik $z \in Group(y)$, hogy $d(z,u) = \min_{w\in(\mathcal{T}_m^*\setminus\{y\})\cup\{x\}} d(z,w)$ esetén $class(u) \neq class(z)$.

Az **a.** esetben y *behelyettesíthető* x-szel. Ha az x-szel behelyettesíthető $y \in Sub(x)$ elemek száma egynél nagyobb ($cond(x) > 1$), akkor az x ponttal $\mathcal{T}_m^*$ *sűríthető*. Tehát $cond(x) = \sum_{y\in sub(x)} I_{\{y \text{ behelyettesíthető } x\text{-szel}\}}$.

A fenti jelöléseket és fogalmakat felhasználva az alábbi algoritmus készíthető el. (A leírásban $\#A$ az A halmaz elemeinek a számát jelöli.)

0^o $\mathcal{T}^{**} := \mathcal{T}_m^*, \mathcal{T}^{***} := \emptyset, \mathcal{A}^* := \mathcal{SA}_k$.

1^o Határozzuk meg minden $x \in \mathcal{A}^*$ esetében a $Sub(x) \subset \mathcal{T}^{**}$ halmazokat!

2^o Vegyük azt az $x \in \mathcal{A}^*$ elemet, ahol $\#Sub(x)$ maximális! Legyen $cond(x) := 0$!

3^o Minden $y \in Sub\,(x)$ esetén határozzuk meg a $Group(y)$ halmazokat!

4^o Vegyük egy olyan $y \in Sub\,(x)$ elemet, ahol $\#Group(y)$ minimális! (Az ilyen elemeket könnyű behelyettesíteni!)

5^o Ha minden $z \in Group(y)$ esetén $d\,(z,u) = \min\limits_{w \in (\mathcal{T}_m^* \setminus \{y\}) \cup \{x\}} d(z,w)$ esetén $class(u) = class(z)$, akkor $\mathcal{T}^{**} := \mathcal{T}^{**} \setminus \{y\}$, $sub(x) := sub(x) \setminus \{y\}$, $cond(x) := cond(x) + 1$. Ugorjunk vissza a 4^o pontra!

6^o Ha már nincs több elem $Sub(x)$-ben, akkor $cond(x) > 1$ esetén legyen $\mathcal{T}^{***} := \mathcal{T}^{***} \cup \{x\}$, és ugorjunk a 7^o-pontra!

7^o Legyen $\mathcal{A}^* := \mathcal{A}^* \setminus \{x\}$. Ha már $\mathcal{A}^* = \emptyset$, akkor $\mathcal{P}_l = \mathcal{T}^{**} \cup \mathcal{T}^{***}$ és STOP. Különben ugorjunk az 1^o-re!

1.1. megjegyzés. *Az algoritmusból látszik, hogy $l \leq m$ és $\hat{L}_n = 0$. Nem feltétlenül használunk fel minden elemet $\mathcal{A}_k$-ból a tömörítésre, sőt szerencsétlen esetben $\mathcal{P}_l \equiv \mathcal{T}_m^*$ is előfordulhat!*

1.2.4. Prototípus halmaz szerkesztése k-NN osztályozással

HATTORI és TAKAHASHI [20] a k-NN osztályozás segítségével javasolja a prototípushalmaz megszerkesztését. Algoritmusuk három lépésből áll.

1^o Először a $\mathcal{T}_n$ tanulópont halmazból kiválasztjuk az ún. „tipikusakat". Egy $t \in \mathcal{T}_n$ tanulópont tipikus, ha t legközelebbi k társa $\mathcal{T}_n \setminus \{t\}$-ből ugyanahhoz az osztályhoz tartozik, mint t. Jelölje $\mathcal{T}_n^*\,(k)$ a tipikus tanulópontok halmazát! Ez a halmaz függ a k megválasztásától. Határozzuk meg a tipikus tanulóponthalmazokat $k = 1, 2, ..., K$ esetén!

2^o Minden lehetséges k és $k' = 1, ..., k$ esetén határozzuk meg, hány tanulópontot lehet jól osztályozni a $\mathcal{T}_n \setminus \mathcal{T}_n^*\,(k)$ tanulópontok közül a $k' - NN$ osztályozást alkalmazva, amikor a tananyag $\mathcal{T}_n^*\,(k)$. Ehhez adjuk hozzá azon $t \in \mathcal{T}_n^*\,(k)$ tanulópontok számát, melyeket $k' - NN$ osztályozással helyes osztályba lehet sorolni, amikor a tanulópont-halmaz $\mathcal{T}_n^*\,(k) \setminus \{t\}$! Válasszuk ki azokat a (k, k') párokat, ahol a legnagyobb osztályozási pontosságot kaptuk. Jelölje Γ ezen optimális párok halmazát!

3^o Minden $(k, k') \in \Gamma$ és $t \in \mathcal{T}_n \setminus \mathcal{T}_n^*\,(k)$ tanulópont esetén megvizsgáljuk, hogy milyen osztályba sorolná t-t a $k' - NN$ osztályozás, amikor a tanulópont-halmaz $\mathcal{T}_n^*\,(k)$. Ha $t \in \mathcal{T}_n^*\,(k)$, akkor a tanulópont-halmaz $\mathcal{T}_n^*\,(k) \setminus \{t\}$ lesz.

Végül, a t tanulópontnak azt a címkét fogjuk adni, amit az osztályozások többségénél kapott. A szerkesztett prototípushalmaz tehát annyiban különbözik az erdeti tananyagtól, hogy a tanulópontok egy részénél megváltozik az eredeti osztálybasorolás.

1.2.5. Prototípus halmaz szerkesztése hibafüggvény minimalizálásával

Lineáris alakzattérben, súlyozott Euklideszi metrika mellett alkalmazható HUANG ET AL. [21] prototípusszerkesztő eljárása. A szerkesztendő prototípushalmaz egy hibafüggvény minimalizálása révén keletkezik. A prototípus-vektorok iterációk során, az általánosított δ-szabállyal módosulnak a hibafüggvény gradiense irányában.

Ha $\mathcal{X} = \mathbb{R}^p$ és $\mathcal{Y} = \{0, 1\}$, akkor az alábbi eredmény mutatja a prototípus halmazok alkalmazásának hatékonyságát.

Jelölje $\mathcal{C}$ az összes olyan legközelebbi társ osztályozás döntésfüggvények halmazát, ahol a tananyag végigfut az összes m elemszámú prototípus halmazon $\mathcal{X} \times \{0, 1\}$-ban. Legyen $\mathcal{D}_n$ egy tananyag. Jelölje g_n azt a döntésfüggvényt, amely minimalizálja az

$$\hat{L}_n(\phi) = \frac{1}{n} \sum_{(X,Y) \in \mathcal{D}_n} I_{\{\phi(X) \neq Y\}}$$

hibát. Ekkor tetszőleges $\varepsilon > 0$ esetén

$$\mathbf{P}\left(L(g_n) - \inf_{\phi \in \mathcal{C}} L(\phi) > \varepsilon \right) \leq 8 \cdot 2^n \cdot n^{\frac{(p+1)m(m-1)}{2}} \cdot e^{-\frac{n\varepsilon^2}{128}}.$$

Ha $m^2 \log m = o(n)$, akkor az így megszerkesztett prototípushalmazzal $m \to \infty$ esetben konzisztens osztályozás létesíthető. (L. DEVROYE, GYÖRFI, LUGOSI [8], 19.6 tétel).

1.3. A tananyag osztályainak átdefiniálása

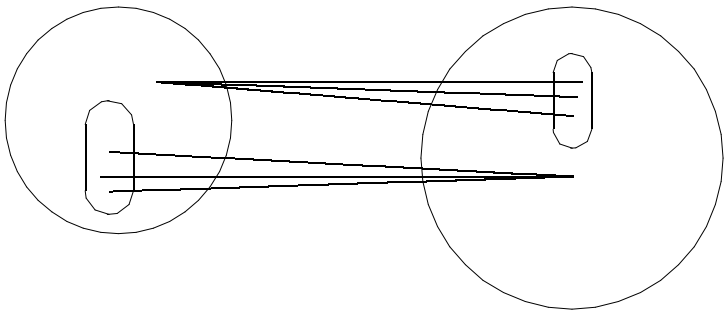
Adott a $\mathcal{T}_n = \bigcup_{i=1}^{L} C_i$ tanulópont-halmaz, ahol a tanulópontok az $(\mathcal{X}, d)$ metrikus tér pontjai. Tegyük fel, hogy $\mathcal{T}_n$-et itt most nem részletezett módon egy jó klaszterezési eljárással M különböző, diszjunkt klaszterbe soroltuk: $\mathcal{T}_n = \bigcup_{i=1}^{M} K_i$. (A $\mathcal{T}_n$ tanulópont-halmaz klaszterezési problémáiról a 1.3.3. és 1.3.4. pontokban

lesz szó.) A klaszterezést az C_i osztályok átdefiniálására fogjuk felhasználni. Elvileg ugyanis lehetséges, hogy bizonyos osztályok tanulópontjai a d metrika szerint túl közel vannak egymáshoz, így az osztályok szétválasztása irreális feladat lenne. Az ilyen osztályokat célszerű összevonni. Másrészt az is lehet, hogy valamely osztály tanulópontjai több markánsan elkülönülő klaszterba sorolódtak. Ilyenkor felmerül az a kérdés, nem célszerű-e ezt az osztályt több önálló osztályba szétbontani? Egyes alkalmazásoknál indokolt lehet ezen az alapon a tananyag tanításainak megváltoztatása.

1.2. megjegyzés. *Például egy többsávos digitális műholdkép pixeleinek osztályozásakor lehet találkozni az alább vázolt problémával. Térkép koordinátái alapján a műholdképen jól be lehet azonosítani a földi objektumokat. Ki lehet jelölni, hogy melyik folt képpontjai tartoznak településhez, vízfelülethez, búzatáblához vagy más ültetvényhez stb.. Így a képről leválasztott képpontokat osztályokba lehet sorolni, azaz tananyagot lehet szerkeszteni az osztályozáshoz. A tanulópontok a képpontoknak megfelelő vektorok, a tanítások pedig a földi objektumnak megfelelő osztályok azonosítói lesznek. A vegetációs időszak kezdetén készült felvételen azonban előfordul, hogy különböző kultúrnövények (búza, kukorica, napraforgó) alakzatvektorai azonos eloszlást követnek, így a megfelelő osztályok szétválaszthatatlanok lesznek. Ugyanakkor néhány hónap múlva, ugyanahhoz a felszínről készült felvételen azt lehet tapasztalni, hogy az egy osztályba tartozó képpontok halmazában markáns klaszterek vannak. A búza esetében pl. statisztikai jellemzők alapján szignifikáns különbségek alakulnak ki az egyes búzafajtáknál. Ilyenkor indokolt az egyes típusok szerint részekre darabolni a búzaosztályt.*

*Egy másik alkalmazási példa a betűfelismeréshez tartozik. Egy szöveget digitalizálva a betűket és egyéb jeleket alakzatvektorokra képezzük le. A feladat az, hogy a szöveg jeleit a megfelelő betűnek vagy írásjelnek feleltessük meg automatikusan. A megoldást nehezíti az, hogy az egyes betűk különböző kurzív típusokhoz tartoznak. Például egy Word fájl ,,a" betűje lehet Times New Roman, Ariel, Courier stb., és azon belül még lehet boldface, italic vagy underline formátumú. Vagyis a szöveg ,,a" betűinek alakzatvektorai jól elkülöníthető klaszterekbe csoportosulnak. Másrészt teljesen eltérő betűosztályok lehetnek könnyen összetéveszthetők. Az italic ,,v" betű osztálya nehezen különül el a görög ,,nű" betűtől: ν. (*Lee és Chae *[24] kézzel írott számjegyek felismerésekor pl. az egyes digitekhez 8-8 osztályt definiáltak.)*

A tananyag osztályainak átdefiniálása a jobb alkalmazhatóságot, az interpretálhatóság javítását szolgálja. Természetesen ilyenkor javul az osztályozás pontossága és a legközelebbi szomszéd keresésének hatékonysága is.



1.3. ábra Az átdefiniálás relációja. Azt zárjuk ki, hogy egy régi osztály egy részhalmaza egy tőle különböző másik régi osztály egy részhalmazával egyesítve alkothasson új osztályt.

1.1. definíció. *A $\mathcal{T}_n$ tananyag osztályainak átdefiniálásán egy $\Im \subset \{1,2,...,L\}\times\{1,2,...,M\}$ relációt értünk, ahol*
a.) $\forall i \in \{1,2,...,L\}$*-hez* $\exists j \in \{1,2,...,M\} : (i,j) \in \Im$;
b.) $\forall j \in \{1,2,...,M\}$*-hez* $\exists i \in \{1,2,...,L\} : (i,j) \in \Im$;
c.) Ha $i_1, i_2 \in \{1,2,...,L\}$*-hoz* $\exists j_1 \in \{1,2,...,M\} : (i_1,j_1),(i_2,j_1) \in \Im$, *akkor* $\forall j_2 \neq j_1$ *esetén* $(i_1,j_2),(i_2,j_2) \notin \Im$.
d.) Ha $j_1, j_2 \in \{1,2,...,M\}$*-hoz* $\exists i_1 \in \{1,2,...,L\} : (i_1,j_1),(i_1,j_2) \in \Im$, *akkor* $\forall i_2 \neq i_1$ *esetén* $(i_2,j_1),(i_2,j_2) \notin \Im$.

Az 1.3. ábrán szemléltetjük az átdefiniálás relációját.

1.2. definíció. *Az $\Im$ átdefiniálási reláció mátrix reprezentánsa az a $L \times M$-es rendű $\underline{\underline{\Phi}}$ mátrix, ahol $\varphi_{ij} = I_{\{(i,j)\in\Im\}}$.*

1.3. megjegyzés. $o_j = \sum_{i=1}^{L} \varphi_{ij} \geq 1$, *és* $s_i = \sum_{j=1}^{M} \varphi_{ij} \geq 1$. *Ha* $s_i > 1$ *akkor* $\forall\alpha$ *indexre ahol* $\varphi_{i\alpha} = 1$ *szükségképpen* $o_\alpha = 1$. *Ha viszont* $o_j > 1$ *akkor* $\forall\beta$ *indexre ahol* $\varphi_{\beta j} = 1$ *szükségképpen* $s_\beta = 1$. *(Vagyis* $s_i \cdot o_j > 1$ *esetén* $\varphi_{ij} = 0$ *kell, hogy teljesüljön). Másrészt annak is teljesednie kell, hogy* $\sum_{i=1}^{L} s_i = \sum_{j=1}^{M} o_j \geq \min\{L,M\}$.

Legyen most $\vartheta_{ij} = \sum_{t\in\mathcal{T}_n} I_{\{t\in C_i\cap K_j\}}$. Jelölje $\underline{\underline{V}} = (\vartheta_{ij})$ a gyakoriságok kontingencia táblázatát! Cél olyan $\Im$ átdefiniálás megadása, hogy $\sum_{(i,j)\in\Im} \vartheta_{ij} =$

$\sum_{i=1}^{L}\sum_{j=1}^{M}\varphi_{ij}\cdot\vartheta_{ij}$ maximális legyen! Ha valamely $\underline{\underline{\Phi}}$ mátrixnál eléretik a maximum, akkor az $o_j > 1$ esetén az $\bigcup_{\forall\beta:\varphi_{\beta j}=1} C_\beta$ osztályösszevonást célszerű végrehajtani, míg $s_i > 1$ esetén az i-edik osztályt célszerű szeparálni: $\forall\alpha$ indexhez, ahol $\varphi_{i\alpha}=1$ tartozik majd egy $C_{i\alpha}=C_i\cap K_\alpha$ új osztály.

1.4. megjegyzés. *A $\underline{\underline{V}}$ kontingencia táblázatot nem csak klaszterezéssel állíthatjuk elő, hanem a $\mathcal{T}_n$ tanulópont-halmaz pontjainak ún. törölt legközelebbi társ osztályozás módszere segítségével is. Jelölje ϕ_{dNN} a törölt legközelebbi társ döntésfüggvényét, azaz $\phi_{dNN}(x)=y_i$, ha $d(x,x_i)=\min_{t\in\mathcal{T}_n\setminus\{x\}} d(t,x)$, minden $x\in\mathcal{T}_n$-re. Számoljuk ki a $\vartheta_{ij}=\sum_{x\in\mathcal{T}_n} I_{\{class(x)=i,\phi_{dNN}(x)=j\}}$ gyakoriságokat, és ezekkel képezzük a $\underline{\underline{V}}=(\vartheta_{ij})$ kontingencia mátrixot! Most ez lesz az inputja a $\underline{\underline{\Phi}}$ hozzárendelési mátrixot előállító algoritmusnak. Ilyenkor az $L=M$ speciális esettel állunk szemben.*

1.3.1. Algoritmusok átdefiniálási reláció mátrix-reprezentációjának előállítására

Szelektáló algoritmus. *Inicializálás:* Legyen $T:=0$.

Ciklus: Elindítunk egy M-szeresen egybeágyazott ciklust.
$IND_1:=1,2,...,2^M; IND_2:=1,2,...,2^M; ...; IND_L:=1,2,...,2^M$.

1^o Előállítjuk a $\underline{\underline{\Phi}}$ mátrix elemeit az $IND_1, IND_2, ..., IND_L$ pillanatnyi értékének megfelelően. $\underline{\underline{\Phi}}$ i-edik sora IND_i-1 kettes-számrendszerbeli digitjeiből állnak. Ha $IND_i-1=\sum_{j=0}^{M-1}\varepsilon_j 2^j$, akkor $\varphi_{ij}=\varepsilon_j$.

2^o Kiszámoljuk az $\underline{s}$ és $\underline{o}$ sor- illetve oszlopösszeg-vektorokat:
$s_i=\sum_{j=1}^{M}\varphi_{ij}, o_j=\sum_{i=1}^{L}\varphi_{ij}$.

3^o Ha $\sum_{i=1}^{L} s_i=\sum_{j=1}^{M} o_j\geq\min\{L,M\}, \min s_i\geq 1, \min o_j\geq 1$,és $s_i\cdot o_j>1$ esetén $\varphi_{ij}=0$, akkor $\underline{\underline{\Phi}}$ hozzárendelési mátrix. Ekkor kiszámítjuk az $S=\sum_{i=1}^{L}\sum_{j=1}^{M}\varphi_{ij}\cdot\vartheta_{ij}$ összeget. Ha $S>T$, akkor $T:=S, \underline{\underline{\Phi}}_{\max}:=\underline{\underline{\Phi}}$. Innen a ciklus végére ugrunk.

4^o Ha az előző szakaszban leírt feltételrendszer nem teljesül, egyből a ciklus végére ugrunk.

5^o A ciklus lejártakor T tartalmazza a maximális hozzárendelési értéket, $\underline{\underline{\Phi}}_{\max}$ pedig az optimális hozzárendelési mátrixot.

Mohó algoritmus. Az alábbi algoritmus ugyan nem garantálja a maximumot szolgáltató $\underline{\underline{\Phi}}$ előállítását, de a maximumhoz közeli megoldást ad, jóval egyszerűbben megvalósítható algoritmussal.

1^o Induláskor a $\underline{\underline{\Phi}}, \underline{o}, \underline{s}$ komponensei mind 0-k, ahol $\underline{o}$ az oszlopok peremösszegeit, $\underline{s}$ a sorok peremösszegeit tartalmazó vektorok. Jelölje ϑ_{ij} egy pillanatra az $\underline{\underline{V}}$ mátrix maximális elemét. Legyen $\varphi_{ij} = o_j = s_i = 1$.

2^o Az $\underline{\underline{V}}$ második legnagyobb elemét ha ϑ_{kl} jelöli, akkor $\varphi_{kl} := 1$ legyen és $o_l := o_l + 1, s_k := s_k + 1$.

3^o Keressük meg azt a ϑ_{gh} elemet, amely maximális a $\varphi_{gh} = 0 \wedge \min\{o_h, s_g\} = 0 \wedge \left(\max_{i=1,\dots,L} \varphi_{ih} \cdot s_i \leq 1\right)$ feltételt kielégítő elemek között. Legyen ilyenkor $\varphi_{gh} := 1,\ o_h := o_h + 1,\ s_g := s_g + 1$.

4^o A 3^o lépést addig ismételjük, amíg lehetséges. Ha nincs a $\varphi_{gh} = 0 \wedge \min\{o_h, s_g\} = 0 \wedge \left(\max_{i=1,\dots,L} \varphi_{ih} \cdot s_i \leq 1\right)$ feltételt kielégítő elem már, akkor megállunk. Ezután lehet $\underline{\underline{\Phi}}$ szerint átdefiniálni az eredeti $C_1, C_2, .., C_L$ osztályokat.

A $\underline{\underline{\Phi}}$ mátrix alapján az alábbi módon végezhető el a tanulópontok átcímkézése.

Azokat a $C_i, C_j, ..., C_k$ osztályokat vonjuk egybe egyetlen C_l^* osztályba, ahol $\varphi_{il} = \varphi_{jl} = \cdots = \varphi_{kl} = 1$. Azon C_l osztály bomlik szét $C_i^*, C_j^*, ..., C_k^*$ osztályokba, ahol $\varphi_{li} = \varphi_{lj} = \cdots = \varphi_{lk} = 1$.
Nyilván $C_i^* \supseteq C_l \cap K_i, C_j^* \supseteq C_l \cap K_j, ..., C_k^* \supseteq C_l \cap K_k$. A $C_l \setminus \bigcup_{\alpha=i,j,\dots,k} K_\alpha$ tanulópontokat a legközelebbi osztályba soroljuk a $d(t, A) = \min_{u \in A} d(t, u)$ távolság alapján.

1.2. Példa. A mohó algoritmus működése.

Inicializáló lépés:

$$\underline{\underline{V}} = \begin{pmatrix} 6 & 11 & 2 & 8 & 1 \\ 0 & 9 & 7 & 1 & 2 \\ 8 & 7 & 1 & 3 & 2 \\ 1 & 1 & 8 & 1 & 10 \\ 2 & 3 & 0 & 9 & 7 \\ 4 & 8 & 5 & 11 & 2 \\ 1 & 1 & 7 & 1 & 1 \end{pmatrix}, \underline{\underline{\Phi}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \underline{s} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\underline{o} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Első lépés:

$$\underline{\underline{V}} = \begin{pmatrix} 6 & [11] & 2 & 8 & 1 \\ 0 & 9 & 7 & 1 & 2 \\ 8 & 7 & 1 & 3 & 2 \\ 1 & 1 & 8 & 1 & 10 \\ 2 & 3 & 0 & 9 & 7 \\ 4 & 8 & 5 & 11 & 2 \\ 1 & 1 & 7 & 1 & 1 \end{pmatrix}, \underline{\underline{\Phi}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \underline{s} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\underline{o} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Második lépés:

$$\underline{\underline{V}} = \begin{pmatrix} 6 & [11] & 2 & 8 & 1 \\ 0 & 9 & 7 & 1 & 2 \\ 8 & 7 & 1 & 3 & 2 \\ 1 & 1 & 8 & 1 & 10 \\ 2 & 3 & 0 & 9 & 7 \\ 4 & 8 & 5 & [11] & 2 \\ 1 & 1 & 7 & 1 & 1 \end{pmatrix}, \underline{\underline{\Phi}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \underline{s} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$$\underline{o} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Harmadik lépés:

$$\underline{\underline{V}} = \begin{pmatrix} 6 & [11] & 2 & 8 & 1 \\ 0 & 9 & 7 & 1 & 2 \\ 8 & 7 & 1 & 3 & 2 \\ 1 & 1 & 8 & 1 & [10] \\ 2 & 3 & 0 & 9 & 7 \\ 4 & 8 & 5 & [11] & 2 \\ 1 & 1 & 7 & 1 & 1 \end{pmatrix}, \underline{\underline{\Phi}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \underline{s} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$$\underline{o} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Negyedik lépés:

$$\underline{\underline{V}} = \begin{pmatrix} 6 & [11] & 2 & 8 & 1 \\ 0 & [9] & 7 & 1 & 2 \\ 8 & 7 & 1 & 3 & 2 \\ 1 & 1 & 8 & 1 & [10] \\ 2 & 3 & 0 & 9 & 7 \\ 4 & 8 & 5 & [11] & 2 \\ 1 & 1 & 7 & 1 & 1 \end{pmatrix}, \underline{\underline{\Phi}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \underline{s} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$$\underline{o} = \begin{pmatrix} 0 & 2 & 0 & 1 & 1 \end{pmatrix}$$

Ötödik lépés:

$$\underline{\underline{V}} = \begin{pmatrix} 6 & [11] & 2 & 8 & 1 \\ 0 & [9] & 7 & 1 & 2 \\ 8 & 7 & 1 & 3 & 2 \\ 1 & 1 & 8 & 1 & [10] \\ 2 & 3 & 0 & [9] & 7 \\ 4 & 8 & 5 & [11] & 2 \\ 1 & 1 & 7 & 1 & 1 \end{pmatrix}, \underline{\underline{\Phi}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \underline{s} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

$$\underline{o} = \begin{pmatrix} 0 & 2 & 0 & 2 & 1 \end{pmatrix}$$

Hatodik lépés:

$$\underline{\underline{V}} = \begin{pmatrix} 6 & [11] & 2 & 8 & 1 \\ 0 & [9] & 7 & 1 & 2 \\ [8] & 7 & 1 & 3 & 2 \\ 1 & 1 & 8 & 1 & [10] \\ 2 & 3 & 0 & [9] & 7 \\ 4 & 8 & 5 & [11] & 2 \\ 1 & 1 & 7 & 1 & 1 \end{pmatrix}, \underline{\underline{\Phi}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \underline{s} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

$$\underline{o} = \begin{pmatrix} 1 & 2 & 0 & 2 & 1 \end{pmatrix}$$

Hetedik lépés:

$$\underline{\underline{V}} = \begin{pmatrix} 6 & [11] & 2 & 8 & 1 \\ 0 & [9] & 7 & 1 & 2 \\ [8] & 7 & 1 & 3 & 2 \\ 1 & 1 & [8] & 1 & [10] \\ 2 & 3 & 0 & [9] & 7 \\ 4 & 8 & 5 & [11] & 2 \\ 1 & 1 & 7 & 1 & 1 \end{pmatrix}, \underline{\underline{\Phi}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \underline{s} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

$$\underline{o} = \begin{pmatrix} 1 & 2 & 1 & 2 & 1 \end{pmatrix}$$

Nyolcadik lépés:

$$\underline{\underline{V}} = \begin{pmatrix} 6 & [11] & 2 & 8 & 1 \\ 0 & [9] & 7 & 1 & 2 \\ [8] & 7 & 1 & 3 & 2 \\ 1 & 1 & [8] & 1 & [10] \\ 2 & 3 & 0 & [9] & 7 \\ 4 & 8 & 5 & [11] & 2 \\ [1] & 1 & 7 & 1 & 1 \end{pmatrix}, \underline{\underline{\Phi}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}, \underline{s} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\underline{o} = \begin{pmatrix} 2 & 2 & 2 & 1 & 1 \end{pmatrix}$$

Befejezés. Nem lehet több elemet bevonni az $\Im$ relációba. $\sum\limits_{(i,j)\in\Im} \vartheta_{ij} = \sum\limits_{i=1}^{7}\sum\limits_{j=1}^{5} \varphi_{ij} \cdot \vartheta_{ij} = 11+9+8+8+10+9+11+1 = 67.$

A $\underline{\underline{\Phi}}$ értelmezése: a C_1 és C_2 osztályokat, illetve a C_5 és C_6 a osztályokat össze kell vonni, a C_4 osztályt szét kell bontani két osztályba a K_3 és K_5 klaszterek alapján. A más klaszterekbe sorolódott C_4 osztálybeli tanulópontokat a $d(t,A) = \min\limits_{u\in A} d(t,u)$ metrika alapján a közelebbi részosztályba soroljuk K_3 és K_5 közül.

1.3.2. Az osztályok kötetlen átdefiniálása

A KETSKEMÉTY [22] cikk az osztályok kötetlenebb átdefiniálási eljárását javasolja. Az első lépésben az egyes osztályokat „homogén" részhalmazokra partícionáljuk, majd a második lépésben ezen részhalmazok közül az egymáshoz „közelieket" összevonjuk.

1^o A C_i osztályokat homogén alosztályokra bontjuk klaszterezéssel. Ha $\mathcal{T}_n$ eredeti osztályokrabontását $\mathcal{T}_n = \bigcup\limits_{i=1}^{L} C_i$ jelöli, és minden C_i osztályt itt nem részletezett módon az $C_i = \bigcup\limits_{\alpha=1}^{M_i} C_{i\alpha}$ partíciókra bontjuk klaszterezéssel. Jelölje $\Gamma = \{(1,1),...,(1,M_1),(2,1),...,(2,M_2),...,(L,M_L)\}$ az osztálypartíciók indexeinek halmazát, és legyen $M = \sum\limits_{i=1}^{L} M_i$, az indexhalmaz elemeinek száma.

2^o Az $C_\alpha, \alpha \in \Gamma$ osztályok közül a homogéneket összevonjuk, azaz megadunk egy $\Gamma = \bigcup_{i=1}^{L^*} \Gamma_i$ partícionálást. Az új osztályok tanulópontjai a $C_i^* = \bigcup_{\alpha \in \Gamma_i} C_\alpha$ halmazokba fognak esni, $i = 1, 2, ..., L^*$.

A cikkben a második lépés speciális esetre van kidolgozva. Ott az alábbi összevonási algoritmus található.

Feltételezzük, hogy az alakzattér most a p-dimenziós Euklideszi tér. Feltesszük továbbá, hogy az $C_\alpha, \alpha \in \Gamma$ osztályok p-dimenziós normális eloszlást követnek. Ilyen feltételek mellett az osztályok homogenitására statisztikai próba szerkeszthető az ún. *Swain-Fu* metrika segítségével. $\rho_{S-F}(C_\alpha, C_\beta) = \frac{\sqrt{w_\alpha w_\beta}}{\sqrt{w_\alpha}+\sqrt{w_\beta}}$, $(\alpha \neq \beta)$, ahol pl. $w_\alpha = \left(\underline{\bar{x}}_\alpha - \underline{\bar{x}}_\beta\right)^T \underline{\underline{S}}_\alpha^{-1} \left(\underline{\bar{x}}_\alpha - \underline{\bar{x}}_\beta\right)$. A képletekben $\underline{\bar{x}}_\alpha, \underline{\bar{x}}_\beta$ az osztályok átlagvektorai, $\underline{\underline{S}}_\alpha$,pedig az C_α osztály kovarianciamátrixának inverze. A C_α és C_β azonos p-dimenziós normális eloszlásból származása esetén $\frac{(n_\alpha+n_\beta)(n_\alpha-p)}{n_\alpha \cdot n_\beta \cdot p} w_\alpha$ eloszlása $F_{p,n_\alpha-p}$, míg $\frac{(n_\alpha+n_\beta)(n_\beta-p)}{n_\alpha \cdot n_\beta \cdot p} w_\beta$ eloszlása $F_{p,n_\beta-p}$.

Így fennáll

$$(*) \qquad \mathbf{P}\left(\rho_{S-F}(C_\alpha, C_\beta) < h\left(F_\alpha(\varepsilon), F_\beta(\varepsilon)\right)\right) \geq 1 - \varepsilon,$$

ahol pl. az $F_\alpha(\varepsilon)$ kritikus érték úgy van meghatározva a *Fisher*-eloszlás táblázatából, hogy

$$\mathbf{P}\left(F_{p,n_\alpha-p} < \frac{(n_\alpha+n_\beta)(n_\beta-p)}{n_\alpha \cdot n_\beta \cdot p} F_\alpha(\varepsilon)\right) = 1 - \varepsilon \text{ és } h(x,y) = \frac{\sqrt{xy}}{\sqrt{x}+\sqrt{y}}.$$

Tehát a $(*)$ alapján $1-\varepsilon$ szintű statisztikai próbával ellenőrizhetjük az alosztályok homogenitását. Ha két alosztályra feltehető a homogenitás, összevonjuk őket.

Az összevonás algoritmusa:

1^o Legyen $\varepsilon > 0, \underline{\underline{H}} = \left(h\left(F_\alpha(\varepsilon), F_\beta(\varepsilon)\right)\right), \underline{\underline{T}} = \left(\rho_{K-F}(C_\alpha, C_\beta)\right)$, $r_\gamma := \gamma, (\gamma = 1, 2, ..., M)$.

2^o Ha $T_{\alpha\beta} < H_{\alpha\beta}$, akkor $r_\alpha, r_\beta := \min\{r_\alpha, r_\beta\}$.

3^o Az $\underline{r}$ vektor feldolgozása:

$$\left.\begin{array}{l} \text{Ha } r_i = j_1 \neq i \\ r_{j_1} = j_2 \neq j_1 \\ \vdots \\ r_{j_{n-1}} = j_n \neq j_{n-1} \\ \text{de } r_{j_n} = j_n \end{array}\right\} \Longrightarrow r_i := j_n \ (i = 1, 2, ..., M).$$

4^o Az alosztályok összevonása $\underline{r}$ segítségével:
A $C_{\alpha_1}, C_{\alpha_2}, ..., C_{\alpha_k}$ partíciókat akkor vonjuk össze, ha $r_{\alpha_1} = r_{\alpha_2} = ... = r_{\alpha_k}$ teljesül.

1.3.3. Egy dinamikus klaszterezési eljárás metrikus térben

Adott az $(\mathcal{X}, d)$ metrikus tér és benne a $\mathcal{T}_n \subset \mathcal{X}$ tanulópont-halmaz. Tegyük fel, hogy a tanulópontok osztályait az $C_1, C_2, ..., C_L$ halmazok tartalmazzák. $\bigcup_{i=1}^{L} C_i = \mathcal{T}_n$, $C_i \cap C_j = \emptyset$. Klaszterezni fogjuk $\mathcal{T}_n$ elemeit egy dinamikus eljárással. Az algoritmus ismertetése előtt néhány fogalmat bevezetünk. Legyen $K \subset \mathcal{X}$ tetszőleges véges elemszámú halmaz.

1.3. definíció. $c(K) = \arg\min_{x \in K} \max_{y \in K} d(y, x)$ *a* K *centrumeleme.* $c(K)$ *körül szóródnak legkisebb mértékben* K *elemei.*

1.4. definíció. $T(K) = \min_{x \in K} \max_{y \in K} d(y, x)$ *a* K *halmaz tömörségi mérőszáma. (Minél kisebb ez a szám, annál tömörebb* K*.) Tekintsük most a* $\mathcal{T}_n$ *halmaz egy* $K_1, K_2, ..., K_L$ *partícióját! Ezen partíció tömörségén a* $W(K_1, K_2, ..., K_L) = \sum_{i=1}^{L} T(K_i)$ *számot értjük.*

Cél olyan partíció megkeresése amelynek a tömörsége minimális. Az alábbi kétfázisú algoritmus ilyen partíciót konstruál.

1. fázis:
1./1. Kivesszük a $c(C_1), c(C_2), ..., c(C_L)$ pontokat $\mathcal{T}_n$ elemei közül. Jelölje ezeket $t_1, t_2, ..., t_L$. Legyen $c(K_i) := t_i, K_i := \{t_i\}$, $T(K_i) := 0$ $(i = 1, 2, ..., L)$.

1./2. Az összes $t \in \mathcal{T}_n \setminus \{t_1, t_2, ..., t_L\}$ tanulópontot besoroljuk valamelyik klaszterbe. Az t-t akkor soroljuk K_i-hez, ha $\nabla T(K_i) = T(K_i \cup \{t\}) - T(K_i)$ az i-nél a legkisebb. Ekkor $K_i := K_i \cup \{t\}$ és újraszámoljuk $c(K_i)$-t és $T(K_i)$-t.

2. fázis:
2./1. Egy ciklusban sorravesszük az $t \in \mathcal{T}_n$ tanulópontokat. Ha pl. $t \in K_i$, akkor két eset lehet

- vagy $T(K_j \cup \{t\}) + T(K_i \setminus \{t\}) \geq T(K_j) + T(K_i)$ minden $j \neq i$-re, vagyis jobban járunk, ha t-t nem soroljuk át.
- vagy létezik j index, melyre $T(K_j \cup \{t\}) + T(K_i \setminus \{t\}) < T(K_j) + T(K_i)$ és j^* jelöli az ilyen indexek között azt, amelynél a baloldali összeg a legkisebb. Ekkor t-t átsoroljuk: $K_{j^*} := K_{j^*} \cup \{t\}$ és $K_i := K_i \setminus \{t\}$.

2./2. Az előző ciklust annyiszor indítjuk újra, amig van olyan tanulópont, amit át kellett sorolni.

1.5. megjegyzés. *Mivel a $W(K_1, K_2, ..., K_L)$ tömörség csak véges sok értéket vehet fel, és a fenti algoritmus minden lépésében csökkenti a tömörséget, azaz véges lépésben minimalizál.*

1.3.4. A tanulópont-halmaz ellenőrzése klaszterezéssel

Tekintsük most a fenti algoritmus által előállított $K_1, K_2, ..., K_L$ partíciót! Jelölje $\vartheta_{ij} = \sum_{t \in \mathcal{T}_n} I_{\{t \in C_i \wedge t \in K_j\}}$ és legyen $\underline{\underline{V}} = (\vartheta_{ij})$. Tekintsük azt a $h(p) = \sum_{i=1}^{L} \vartheta_{i\alpha_i}$ függvényt, ahol $p = (\alpha_1, \alpha_2, ..., \alpha_L)$ egy permutációja az $(1, 2, ..., L)$-nek. Tegyük fel, hogy a h függvényt a $p^* = (\beta_1, \beta_2, ..., \beta_L)$ permutáció maximalizálja.

1.5. definíció. *Az $A(\mathcal{T}_n, d) = \frac{h(p^*)}{n}$ szám méri a d metrika alkalmasságát $\mathcal{T}_n$-hez.*

Nyilván $0 \leq A(\mathcal{T}_n, d) \leq 1$ és egy d metrika annál alkalmasabb $\mathcal{T}_n$-hez, minél közelebb van $A(\mathcal{T}_n, d)$ az egyhez. A p^* permutációt a $\underline{\underline{V}}$ kontingencia mátrixhoz tartozó hozzárendelési probléma megoldásával határozhatjuk meg, ilyen pl. a magyar módszer. Ha az osztályozáshoz több metrika is szóba jöhet, azt érdemes választani, amelynél az alkalmasság a legnagyobb, hiszen ekkor szeparálódnak leginkább az osztályoknak megfelelően a tanulópontok.

1.4. A tananyag szűrése

A szűrés feladata abban áll, hogy kizárjon a $\mathcal{T}_n$ tanulópont-halmazból olyan tanulópontokat, melyek szélsőségesek abban az értelemben, hogy más jellemzőik vannak, mint saját osztályának „átlagos", tipikus tanulópontjainak. Ezen deviáns

tulajdonságot mutató „outlier" pontok $\mathcal{O}$ halmazát elhagyjuk a $\mathcal{T}_n$ tanulópont-halmazból, vagy az osztályozáskor az elemeit másképpen vesszük figyelembe, amitől az osztályozás pontosságának javulását várjuk. A deviáns- (vagy outlier) pontokat sokféleképpen definiálhatjuk. Outlier az a tanulópont, amely nagyobb valószínűséggel osztályozzuk rosszul, mint jól, de outlier az a pont is, amely a saját osztályának tanulópontjaitól jóval távolabb van, mint az osztály tanulópontjainak egymástól vett átlagos távolsága stb. WILSON [36] az alábbi szűrést javasolta: hagyjuk el azokat a tanulópontokat, amelyeket a legközelebbi k-társ osztályozással rosszul lehet osztályozni a többi $n-1$ tanulópont felhasználásával. Ezek alapján lehet megadni a deviáns tanulópontok *Wilson*-féle értelmezését.

1.6. definíció. *Egy $t \in \mathcal{T}_n$ tanulópont (Wilson- vagy) $\boldsymbol{d}_0$-deviáns, ha t-t a legközelebbi k-társ alapján $\mathcal{T}_n \setminus \{t\}$-ből a k-NN módszer rosszul osztályozza.*

A következő pontban további deviáns tulajdonságokat fogalmazunk meg, amik a tananyag szűrésének különböző eljárásait alapozhatják meg.

1.4.1. Deviancia-tulajdonságok

Az alábbi definíciók mindegyike valamilyen értelemben deviáns tulajdonságot megfogalmazva értelmezi az outlier tanulópontot.
A definíciókban $C(t) = \{z; z \in \mathcal{T}_n, class(z) = class(t)\}$, a t-vel egy osztályba tartozó tanulópontok halmazát jelöli.

Az első definícióban azt a deviáns tulajdonságot fogalmazzuk meg, amikor a t tanulópont k legközelebbi társ súlyainak összege nagyobb az idegen osztályok esetén, mint a t saját osztályánál. A definícióban w_i-vel jelölt súlyokat DUDANI [10] cikkéből vettük át.

1.7. definíció. *Legyenek a t tanulópont k legközelebbi társa $\mathcal{T}_n$-ben $t_1, t_2, ...t_k$, azaz $d(t_1,t) \leq d(t_2,t) \leq \cdots \leq d(t_k,t) \leq d(t,z)$, $z \in \mathcal{T}_n \setminus \{t_1, t_2, ...t_k, t\}$. Legyen $w_i = \frac{d(t_k,t)-d(t_i,t)}{d(t_k,t)-d(t_1,t)}$, ha $d(t_k,t) \neq d(t_1,t)$, és $w_i = 1$ különben. Legyen $w_*(t) = \sum_{\forall i: t_i \in C(t)} w_i$ és $w^*(t) = \sum_{\forall i: t_i \notin C(t)} w_i$. Ha $w_*(t) < w^*(t)$, akkor a t tanulópont $\boldsymbol{d}_1$-deviáns.*

A következő definícióban a t tanulópontot akkor minősítjük szélsőségesnek, ha saját osztálybeli szomszédainak többsége messzebb van t-től, mint az idegen osztálybeli k-adik legközelebbi szomszéd.

1.8. definíció. *Legyenek* $z_1, z_2, \ldots, z_k \notin C(t)$ *olyanok, hogy* $d(z_1,t) \leq d(z_2,t) \leq \cdots \leq d(z_k,t) \leq d(t,y)$, $y \in \mathcal{T}_n \backslash (C(t) \cup \{z_1, z_2, \ldots z_k, t\})$. $p_k(t) = \frac{\sum_{u \in C(t)} I_{\{d(u,t) > d(z_k,t)\}}}{\#C(t)}$. *Ha* $p_k(t) > \frac{1}{2}$, *akkor a* t *tanulópont* k*-szinten* $\boldsymbol{d}_2$*-deviáns.*

A következő deviancia fogalom azt értelmezi, hogy a saját osztálybeli pontok csak kevesebb mint felének a távolsága kisebb t-től, mint az átlagos távolság a tananyagban.

1.9. definíció. *Legyen* $q(t) = \frac{\sum_{u \in C(t)} I_{\{d(u,t) < K\}}}{\#C(t)}$, *ahol* $K = \frac{1}{n(n-1)} \sum_{u,v \in \mathcal{T}_n} d(u,v)$ *az átlagos távolság. A* t *tanulópont* $\boldsymbol{d}_3$*-deviáns, ha* $q(t) < \frac{1}{2}$.

Szintén deviáns egy t tanulópont, ha a $b(t)$ külső távolsága több mint kétszerese az átlagos külső távolságnak, ugyanakkor a $w(t)$ belső távolsága csak fele az átlagos belső távolságnak.

1.10. definíció. *Legyen* $w(t) = \min_{u \in C(t) \backslash \{t\}} d(u,t)$ *a belső távolság,* $b(t) = \min_{v \in \mathcal{T}_n \backslash C(t)} d(v,t)$ *a külső távolság. Legyen továbbá* $K_w = \frac{1}{n} \sum_{t \in \mathcal{T}_n} w(t)$ *és* $K_b = \frac{1}{n} \sum_{t \in \mathcal{T}_n} b(t)$ *az átlagos belső és külső távolságok. A* t *tanulópont* $\boldsymbol{d}_4$*-deviáns, ha* $w(t) > 2K_w$ *és* $b(t) < \frac{1}{2} K_b$.

Outliernek számít a t tanulópont akkor is, ha a saját osztályabeli legközelebbi k szomszéd távolságainak átlaga nagyobb, mint az átlagos távolság.

1.11. definíció. *Legyenek* $z_1, z_2, \ldots, z_k \in C(t) \setminus \{t\}$ *olyanok, hogy* $d(z_1,t) \leq d(z_2,t) \leq \cdots \leq d(z_k,t) \leq d(t,y)$, $y \in C(t) \setminus \{z_1, z_2, \ldots z_k, t\}$. *Ha* $\frac{1}{k} \sum_{i=1}^{k} d(t, z_i) > K$, *ahol* $K = \frac{1}{n(n-1)} \sum_{u,v \in \mathcal{T}_n} d(u,v)$ *akkor a* t *tanulópont* k*-szinten* $\boldsymbol{d}_5$*-deviáns.*

A t tanulópont akkor is deviáns, ha a k legközelebbi szomszédjának a többsége nem a saját osztályához tartozik.

1.12. definíció. *Legyenek* $z_1, z_2, \ldots, z_k \in \mathcal{T}_n \setminus \{t\}$ *olyanok, hogy* $d(z_1,t) \leq d(z_2,t) \leq \cdots \leq d(z_k,t) \leq d(t,y)$, $y \in \mathcal{T}_n \setminus \{z_1, z_2, \ldots z_k, t\}$. *Ha* $r(t) = \sum_{i=1}^{k} I_{\{z_i \in C(t)\}} < \frac{k}{2}$, *akkor* t *a* k*-szinten* $\boldsymbol{d}_6$*-deviáns.*

Akkor is szélsőséges tulajdonságú lehet a t tanulópont, ha az a saját osztályát lefedő gömb centrumelemétől messze helyezkedik el.

1.13. definíció. *Jelölje* c *a* $C(t)$ *centrumelemét (l. a 1.3. definíciót),* R *pedig a takaró sugarát, azaz legyen* $R = \max_{u \in C(t)} d(u,c)$. *A* t *tanulópont* $\boldsymbol{d}_7$*-deviáns, ha* $d(t,c) > \frac{3}{4}R$.

A következő definícióban akkor minősítünk egy tanulópontot deviánsnak, ha a saját osztályából származó legközelebbi k távolság és az idegen osztályból származó legközelebbi k távolság összefésült mintájában a saját osztályhoz tartozó minta rangszámösszege nagy.

1.14. definíció. *Jelölje* $X_i^* = d(t,u_i)$, $i = 1,2,\ldots,k$, $u_i \in C(t) \setminus \{t\}$ a t*-vel azonos osztályba tartozó pontok* t*-től vett távolságai között a* k *legkisebbet,* $Y_i^* = d(t,v_i)$, $i = 1,2,\ldots,k$, $v_i \notin C(t)$ *pedig a* t*-vel idegen osztályba tartozó pontok* t*-től vett távolságai között a* k *legkisebbet. Képezzük a két minta összefésült rendezett mintáját, melyet* $Z_1^* \leq Z_2^* \leq \cdots \leq Z_{2k}^*$*-val jelölünk. Legyen* $r_i = j$, *ha* $X_i^* = Z_j^*$. *Akkor a* t *tanulópont* $\boldsymbol{d}_8$*-deviáns, ha* $\sum_{i=1}^{k} r_i > \frac{3k(k-1)}{2}$.

A tanulópontokon elvégzett klaszterezéssel is kiszűrhetők a deviáns pontok. Ebből a szempontból akkor minősítünk deviánsnak egy tanulópontot, ha nem a saját osztályának megfelelő klaszterbe esik. Az osztályoknak megfelelő klaszterek kiválasztását egy hozzárendelési feladat megoldásával végezhetjük el.

1.15. definíció. *Jelölje* $C_1, C_2, \ldots, C_L$ *az osztályokat,* $K_1, K_2, \ldots, K_L$ *pedig a klasztereket. Legyen* $v_{i,j} = \sum_{u \in \mathcal{T}_n} I_{\{u \in C_i \cap K_j\}}$. *Jelölje továbbá* $(i_1, i_2, \ldots, i_L)$ *azt a permutációt, ahol* $v_{1,i_1} + v_{2,i_2} + \cdots + v_{L,i_L}$ *maximális. Akkor az* $\bigcup_{\alpha=1}^{L} (C_\alpha \setminus K_{i_\alpha})$ *tanulópontok halmaza* $\boldsymbol{d}_9$*-deviáns.*

Tegyük fel, hogy adott $r \geq 2$ különböző deviancia-kritérium (pl. $\mathbf{d}_0 - \mathbf{d}_9$ közül). Rendeljünk mindegyik kritériumhoz egy-egy büntető-súlyt:

$$s_1, s_2, ..., s_r > 0, \sum_{i=1}^{r} s_i = 1.$$

Ekkor értelmezni tudjuk minden tanulópont deviancia súlyát.

1.16. definíció. *A $t \in \mathcal{T}_n$ tanulópont deviancia súlya*

$$dev(t) = \sum_{i=1}^{r} s_i \cdot I_{\{t \text{ megfelel az } i\text{-edik deviancia kritériumnak}\}}.$$

1.4.2. A deviáns tanulópontok felhasználása a tananyag szűréséhez

Egyszempontú szűrés. $\mathcal{D}_n$-ből töröljük azokat a tanulópontokat, amelyek megfelelnek egy adott deviancia-kritériumnak.

Többszempontú szűrések. Válasszunk ki $r \geq 2$ különböző deviancia-kritériumot! Az alábbi szűrési eljárások mindegyiket együttesen veszik figyelembe.

- *Diszjunktív szűrés.* $\mathcal{D}_n$-ből töröljük azokat a tanulópontokat, amelyek valamelyik adott deviancia-kritériumnak megfelelnek.

- *Konjuktív szűrés.* $\mathcal{D}_n$-ből töröljük azokat a tanulópontokat, amelyek az adott deviancia-kritériumok mindegyikének szimultán megfelelnek.

- *Többségi szűrés.* $\mathcal{D}_n$-ből töröljük azokat a tanulópontokat, amelyek az adott r deviancia-kritérium közül legalább $q\,(< r)$-nak megfelelnek.

- *Súlyozott szűrés.* $\mathcal{D}_n$-ből azokat a t tanulópontokat töröljük, melyek deviancia súlya meghalad egy előírt küszöböt: $dev(t) > T_{dev}$.

1.4.3. Deviáns pontok felhasználása k-NN osztályozáskor

Nem feltétlenül kell a deviáns pontokat kizárni az osztályozásból. Elegendő, ha kisebb súllyal vesszük őket figyelembe a döntéskor.

Legyen az $x \in \mathcal{X}$ osztályozandó pont legközelebbi k társa $\mathcal{T}_n$-ben $t_1, t_2, ..., t_k$. Legyen $d_i = d(x, t_i)$, ahol $d_1 \leq d_2 \leq \cdots \leq d_k$. Tekintsük most az $i \in \mathcal{Y}$ osztályt! Az x pont tagsági súlya az i-edik osztályon

$$cw_i(x) = \sum_{j=1}^{k} (1 - dev(t_j)) \cdot w_j \cdot I_{\{class(t_j)=i\}},$$

ahol

$$w_j = \begin{cases} \frac{d_k - d_j}{d_k - d_1}, & \text{ha } d_k > d_1 \\ 1, & \text{ha } d_k = d_1 \end{cases}$$

a j-edik *Dudani*-súly. (L. DUDANI [10].) Az x-et ahhoz az osztályhoz fogjuk sorolni, amelynél a legnagyobb az osztály tagsági súlya:

$$\phi(x) = i, \text{ ha } cw_i(x) \geq cw_j(x) \text{ minden } j \neq i\text{-re.}$$

2. fejezet

A legközelebbi szomszéd gyors megkeresése

2.1. A probléma megfogalmazása, jelölések

Legyen $(\mathcal{X}, d)$ egy metrikus tér. Adott egy $\mathcal{T}_n = \{t_1, t_2, ..., t_n\} \subset \mathcal{X}$ véges elemszámú halmaz, melyet *tanulópont-halmaznak* nevezünk. A $\mathcal{D}_n = \{(t_1, l_1), (t_2, l_2), ..., (t_n, l_n)\} \subset \mathcal{X} \times \{1, 2, ..., L\}$ halmazt nevezzük *tananyagnak*. A $class(t_i) = l_i \in \{1, 2, ..., L\}$ a t_i tanulópont *tanítása*. A tanítások segítségével a tanulópontok halmaza *osztályokra* partícionálható: $\mathcal{T}_n = \bigcup_{\alpha=1}^{L} \mathcal{C}_\alpha$, ahol $t_i, t_j \in \mathcal{C}_\alpha$ esetén $l_i = l_j$ és $t_i \in \mathcal{C}_\alpha,\ t_j \in \mathcal{C}_\beta, \alpha \neq \beta$ esetén $l_i \neq l_j$. Az $x \in \mathcal{X}$ *osztályozandó pont* (vagy *tesztpont* (query point)) *legközelebbi szomszédja (*vagy *társa)* az a $t \in \mathcal{T}_n$ tanulópont, melyre $d(x,t) \leq d(x, t_i)$, $i = 1, 2, ..., n$. A legközelebbi társ módszerhez olyan algoritmusok tartoznak, amelyek megadják az x legközelebbi társát, vagy legalábbis az x legközelebbi társának tanítását. A legközelebbi társ megkeresésének folyamatában újabb és újabb t_i tanulópontot veszünk sorra és számoljuk ki x-szel a $d(x, t_i)$ távolságot. Ha ez megtörtént, azt mondjuk, hogy „feldolgoztuk" a t_i tanulópontot is. A cél itt az, hogy minél kevesebb tanulópont feldolgozásával jussunk el az x legközelebbi társához. Ebben a fejezetben olyan az x legközelebbi társát kereső algoritmusokat elemzünk, amelyeknél nem kell feltétlenül kiszámítani az osztályozandó pont és az összes tanulópont távolságát. A háromszög-egyenlőtlenségen alapuló *kizárási feltételeknek* eleget tevő $t \in \mathcal{T}_n$ tanulópontok biztosan nem lehetnek a legközelebbi társai az x osztályozandó pontnak, így nem kell kiszámolni a $d(x,t)$ távolságokat sem. Egy kereső algoritmus *lépésszáma* azoknak a $d(x,t)$ távolságszámításoknak a száma, amelyet a legközelebbi társ megkereséséig elvégzünk. A lépésszámot $step_n(x)$-szel fogjuk jelölni.

A legközelebbi társ gyors megkeresésének sikeressége nagyrészt függ attól, hogy a tananyag pontjait milyen sorrendben dolgozzuk fel. Egy algoritmust, amely egyértelmű utasítást ad arra, hogy melyik legyen a következő tanulópont a még fel nem dolgozottak közül, *kereső stratégiának* nevezzük, és $\mathcal{S}$ szimbólummal jelöljük. Ezekkel a 2.5 pontban fogunk részletesen foglalkozni.

A keresés folyamatában az x osztályozandó pont újabb és újabb távolságait számoljuk ki a $\mathcal{T}_n$ tanulópontjaitól. Ezeknek a távolságoknak és a $\mathcal{T}_n$ pontjainak egymástól vett távolságait tartalmazó távolságmátrixa ismeretében kizárási feltételek állíthatók fel arra vonatkozólag, hogy a hátralévő tanulópontok közül melyek azok, amelyek biztosan nem lehetnek az x legközelebbi társa. A kizárási feltételek jelölésére a $\mathcal{K}$ szimbólumot használjuk. A kizárási feltételeket kielégítő tanulópontokat elhagyjuk a keresés folyamatából, így távolságukat az x tesztponttól ki sem kell számítani. Ez azokban az esetekben gyorsítja fel a keresést, amikor a d metrikafüggvény kiszámítása hosszadalmas, költséges. Ez pl. a helyzet, ha az alakzattér sokdimenziós vektortér vagy függvénytér (Ilyen alkalmazás pl. a bevezetésben említett hosszútávú meteorológiai előrejelzés). A pontonkénti kizárási feltételekkel a 2.2 pontban foglalkozunk.

A fent említett kizárási feltételeket lehet egyrészt pontonként évényesíteni amikor a kizárást a feldolgozás sorrendjében éppen következő tanulópontra alkalmazzuk. Másrészt történhet az csoportosan is, ha előzőleg megfelelően struktúráljuk a $\mathcal{T}_n$ halmazt. A keresés folyamatában előállnak olyan szituációk, amikor $\mathcal{T}_n$ egy vagy több részhalmazának minden pontjáról egyöntetűen állítható, hogy egyik sem lehet a keresett legközelebbi társ anélkül, hogy a csoport elemeire külön-külön ellenőriznénk a kizárási feltétel teljesülését. Csoportos kizárással a 2.3 és 2.4 pontokban foglalkozunk.

A kizárási feltételek segítségével tehát a legközelebbi társ megkeresésének folyamata gyorsítható. A kizárás módja alapján négy algoritmus típust különböztethetük meg.

- **Szekvenciális kizárás** Az egyik lehetséges séma szerint szekvenciálisan sorravesszük a tanulópontokat, de indokolt esetben elhagyjuk a távolságszámítást az x osztályozandó ponttal. Ilyenkor a megválasztott $\mathcal{S}$ keresési stratégia szerint vesszük az első, t_{i_1} tanulópontot $\mathcal{T}_n$-ből és kiszámoljuk a $d(x, t_{i_1})$ távolságot. Majd vesszük a következő tanulópontot és a $\mathcal{K}$ kizárási feltétel segítségével eldöntjük, hogy feldolgozzuk-e, azaz számoljuk-e a távolságát x-szel, vagy sem. Ha ez a pont (jelöljük t_{i_2}-vel) nem zárható ki a keresésből, kiszámoljuk a $d(x, t_{i_2})$ távolságot, amivel a $\mathcal{K}$ kizárási feltétel is élesíthető. Aztán az $\mathcal{S}$ kereső stratégia előírása alapján vesszük sorra a pontokat, de a távolságot x-szel a vizsgált tanulópont esetében csak akkor számoljuk ki, ha a kizárási feltétel nem vonatkozik rá.

- **Lépésenkénti rostálás** Egy másik lehetséges séma az, ha a $\mathcal{K}$ kizárási feltétel élesedése esetén mindig újra és újra megvizsgáljuk egyenként a még fel

nem dolgozott tanulópontokat, melyek zárhatók ki a keresésből. Ezeket töröljük $\mathcal{T}_n$-ből, és a következő tanulópontot az $\mathcal{S}$ kereső stratégia segítségével a leszűkített tanulópont-halmazból vesszük. Tehát ilyenkor is úgy indul a keresés, hogy az $\mathcal{S}$ kereső stratégia alapján kiválasztjuk az első, t_{i_1} tanulópontot, és kiszámoljuk a $d(x, t_{i_1})$ távolságot. A $\mathcal{K}$ kizárási feltétellel meghatározzuk az *összes* kizárandó pontok $\mathcal{E}_1$ halmazát. A következő tanulópontot $\mathcal{S}$ segítségével az $\mathcal{R}_1 = \mathcal{T}_n \setminus (\mathcal{E}_1 \cup \{t_{i_1}\})$ halmazból választjuk. Folytatva az eljárást tegyük fel, hogy a k-adik lépésben már feldolgoztuk a $t_{i_1}, t_{i_2}, ..., t_{i_k}$ tanulópontokat, azaz kiszámoltuk a $d(x, t_{i_j}), j = 1, 2, ..., k$ távolságokat, meghatároztuk a kizárható pontok $\mathcal{E}_k$ halmazát. Az $\mathcal{S}$ kereső stratégia alapján meghatározzuk a $t_{i_{k+1}}$ tanulópontot az $\mathcal{R}_k = \mathcal{T}_n \setminus (\mathcal{E}_k \cup \{t_{i_1}, t_{i_2}, ..., t_{i_k}\})$ halmazból és számoljuk a $d(x, t_{i_{k+1}})$ távolságot, stb. Az eljárást addig az m lépésig folytatjuk, amíg $\mathcal{R}_m = \emptyset$ nem lesz, vagyis el nem fogynak a tanulópontok. Ekkor az x legközelebbi társa a $\{t_{i_1}, t_{i_2}, ..., t_{i_m}\}$ között van, az amelyiknél $d(x, t_{i_j})$ minimális. A kizárások révén $n-m$ távolságszámítást megtakarítottunk a keresés során.

- **Keresés megállítási szabállyal** A kizárási elvek segítségével *megállítási szabályokat* lehet felállítani. A megállítási szabályok segítségével a még fel nem dolgozott tanulópontok sorba rendezhetők azon elv szerint, hogy a következő feldolgozandó pont az lesz, amelyik a pillanatnyi kizárási feltételnek legkevésbé látszik megfelelni. Az így definiált sorrend szerint feldolgozva a pontokat eljutunk egy olyan végállapotig (STOP), amikor már minden hátralévő tanulópontról biztosan állítható, hogy nem lehet az x legközelebbi társa. Ilyen elven működik a 2.5 pontban részletezett *minimax* kereső algoritmus.

- **Csoportos kizárás** Ha a csoportos kizárás elvét alkalmazzuk a kereséskor, akkor előzetesen partícionálni kell a $\mathcal{T}_n$ halmazt diszjunkt $G_1, G_2, ..., G_s$ részhalmazok uniójára. A kereső algoritmushoz most is lennie kell egy $\mathcal{S}$ kereső algoritmusnak a háttérben. Újabb és újabb tanulópontok feldolgozása után a kialakított csoportok kizárására felállítható kizárási feltételek egyre élesedni fognak, növelve annak az esélyét, hogy valamelyikük elhagyható lesz a keresés folyamatából. Minden pont feldolgozása után megvizsgálandó, hogy a $G_1, G_2, ..., G_s$ közül melyik csoport, vagy csoportok hagyhatók el, mert biztosan nem tartalmazhatják a legközelebbi társat. Az új feldolgozandó tanulópontot ezután a leszűkített tanulópont-halmazból az $\mathcal{S}$ segítségével választjuk ki. Itt is értelemszerűen addig tart a keresés, amíg van feldolgozandó tanulópont.

Az irodalomban többen publikáltak különböző kereső stratégiákról és kizárási feltételekről. SETHI [30] arra az esetre javasol kereső stratégiát, amikor az $\mathcal{X}$ alakzattér lineáris. Ez a keresési elv átvihető tetszőleges metrikus térbe is. Azt javasolta, hogy válasszunk ki tetszőlegesen három tanulópontot; jelöljük ezeket

t_1, t_2, t_3-mal. Számoljuk ki ezek távolságait az x osztályozandó ponttól. Jelöljük ezeket D_1, D_2, D_3-mal! Ezután megkeressük azt a legkisebb $\varepsilon > 0$-át, ahol az

$$A = \bigcap_{i=1}^{3} \left[S\left(t_i, D_i + \varepsilon\right) \setminus S\left(t_i, D_i - \varepsilon\right) \right]$$

ponthalmaznak már van a $\mathcal{T}_n$ tanulópont-halmazzal közös eleme. Ezt a közös elemet tekintjük legközelebbi szomszédnak. A képletben $S\left(x, R\right)$ az x középpontú, R sugarú nyílt gömböt jelöli, tehát A három

$$\varepsilon = \min_{t \in \mathcal{T}_n \setminus \{t_1, t_2, t_3\}} \max_{i=1,2,3} \left| d\left(t, t_i\right) - d\left(x, t_i\right) \right|$$

vastagságú gömbhéj metszete. A javasolt algoritmus nagy hibája, hogy nem garantálja a legközelebbi társ megtalálását. A gömbhéjak metszetébe, még kis ε esetén is az x-től igen távoli tanulópontok is beleshetnek, ha a kiválasztott t_1, t_2, t_3 tanulópontok egymáshoz lényegesen közelebb esnek, mint x-hez. FARAGÓ ET AL. [11] bebizonyították, hogy $\mathbb{R}^p$-ben $p + 2$ tanulóponthoz a fenti gömbhéjas technikával a legközelebbi társ megtalálható. SETHI kereső stratégiájával tehát a $p = 1$ esetben garantált csak a legközelebbi társ megtalálása. Többen, így pl. VIDAL [35], FARAGÓ ET. AL. [11], [12] javasolták a *geometriai*, vagy *minimax* keresést, ami a fenti gömbhéjas keresési technikának az általánosítása. Részletesen a 2.5. szakaszban fogunk vele foglalkozni.

FUKUNAGA ÉS NARENDRA [15] $\mathbb{R}^p$-ben egy kereső fa felépítését javasolta, amelyet bejárva egyre közelebb jutunk a legközelebbi társhoz. A keresés közben kizárási feltétellel a fa egyre több ágát tudjuk törölni, ezáltal a bejárandó utat rövidítjük. A kereső fát hierarchikus klaszterezéssel alakítjuk ki úgy, hogy az összevonások során a klaszterek száma minden lépésben az l-ed részére csökken. A 0-adik generációhoz maga a $\mathcal{T}_n$ tanulópont-halmaz tartozik: $C_1^0 = \mathcal{T}_n$. Az első generáció a $\mathcal{T}_n$ egy l halmazból álló partíciója: $\mathcal{T}_n = \bigcup_{j=1}^{l} C_j^1$. A következő generációban a partíció halmazainak a száma l^2, $C_j^1 = \bigcup_{k=(j-1)l+1}^{jl} C_j^2$, stb. Tegyük fel, hogy a k-adik generációban már csak egyelemű klaszterek találhatók. Ezután minden partícióban kiszámoljuk az átlagvektort, és meghatározzuk azt a sugarat, amelyhez tartozó gömb az átlagvektor körül már lefedi a partíciót. Az első generációból azt a partíciót vesszük először, amelyik átlagvektora a legközelebb van az $x-$hez. Ezen legközelebbi távolsággal egész partíciók zárhatók ki a további vizsgálatból. Az ezekhez tartozó ágakat töröljük a kereső fából. A kiválasztott partíción belül szekvenciálisan haladva, a pillanatnyi legközelebbi távolság egyre csökken, ami által a kizárási feltétel egyre élesedik. Az algoritmusban javasolt kizárási feltétel a később ismertetett $\mathcal{K}_1$ kizárási feltételhez

hasonló, de itt a tér linearitása is erősen ki van használva. Ehhez hasonló, általános metrikus térben érvényes csoportos kizárási technikát tárgyalunk a 2.3. szakaszban. A 2.2. szakaszban ismertetendő $\mathcal{K}_3$-mal jelölt kizárási feltételt egymástól függetlenül javasolta VIDAL [35] és FARAGÓ ET. AL. [12].

A legközelebbi társ gyors megkeresése témakörében a dolgozat célkitűzései és az elért eredmények az alábbiak:

- A metrika általános tulajdonságai alapján milyen kizárási feltételek fogalmazhatók meg? A kizárási feltétel a még fel nem dolgozott tanulópontokra egyenként vonatkozik, és segítségükkel eldönthető az, hogy lehet-e a pont legközelebbi társ, vagy sem. A dolgozatban öt kizárási feltételt fogalmazunk meg, melyekre a 2.1 tételben a $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4, \mathcal{K}_5$ jelölésekkel hivatkozunk. Korábban csak a $\mathcal{K}_3$-hoz hasonlót publikáltak egy-egy kereső algoritmusba beépített formában. ([12],[35]).

- Mi a $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4, \mathcal{K}_5$ kizárási feltételek között meglévő erőkapcsolat? Ezeket a kérdéseket a 2.1 és 2.2 tételekben vizsgáljuk.

- A $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4, \mathcal{K}_5$ kizárási elvek pontonként érvényesíthetők. A dolgozat egy másik célkitűzése az volt, hogy csoportos kizárási elveket konstruáljon. Lehet-e olyan a metrikus terekben érvényes kizárási elveket megfogalmazni, melyek lehetővé teszik a $\mathcal{T}_n$ egy vagy több részhalmaza minden pontjának csoportos elhagyását a folyamatból? A csoportos kizárásnak vektortérben, lineáris térben vannak publikált eredményei, de metrikus terekbe történő átültetésük ennek a dolgozatnak az eredménye, melyek a 2.3 és 2.4 pontban találhatók.

- Hogyan lehet a kizárási elveket a kereső algoritmusokba beépíteni? Egyáltalán, milyen kereső algoritmusok léteznek, létezhetnek? Milyen jó tulajdonságai lehetnek egy kereső algoritmusnak? Hogyan lehet összehasonlítani a kereső algoritmusokat? Ezekkel a kérdésekkel foglalkozunk a 2.5., 2.6., 2.7., és 2.8. pontokban. A 2.5. pontban a lehetséges kereső stratégiákat vesszük sorra, melyekben az a közös, hogy egyértelmű utasítást adnak a tananyag pontjainak feldolgozási sorrendjére. A 2.6. pontban tárgyaljuk azokat a kereső stratégiákat, amelyek kizárási elveken alapulnak. Ezek ennek a dolgozatnak a termékei. A 2.7. pontban néhány, az irodalomban publikált kizárásos kereső algoritmust ismertetünk. A leírásnál módosításokat javaslunk, melyek részben a gyorsítást, részben az általánosabb feltételek melletti alkalmazhatóságot szolgálják. A 2.7. pont második algoritmusa önálló eredmény, egy csoportos kizáráson alapuló kereső stratégia.

- Hogyan lehet a kereső stratégiák hatékonyságát ellenőrizni? Ezzel a kérdéssel a 2.8. pontban foglalkozunk. Egy olyan algoritmust ismertetünk, amivel

különböző kereső stratégiákat lehet összehasonlítani. Az összehasonlítás alapja az, hogy mindegyiket az ideális lépésszámú algoritmushoz hasonlítjuk. Meghatározható egy olyan minimális lépésszám, aminél kevesebb tanulópont feldolgozásával semmiképp sem lehet megtalálni a legközelebbi társat az adott $\mathcal{T}_n$ tananyagban az adott x osztályozandó pontnál. A vizsgált kereső algoritmusok lépésszámát ehhez az ideális értékhez kell hasonlítani. Az elemzéseket számítógépes szimulációval végzett kísérletsorozattal zárjuk.

2.2. Kizárási feltételek

Jelölje $\underline{T} = (d(x,t_1), d(x,t_2), ..., d(x,t_n))^T$ az $x \in \mathcal{X}$ tesztpont *távolságvektorát* a $\mathcal{T}_n$ tanulópont-halmaz elemeitől. A háromszög-egyenlőtlenségből és a metrika folytonosságát jelentő ismert egyenlőtlenség alapján $\underline{T}$ komponenseinek ki kell elégítenie az alábbi reláció rendszert. Minden i, j indexpárra fenn kell állnia, hogy $|T_i - d_{ij}| \leq T_j \leq T_i + d_{ij}$, ahol $d_{ij} = d(t_i, t_j)$. Az, hogy a $\underline{T}$ komponensei között kényszerfeltételek állnak fenn ad lehetőséget arra, hogy kizárási feltételeket fogalmazzunk meg. Tegyük fel, hogy az legközelebbi szomszéd keresése (rövidítve NN-keresés) folyamán már feldolgoztuk a $t_{i_1}, t_{i_2}, ..., t_{i_k}$ tanulópontokat kiszámolva a $d(x,t_{i_1})$, $d(x,t_{i_2})$, ..., $d(x,t_{i_k})$ távolságokat. Vezessük be az alábbi jelöléseket: $t_{NN}^{(k)} = \arg \min_{j=1,...,k} d(x,t_{i_j})$, $d_{\min}^{(k)} = d\left(x, t_{NN}^{(k)}\right)$, $t_{FN}^{(k)} = \arg \max_{j=1,...,k} d(x,t_{i_j})$, $d_{\max}^{(k)} = d\left(x, t_{FN}^{(k)}\right)$.

2.1. tétel. *A $t \in \mathcal{T}_n \setminus \{t_{i_1}, t_{i_2}, ..., t_{i_k}\}$ tanulópont biztosan nem legközelebbi szomszédja x-nek, ha az alábbi kizárási feltételek valamelyike igaz rá:*

$$\mathcal{K}_1: \quad 2d_{\min}^{(k)} < d\left(t, t_{NN}^{(k)}\right);$$

$$\mathcal{K}_2: \quad 0 < \max_{j=1,...,k} \left(d(t,t_{i_j}) - 2d(x,t_{i_j})\right);$$

$$\mathcal{K}_3: \quad d_{\min}^{(k)} < \max_{j=1,...,k} \left|d(t,t_{i_j}) - d(x,t_{i_j})\right|;$$

$$\mathcal{K}_4: \quad d_{\max}^{(k)} - d_{\min}^{(k)} > d\left(t_{FN}^{(k)}, t\right);$$

$$\mathcal{K}_5: \quad 2d_{\min}^{(k)} < d\left(t, t_{NN}^{(k)}\right) \text{ vagy } d_{\max}^{(k)} - d_{\min}^{(k)} > d\left(t_{FN}^{(k)}, t\right);$$

Bizonyítás.
Legyenek $x, a, b \in \mathcal{X}$ tetszőlegesek!

(*) Ha $2d(x,a) < d(a,b)$, akkor $d(x,a) < d(x,b)$.

Ugyanis $2d(x,a) < d(a,b) \leq d(x,a) + d(x,b) \Longrightarrow d(x,a) < d(x,b)$.

$\mathcal{K}_1$: Alkalmazzuk a (*) tulajdonságot a $a = t_{NN}^{(k)}$, $b = t$ szereposztásban!

$\mathcal{K}_2$: $0 < \max\limits_{j=1,\ldots,k} \left(d\left(t, t_{i_j}\right) - 2d\left(x, t_{i_j}\right)\right) \Longrightarrow$

$\Longrightarrow \quad \exists\, 1 \leq j \leq k:\ d\left(t, t_{i_j}\right) > 2d\left(x, t_{i_j}\right)$.

Innen a (*) tulajdonságból az $a = t_{i_j}$, $b = t$ szereposztással kapjuk, hogy $d_{\min}^{(k)} \leq d\left(x, t_{i_j}\right) < d(x,t)$.

$\mathcal{K}_3$: $d\left(t, t_{i_j}\right) \leq d(t,x) + d\left(x, t_{i_j}\right) \Longrightarrow d\left(t, t_{i_j}\right) - d\left(x, t_{i_j}\right) \leq d(t,x)$;

$d\left(x, t_{i_j}\right) \leq d(t,x) + d\left(t, t_{i_j}\right) \Longrightarrow d\left(x, t_{i_j}\right) - d\left(t, t_{i_j}\right) \leq d(t,x)$,

ahonnan következik $\left|d\left(x, t_{i_j}\right) - d\left(t, t_{i_j}\right)\right| \leq d(t,x)$.

Mivel j tetszőleges volt, ezért

$$\max_{j=1,\ldots,k} \left|d\left(x, t_{i_j}\right) - d\left(t, t_{i_j}\right)\right| \leq d(t,x).$$

Tehát, ha fennáll $\mathcal{K}_3$, akkor $d_{\min}^{(k)} < d(t,x)$ is igaz lesz.

$\mathcal{K}_4$: $d(t,x) \geq d\left(t_{FN}^{(k)}, x\right) - d\left(t_{FN}^{(k)}, t\right) >$

$> d\left(t_{FN}^{(k)}, x\right) - \left(d(t_{FN}^{(k)}, x) - d_{\min}^{(k)}\right) = d_{\min}^{(k)}$.

$\mathcal{K}_5$: Nyilvánvalóan igaz, hiszen $\mathcal{K}_1$ és $\mathcal{K}_4$ is igaz volt. ■

A következő tétel a kizárási feltételek kapcsolatát adja meg. A leírásban a $\mathcal{K}_i \Longrightarrow \mathcal{K}_j$ implikáció azt állítja, hogyha $\mathcal{K}_i$ kizár egy tanulópontot, akkor $\mathcal{K}_j$ is kizárja azt, vagyis $\mathcal{K}_j$ nem gyengébb $\mathcal{K}_i$-nél.

2.2. tétel. $\mathcal{K}_i \Longrightarrow \mathcal{K}_j$, *ha* $(i,j) \in \{(1,2),(2,3),(4,3),(5,3)\}$.

Bizonyítás. $\mathcal{K}_1 \Longrightarrow \mathcal{K}_2$

$$2d\left(t_{NN}^{(k)},x\right) \leq d\left(t_{NN}^{(k)},t\right) \Longrightarrow$$

$$0 \leq d\left(t_{NN}^{(k)},t\right) - 2d\left(t_{NN}^{(k)},x\right) \leq \max_{j=1,\ldots,k}\left(d\left(t,t_{i_j}\right) - 2d\left(x,t_{i_j}\right)\right).$$

$\mathcal{K}_2 \Longrightarrow \mathcal{K}_3$

$$0 < \max_{j=1,\ldots,k}\left(d\left(t,t_{i_j}\right) - 2d\left(x,t_{i_j}\right)\right) \Longrightarrow \exists\, 1 \leq j \leq k:$$

$$d_{\min}^{(k)} \leq d\left(x,t_{i_j}\right) < d\left(t,t_{i_j}\right) - d\left(x,t_{i_j}\right) \leq \max_{j=1,\ldots,k}\left|d\left(t,t_{i_j}\right) - d\left(x,t_{i_j}\right)\right|.$$

$\mathcal{K}_4 \Longrightarrow \mathcal{K}_3$

$$d_{\max}^{(k)} - d_{\min}^{(k)} > d\left(t_{FN}^{(k)},t\right) \Longrightarrow$$

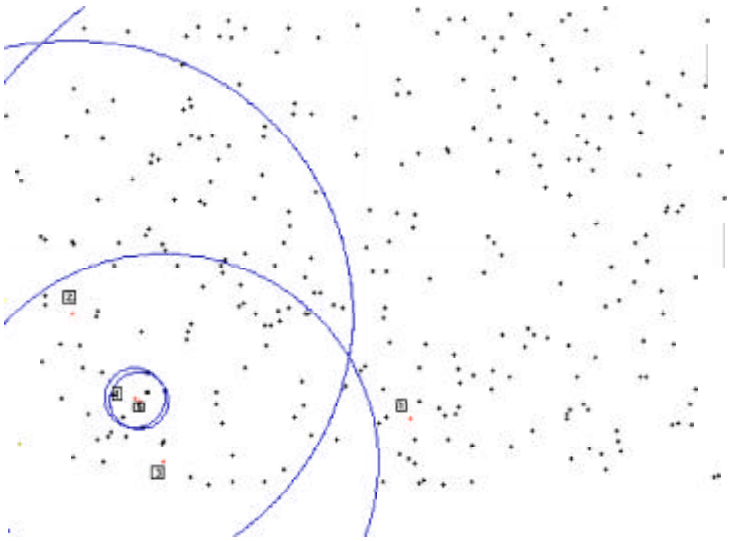
$$d_{\min}^{(k)} < d\left(t_{FN}^{(k)},x\right) - d\left(t_{FN}^{(k)},t\right) \leq \max_{j=1,\ldots,k}\left|d\left(t,t_{i_j}\right) - d\left(x,t_{i_j}\right)\right|.$$

$\mathcal{K}_5 \Longrightarrow \mathcal{K}_3$

Nyilvánvaló, mert $\mathcal{K}_1 \Longrightarrow \mathcal{K}_3$ és $\mathcal{K}_4 \Longrightarrow \mathcal{K}_3$ igaz volt. ■

A $\mathcal{K}_1$ kizárási feltétellel történő keresési folyamatot szemlélteti az 2.1. ábra. Az osztályozandó pontot kis fekete négyzet jelöli az ábra bal alsó felében. A +-szal jelölt tanulópontok a téglalapon egyenletesen oszlanak el. A keresés folyamatában szereplő tanulópontokat halványabban nyomtattuk ki, és a sorszámmal is megcímkéztük. Berajzoltuk azokat a köröket is, amelyen kívüli tanulópontokat kizárunk a további keresésből. Az algoritmus az ötödik lépésben jut el a legközelebbi szomszédhoz. A bejelölt körökön belüli pontok közül véletlenszerűen választjuk ki a legközelebbi tanulópontot.

A következő tétel egyszerű ellenpéldákon keresztül igazolja azt, hogy melyik kizárási feltétel melyik kizárási feltételt nem implikálja általában.



2.1. ábra A $\mathcal{K}_1$ kizárási elv alkalmazása NN-kereséskor.

2.3. tétel. *$\mathcal{K}_i \not\Longrightarrow \mathcal{K}_j$, ha*

$$(i,j) \in \{(3,2),(3,1),(2,1),(3,4),(3,5),(2,4),(4,2),(1,4),(4,1)\}\ .$$

Bizonyítás. Az alábbi egyszerű síkbeli ellenpéldák mutatják, hogy a 2.2 tétel irányai általában nem fordíthatók meg.

$\mathcal{K}_2 \not\Longrightarrow \mathcal{K}_1$:

Legyen $\mathcal{X} = \mathbb{R}^2,\ t_1 = (1,0),\ t_2 = (1.75\ ,\ 0.75),\ t_3 = (-0.5\ ,\ -2),\ x = (0,1)$. A távolságmátrix: $d(t_1,t_2) = \frac{3\sqrt{2}}{4} \approx 1.06, d(t_1,t_3) = 2.5, d(t_2,t_3) = \frac{\sqrt{202}}{4} \approx 3.55$. A pontokat természetes sorrendben dolgozzuk fel: $d(t_1,x) = \sqrt{2} \approx 1.41, d(t_2,x) = \frac{\sqrt{50}}{4} \approx 1.76$. $\mathcal{K}_2$-vel kizárjuk t_3-at, mert $\max\limits_{i=1,2}(d(t_i,t_3) - 2\cdot d(x,t_i)) = d(t_2,t_3) - 2\cdot d(x,t_2) \approx 3.55 - 3.54 > 0$. Viszont $\mathcal{K}_1$-gyel nem zárjuk ki sem t_2-t, sem t_3-mat, hiszen $2.82 \approx 2\cdot d(t_1,x) > d(t_1,t_2) \approx 1.06$ és $2.82 \approx 2\cdot d(t_1,x) > d(t_1,t_3) = 2.5$.

$\mathcal{K}_3 \not\Longrightarrow \mathcal{K}_2$:

Legyen $\mathcal{X} = \mathbb{R}^2,\ t_1 = (0,0),\ t_2 = \left(\frac{4}{5}\sqrt{3}\ ,\ \frac{\sqrt{3}}{5}\right),\ t_3 = \left(\sqrt{3},0\right),\ x = (0,1)$. A távolságmátrix: $d(t_1,t_2) = \sqrt{\frac{51}{25}} \approx 1.42, d(t_1,t_3) = \sqrt{3} \approx 1.73, d(t_2,t_3) =$

$\frac{\sqrt{6}}{5} \approx 0.49$. A pontokat természetes sorrendben dolgozzuk fel: $d(t_1,x)=1, d(t_2,x) \approx 1.53$. $\mathcal{K}_2$-vel nem zárjuk ki t_3-at, mert $\max_{i=1,2}(d(t_i,t_3)-2\cdot d(x,t_i)) = d(t_1,t_3) - 2\cdot d(x,t_1) \approx -0.27 < 0$. Viszont $\mathcal{K}_3$-mal ugyan még nem zárjuk ki t_2-t, de t_3-mat már igen, hiszen $1 = d(t_1,x) > |d(t_1,t_2)-d(t_1,x)| \approx 0.42$, de $1 = d(t_1,x) < \max_{i=1,2}|d(t_i,t_3)-d(x,t_i)| = |d(t_2,t_3)-d(x,t_2)| \approx 1.04$.

$\mathcal{K}_3 \not\Longrightarrow \mathcal{K}_1$:

A $\mathcal{K}_1 \Longrightarrow \mathcal{K}_2$ implikációból triviálisan következik.

$\mathcal{K}_3 \not\Longrightarrow \mathcal{K}_4$:

Legyen $\mathcal{X}=\mathbb{R}^2$, $t_1=(0,0)$, $t_2=\left(\frac{7}{4},\frac{3}{4}\right)$, $t_3=\left(-\frac{1}{2},-2\right)$, $x=(0,1)$. A távolságmátrix: $d(t_1,t_2)=\sqrt{\frac{58}{16}} \approx 1.9, d(t_1,t_3)=\sqrt{\frac{17}{4}} \approx 2.06, d(t_2,t_3)=\frac{\sqrt{202}}{4} \approx 3.55$. A pontokat természetes sorrendben dolgozzuk fel: $d(t_1,x)=1, d(t_2,x)=\sqrt{\frac{50}{16}} \approx 1.77$. $\mathcal{K}_4$ nem zárja ki t_3-mat, mert $0.77 \approx d(t_2,x) - d(t_1,x) = d_{\max} - d_{\min} < d(t_{FN},t_3) = d(t_2,t_3) \approx 3.55$.

Ezzel szemben $\mathcal{K}_3$ igen, hiszen

$1 = d_{\min} = d(t_1,x) < \max_{i=1,2}|d(t_i,t_3)-d(x,t_i)| = |d(t_2,t_3)-d(x,t_2)| \approx 1.78$.

$\mathcal{K}_3 \not\Longrightarrow \mathcal{K}_5$:

Mivel $\mathcal{K}_5 = \mathcal{K}_1 \vee \mathcal{K}_4$ olyan ellenpélda kell, ahol sem $\mathcal{K}_1$-gyel, sem $\mathcal{K}_4$-gyel nem lehet t_3-mat kizárni, míg a $\mathcal{K}_3$-mal igen. Legyen $\mathcal{X}=\mathbb{R}^2$, $t_1=(0,0)$, $t_2=\left(\frac{4\sqrt{3}}{5},\frac{\sqrt{3}}{5}\right)$, $t_3=\left(\sqrt{3},\sqrt{3}\right)$, $x=(0,1)$.

A távolságmátrix: $d(t_1,t_2)=\frac{\sqrt{51}}{5} \approx 1.43, d(t_1,t_3)=\sqrt{6} \approx 2.45, d(t_2,t_3)=\sqrt{\frac{51}{5}} \approx 1.43$.

A pontokat természetes sorrendben dolgozzuk fel: $d(t_1,x)=1, d(t_2,x) \approx 1.53$.

Mivel $2\cdot d(t_1,x)=2 > d(t_1,t_2) \approx 1.43, 2\cdot d(t_1,x)=2 > d(t_1,t_3) \approx 2.45$ valamint $0.53 \approx d(t_2,x)-d(t_1,x) = d_{\max}-d_{\min} < d(t_2,x) \approx 1.43$, ezért $\mathcal{K}_5$ nem zárja ki t_3-mat.

Viszont

$1 = d_{\min} = d(t_1,x) < \max_{i=1,2}|d(t_i,t_3)-d(x,t_i)| = |d(t_1,t_3)-d(x,t_1)| \approx 1.45$

miatt $\mathcal{K}_3$-mal t_3 kizárható.

$\mathcal{K}_2 \not\Longrightarrow \mathcal{K}_4$:

Legyen $\mathcal{X} = \mathbb{R}^2,\ t_1 = (0,0),\ t_2 = \left(\frac{7}{4}, \frac{3}{4}\right),\ t_3 = \left(-\frac{1}{2}, -2\right),\ x = (0,1)$. A távolságmátrix: $d(t_1,t_2) = \sqrt{\frac{58}{16}} \approx 1.9, d(t_1,t_3) = \sqrt{\frac{17}{4}} \approx 2.06, d(t_2,t_3) = \frac{\sqrt{202}}{4} \approx 3.55$. A pontokat természetes sorrendben dolgozzuk fel: $d(t_1,x) = 1, d(t_2,x) = \sqrt{\frac{50}{16}} \approx 1.77$. $\mathcal{K}_4$ nem zárja ki t_3-mat, mert $0.77 \approx d(t_2,x) - d(t_1,x) = d_{\max} - d_{\min} < d(t_{FN},t_3) = d(t_2,t_3) \approx 3.55$. Ezzel szemben $\mathcal{K}_2$ igen, hiszen

$$\max_{i=1,2} (d(t_i,t_3) - 2 \cdot d(x,t_i)) = d(t_1,t_3) - 2 \cdot d(x,t_1) \approx 0.06 > 0.$$

$\mathcal{K}_4 \not\Longrightarrow \mathcal{K}_2$:

Legyen $\mathcal{X} = \mathbb{R}^2,\ t_1 = (0,0),\ t_2 = \left(\sqrt{3}, 0\right),\ t_3 = \left(\frac{4\sqrt{3}}{5}, \frac{\sqrt{3}}{5}\right),\ x = (0,1)$. A távolságmátrix: $d(t_1,t_2) = \sqrt{3} \approx 1.73,\ d(t_1,t_3) = \frac{\sqrt{51}}{5} \approx 1.43,\ d(t_2,t_3) = \frac{\sqrt{6}}{5} \approx 0.49$. A pontokat természetes sorrendben dolgozzuk fel: $d(x,t_1) = 1,\ d(x,t_2) = 2$. $\mathcal{K}_2$ nem zárja ki t_3-at, hiszen $\max_{i=1,2} (d(t_i,t_3) - 2 \cdot d(x,t_i)) = d(t_1,t_3) - 2 \cdot d(x,t_1) \approx -0.57 < 0$. Viszont $\mathcal{K}_4$ kizárja a harmadik pontot, mivel

$$1 = d(t_2,x) - d(t_1,x) = d^{(k)}_{\max} - d^{(k)}_{\min} > d\left(t^{(k)}_{FN}, t_3\right) = d(t_2,t_3) \approx 0.49.$$

$\mathcal{K}_1 \not\Longrightarrow \mathcal{K}_4$:

Legyen $\mathcal{X} = \mathbb{R}^2,\ t_1 = (0,0),\ t_2 = (2,0),\ t_3 = (2,3),\ x = (0,1)$.

A távolságmátrix: $d(t_1,t_2) = 2,\ d(t_1,t_3) = \sqrt{13} \approx 3.6,\ d(t_2,t_3) = 3$. A pontokat természetes sorrendben dolgozzuk fel: $d(x,t_1) = 1,\ d(x,t_2) = \sqrt{5} \approx 2.24$. $\mathcal{K}_1$ nem még zárja t_2-t, de t_3-mat már igen: $2 = 2 \cdot d(t_1,x) \geq d(t_1,t_2) = 2$ és $2 = 2 \cdot d(t_1,x) < d(t_1,t_3) \approx 3.6$. Ezzel szemben $\mathcal{K}_4$ nem zárja ki a harmadik pontot:

$$1.24 \approx d(t_2,x) - d(t_1,x) = d^{(k)}_{\max} - d^{(k)}_{\min} < d\left(t^{(k)}_{FN}, t_3\right) = d(t_2,t_3) = 3.$$

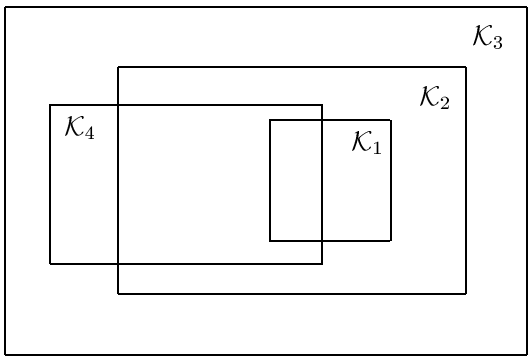
$\mathcal{K}_4 \not\Longrightarrow \mathcal{K}_1$:

Legyen $\mathcal{X} = \mathbb{R}^2,\ t_1 = (0,0),\ t_2 = \left(\sqrt{3}, 0\right),\ t_3 = \left(\frac{4\sqrt{3}}{5}, \frac{\sqrt{3}}{5}\right),\ x = (0,1)$. A távolságmátrix: $d(t_1,t_2) = \sqrt{3} \approx 1.73,\ d(t_1,t_3) = \frac{\sqrt{51}}{5} \approx 1.43,\ d(t_2,t_3) = \frac{\sqrt{6}}{5} \approx 0.49$. A pontokat természetes sorrendben dolgozzuk fel: $d(x,t_1) = 1,\ d(x,t_2) = 2$. $\mathcal{K}_4$ kizárja az első két pont feldolgozása után a harmadikat, hiszen

$$1 = d(t_2,x) - d(t_1,x) = d^{(k)}_{\max} - d^{(k)}_{\min} > d\left(t^{(k)}_{FN}, t_3\right) = d(t_2,t_3) \approx 0.49.$$

$\mathcal{K}_1$ viszont nem zárja ki sem t_2-t sem t_3-mat: $2 = 2 \cdot d(t_1,x) > d(t_1,t_2) \approx 1.73$ és $2 = 2 \cdot d(t_1,x) > d(t_1,t_3) \approx 1.43$. ■

A 2.2. ábrán szemléltetjük a kizárási feltételek kapcsolatát.



2.32. ábra A téglalapok az egyes feltételekkel kizárt tanulópontok halmazait szemléltetik.

2.1. megjegyzés. *a.) A felsorolt kizárási feltételek között a $\mathcal{K}_3$ a legerősebb. Azokat a tanulópontokat, amelyeket a többiek kizárnak, ő is kizárja. Azonban kezelhetőség és programozhatóság szempontjából a $\mathcal{K}_1$ is érdekes lehet.*

b.) A $\mathcal{K}_3$ kizárás programozásakor nem kell a $\max_{j=1,\ldots,k} \left|d\left(t,t_{i_j}\right)-d\left(x,t_{i_j}\right)\right|$ értéket meghatározni, hiszen t már akkor is kizárható, ha létezik olyan $\alpha \in \{i_1,i_2,...,i_k\}$ index, melynél $d_{\min}^{(k)} < |d\left(t,t_\alpha\right)-d\left(x,t_\alpha\right)|$ fennáll.

c.) A kizárási feltételek a legközelebbi k szomszéd keresésekor is alkalmazhatók. Például a $\mathcal{K}_3$ kizárási elv a következőképp módosítandó: a $t \in \mathcal{T}_n \setminus \{t_{i_1},t_{i_2},...,t_{i_r}\}$ tanulópont nem lehet közte az $x \in \mathcal{X}$ k legközelebbi szomszédja között, ha
$d_{k,\min} < \max_{j=1,\ldots,r} \left|d\left(t,t_{i_j}\right)-d\left(x,t_{i_j}\right)\right|$,
ahol $d_{k,\min}$ a k-adik legkisebb távolság $d\left(x,t_{i_1}\right), d\left(x,t_{i_2}\right),...,d\left(x,t_{i_r}\right)$ között.

Az alábbi két tétel a $\mathcal{K}_1$ kizárási feltétellel kapcsolatos.

2.4. tétel. *A $\mathcal{K}_1$ kizárási feltétel antiszimmetrikus és nem tranzitív. (Ha a kizárja b-t, akkor b nem zárhatja ki a-t. Viszont létezhetnek olyan a,b,c tanulópontok, hogy bár a kizárja b-t, b c-t, de a nem zárja ki c-t.)*

Bizonyítás. Ha a kizárja b-t, akkor $2d(x,a) < d(a,b)$ teljesül. Ebből és a háromszög-egyenlőtlenségből $d(x,a) < d(x,b)$. Tehát a háromszög-egyenlőtlenségből $d(a,b) \leq d(a,x) + d(x,b) < 2d(x,b)$, azaz b nem zárhatja ki a-t.

Másrészt tekintsük a síkon az $a = (0,0), b = (3,-1)$ és $c = (\frac{1}{2}, 2)$ tanulópontokat és az $x = (1.3, -0.3)$ tesztpontot! Az Euklideszi-metrikát alkalmazva, egyszerű számolással ellenőrizhetjük, hogy $d(x,a) \approx 1.334, d(x,b) \approx 1.838, d(x,c) \approx 2.435, d(b,a) \approx 3.162, d(a,c) \approx 2.061$ és $d(b,c) \approx 3.905$. Látható, hogy a bár kizárja c-t: $2.668 \approx 2 \cdot d(a,x) < d(a,b) \approx 3.162$, és b kizárja c-t:

$3.676 \approx 2 \cdot d(b,x) < d(b,c) \approx 3.905$,

de a nem zárja ki c-t: $2.668 \approx 2 \cdot d(a,x) > d(a,c) \approx 2.061$ ■

Ha a $\mathcal{K}_1$ kizárási feltételt szigorítjuk, és kétszeres helyett a háromszoros távolságot vesszük, biztosíthatjuk a kizárási elv tranzitivitását.

2.5. tétel. *Ha* $3d(x,a) < d(a,b)$ *és* $3d(x,b) < d(b,c)$, *akkor* $3d(x,a) < d(a,c)$.

Bizonyítás. $3d(x,a) < d(a,b) < d(a,x) + d(b,x) \Longrightarrow 2d(x,a) < d(x,b)$.
$3d(x,b) < d(c,b) < d(c,x) + d(b,x) \Longrightarrow 2d(x,b) < d(x,c)$.

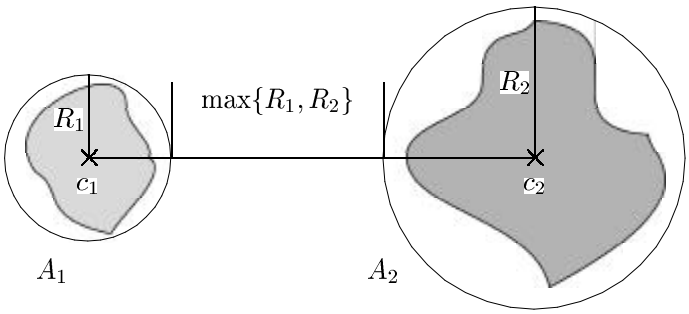
Vagyis $4d(a,x) < d(c,x)$. így $d(a,c) > d(x,c) - d(x,a) > 3d(x,a)$. ■

2.3. A tananyag szeparálása

Tekintsük most a két osztály esetét, azaz legyen $L = 2$! A célunk az, hogy a $\mathcal{T}_n$ tanulópont-halmaz C_1, C_2 osztályainak minél nagyobb elemszámú $M_i \subset C_i$ $(i = 1,2)$ részhalmazait kijelöljük, amelyek szeparáltak abban az értelemben, hogyha az osztályozás során közel jutunk az egyik részhalmaz centrumához, az már garantálni fogja azt, hogy a másik részhalmazban biztosan nem találhatjuk meg a legközelebbi társat.

2.1. definíció. *Legyen* $A \subset \mathcal{T}_n$. *A* $G(c,R) = \{x;\ x \in \mathcal{X},\ d(c,x) \leq R\}$ *c centrumú, R sugarú zárt gömb az A-t lefedő gömb, ha* $c \in A$ *és* $A \subset G(c,R)$. *Az A-t lefedő zárt gömbök között azt a* $G_A(c_A, R_A)$*-val jelöltet fogjuk legkisebb lefedő zárt gömbnek nevezni, amelynél sugár minimális.* c_A *az A halmaz centruma,* R_A *pedig a takaró-sugár.*

2.2. definíció. *Legyenek* $A_1, A_2 \subset \mathcal{T}_n$. *Jelölje rendre* c_1, c_2 *a centrumaikat, és* R_1, R_2 *a takaró-sugaraikat. Az* A_1, A_2 *halmazok szeparálhatóak, ha* $d(c_1,c_2) > R_1 + R_2 + \max\{R_1, R_2\}$.



2.3. ábra Síkbeli szeparálható halmazok

Belátható, ha egy x tesztpont közel esik az A_1 halmaz centrumához, akkor a tőle szeparálható A_2 halmaz mindegyik eleme kizárható lesz a további NN-keresésből.

2.6. tétel. *Legyenek $A_1, A_2 \subset \mathcal{T}_n$ szeparálhatóak. Ha $x \in \mathcal{X}$-re teljesül, hogy $d(x, c_1) < R_1$, akkor $\min_{z \in A_1} d(x, z) < \min_{y \in A_2} d(x, y)$.*

Bizonyítás.
Legyen $z \in A_2$ tetszőleges. A háromszög-egyenlőtlenségből:

$$d(x, u) + d(x, c_1) \geq d(u, c_1) \text{ és } d(c_2, u) + d(u, c_1) \geq d(c_2, c_1),$$

vagyis

$$d(x, u) \geq d(c_1, c_2) - d(c_1, x) - d(u, c_2) \geq d(c_1, c_2) - R_1 - R_2.$$

A szeparálhatósági feltétel miatt:

$$d(c_1, c_2) - R_1 - R_2 \geq \max\{R_1, R_2\} \geq R_1.$$

Ebből már következik, hogy

$$d(x,u) \geq R_1 > d(x,c_1) \geq \min_{z \in A_1} d(x,z).$$

Mivel $z \in A_2$ tetszőleges, ez már igazolja az állítást. ■

Távoli x tesztpontok esetén az alábbi csoportos kizárás lehetséges:

2.7. tétel. *Legyenek $A_1, A_2 \subset \mathcal{T}_n$ szeparálhatóak. Ha $x \in \mathcal{X}$-re teljesül, hogy $d(x,c_1) + R_2 < d(x,c_2)$, akkor $t_{NN}(x) \notin A_2$.*

Bizonyítás. Legyen $y \in A_2$ tetszőleges! Ekkor

$$d(x,c_1) \leq d(x,c_2) - R_2 \text{ és } d(x,c_2) \leq d(c_2,y) + d(y,x) \leq R_2 + d(y,x)$$

amiből már következik az állítás: $d(x,c_1) \leq d(y,x)$. ■

2.8. tétel. *Tegyük fel, hogy már kiszámoltuk a $d(x,t_1), d(x,t_2), ..., d(x,t_k)$ távolságokat. Legyen $d^{(k)}_{\min} = \min_{i=1,...,k} d(x,t_i)$, és $A \subset \mathcal{T}_n \setminus \{t_1, ..., t_k\}$.*
Ha $d^{(k)}_{\min} + R_A < d(x,c_A)$, akkor $t_{NN}(x) \notin A$.

Bizonyítás. Legyen $u \in A$ tetszőleges!
Ekkor $d^{(k)}_{\min} < d(x,c_A) - R_A \leq d(x,c_A) - d(u,c_A) \leq d(x,u)$. ■

2.4. Algoritmusok szeparálható részhalmazok keresésére

Ebben a szakaszban két olyan algoritmust adunk meg, melyek olyan $A_1 \subset C_1, A_2 \subset C_2$ szeparálható részhalmazokat határoznak meg, amik felhasználásával a legközelebbi társ keresésének folyamata majd felgyorsítható lesz.

Leszűkítő algoritmus. Az elsőnek ismertetendő algoritmus az osztályok szűkítésével jut el szeparálható részhalmazokhoz. Kiindulunk a C_1, C_2-t lefedő S_1, S_2 legkisebb gömbökből. Jelölje $c(C_i)$ illetve $R(C_i)$ a megfelelő centrumokat és takaró-sugarakat! Ha már induláskor

$$(**) \qquad d(c(C_1), c(C_2)) > R(C_1) + R(C_2) + \max\{R(C_1), R(C_2)\}$$

lenne, akkor az osztályok tanulópont-halmazai maguk is szeparálhatóak volnának.

Ha nem ez a helyzet, akkor az egymáshoz legközelebbi $x \in C_1$, és $y \in C_2$ tanulópontokat vesszük, ahol $d(x,y) = \min\limits_{u \in C_1, \text{ és } v \in C_2} d(u,v)$. Legyen $A_1 = C_1 \setminus \{x\}$ és $A_2 = C_2 \setminus \{y\}$! Számoljuk újra $c(A_i), R(A_i)$-t $(i = 1,2)$! Nyilván $R(A_i) \leq R(C_i)$. Ha még nem teljesül a (**) feltétel, az A_1, A_2 legközelebbi elemeit hagyjuk el, és számoljuk a maradék halmazok minimális lefedő gömbjének centrumelemét és sugarát. Az algoritmust akkor állítjuk meg, ha valamelyik lépés után már teljesül a (**) feltétel.

Bővítő algoritmus. A második algoritmusban az A_1 és A_2 halmazokat fokozatosan bővítjük. Legyenek $x \in C_1$, és $y \in C_2$ most a legtávolabbi elemek, azaz $d(x,y) = \max\limits_{u \in C_1, \text{ és } v \in C_2} d(u,v)$. Legyenek továbbá a és b a megfelelő legközelebbi osztálytársak, azaz

$$d(a,x) = \min_{u \in C_1} d(u,x) \text{ és } d(b,y) = \min_{v \in C_2} d(v,y).$$

Induláskor

$$A_1 = \{x,a\}, A_2 = \{y,b\},$$

$$c(A_1) = x, R(A_1) = d(a,x), c(A_2) = y, R(A_2) = d(b,y).$$

Ha az így kialakított A_1 és A_2 halmazokra fennáll a (**) feltétel, akkor a bővítést azokkal a $p \in C_1 \setminus A_1, q \in C_2 \setminus A_2$ elemekkel folytatjuk, amelyekre fennállnak

$$\frac{d(x,p)+d(a,p)}{2} = \min_{u \in C_1 \setminus \{x\}} \frac{d(x,u)+d(a,u)}{2}, \text{ illetve } \frac{d(y,q)+d(b,q)}{2} = \min_{v \in C_2 \setminus \{y\}} \frac{d(y,v)+d(b,v)}{2}.$$

Tehát, $A_1 = \{x,a,p\}, A_2 = \{y,b,q\}$ stb. Általában, ha A_1 és A_2-re még fennáll a (**) feltétel, azon $r \in C_1 \setminus A_1, s \in C_2 \setminus A_2$ elemekkel kíséreljük meg az újabb bővítést, melyekre teljesül

$$\frac{\sum\limits_{z \in A_1} d(z,r)}{\sum\limits_{z \in A_1} 1} = \min_{g \in C_1 \setminus A_1} \frac{\sum\limits_{z \in A_1} d(z,g)}{\sum\limits_{z \in A_1} 1}, \text{ illetve } \frac{\sum\limits_{z \in A_2} d(z,s)}{\sum\limits_{z \in A_2} 1} = \min_{g \in C_2 \setminus A_2} \frac{\sum\limits_{z \in A_2} d(z,g)}{\sum\limits_{z \in A_2} 1}.$$

Ha az $A_1 \cup \{r\}$ és $A_2 \cup \{s\}$ halmazokra teljesül még a szeparálhatóság, akkor továbblépünk, különben marad A_1 és A_2.

2.2. megjegyzés. *Ha nem kötjük ki azt, hogy a szeparálható halmazok a C_1, C_2 osztályok részhalmazai legyenek, akkor a fenti algoritmusok kis módosítással C_1', C_2'-nél nagyobb elemszámú $A_1, A_2 \subset \mathcal{T}_n$ szeparálható halmazokat állíthatják elő. A keresés folyamatában ekkor ugyan nagyobb elemszámú tanulópont hagyható el csoportosan, de az osztályozás mégsem biztos hogy felgyorsul, mert a nem kizárt halmazban vegyesen fordulhatnak elő különböző osztályokhoz tartozó tanulópontok.*

2.5. A legközelebbi szomszédot kereső stratégiák

A legközelebbi szomszéd megtalálásának gyorsaságát befolyásolja az is, hogyan választjuk meg a keresés sorrendjét. A keresési folyamat automatizálásához is szükséges annak az elvnek a lerögzítése, hogy a tananyagból melyik legyen a következő feldolgozandó tanulópont. Egy $\mathcal{S}$ keresési stratégia minden $x \in \mathcal{X}$ tesztponthoz definiál egy $(i_1, i_2, ..., i_n)$ index permutációt, amely a tanulópontok indexeinek azt a sorrendjét jelenti amiben a tanulópontokat a keresés folyamán sorravesszük. Tehát $\mathcal{S} : \mathcal{X} \to P_n$ egy függvény, ahol P_n jelenti az $(1, 2, ..., n)$ indexek permutációinak halmazát. A keresés eredményességét jellemző adat a *találati hely*, azaz az $\mathcal{S}(x)$ permutáció azon k pozíciója, ahol az x tesztpont legközelebbi társának indexe áll. Tehát, ha az $\mathcal{S}$ keresési stratégia találati helye k az x tesztpont osztályozásakor, akkor ez azt jelenti, hogy az x legközelebbi társa t_{i_k}. A találati hely egy 1 és n közötti értéket felvevő diszkrét valószínűségi változó, melynek várható értékének n-edrésze az $\mathcal{S}$ stratégia *eredményessége.* Jelölésben $eff(\mathcal{S})$.

A kereső stratégiáknak alapvetően két típusa van. A szekvenciális kereső stratégiák előre rögzítenek egy $(i_1, i_2, ..., i_n)$ kereső sorrendet, és minden $x \in \mathcal{X}$ tesztpont esetén ezt alkalmazzuk. (1. és 2. stratégiák.) A kizáráson alapuló stratégiák a keresés sorrendjét az x tesztponthoz igazítják (3. és 4. stratégiák).

Példák keresési stratégiákra.

1. Véletlen keresés, $\mathcal{S}_r$

   Minden egyes lépésben a hátralévő tanulópontok közül egyet véletlenszerűen választunk ki. Tehát ilyenkor bármelyik permutációt ugyanakkora, $\frac{1}{n!}$ valószínűséggel választhatjuk ki. Ilyenkor a találati hely bárhol, ugyanakkora valószínűséggel előfordulhat, vagyis $eff(\mathcal{S}_r) = \frac{1}{n} \sum_{i=1}^{n} i = \frac{n+1}{2}$.

2. Keresés kötött sorrendben (determinisztikus keresés), $\mathcal{S}_d$

Ilyenkor a $\mathcal{D}_n$ tananyag alapján definiálunk egy $(\gamma_1, \gamma_2, ..., \gamma_n)$ keresési sorrendet, és minden $x \in X$ tesztpont esetén ezt fogjuk alkalmazni. Jelölje $Y_1 = d(x, t_{\gamma_1}), Y_2 = d(x, t_{\gamma_2}), ..., Y_n = d(x, t_{\gamma_n})$ a tesztpontnak a tanulópontoktól vett véletlen távolságait. Ha $\mathbf{P}(Y_1 \leq Y_2 \leq \cdots \leq Y_n) \geq \mathbf{P}(Y_{i_1} \leq Y_{i_2} \leq \cdots \leq Y_{i_n})$ tetszőleges $(i_1, i_2, ..., i_n)$ permutációnál, akkor a $(\gamma_1, \gamma_2, ..., \gamma_n)$ permutációt definiáló $\mathcal{S}_d$ permutáció *optimális*. Az alábbi algoritmus az optimális determinisztikus keresési stratégiát kísérli meg előállítani azáltal, hogy a fenti valószínűség relációt a relatív gyakoriságok relációjával helyettesíti amikor a tanulópontokat osztályozzuk a tananyag segítségével. Jelölje h_{ik} a t_i tanulópont k-adik legközelebbi társának indexét $1, 2, ..., i-1, i+1, ..., n$ közül. Jelölje o_{ij} annak a gyakoriságát, hányszor volt t_i a j-edik legközelebbi társ az osztályozás során:

$$o_{ij} = \sum_{\substack{\alpha=1 \\ \alpha \neq i}}^{n} I\{h_{\alpha j} = i\}.$$

Legyen $\underline{o}_i = (o_{i1}, o_{i2}, ..., o_{in-1})^T$, $\left(\sum_{j=1}^{n-1} o_{ij} = n-1 \right)$. Rendezzük sorrendbe az $\underline{o}_1, \underline{o}_2, ..., \underline{o}_n$ vektorokat úgy, hogy $\underline{o}_i > \underline{o}_j$ legyen, ha $o_{i1} = o_{j1}, o_{i2} = o_{j2}, ..., o_{i\,k-1} = o_{j\,k-1}$, de $o_{ik} > o_{jk}$. Ha $\underline{o}_i = \underline{o}_j$ akkor definíció szerint $\underline{o}_i > \underline{o}_j$-t vegyünk, ha $i < j$. (Ha az $\underline{o}_i$ vektorok dimenzióját csökkentjük azzal, hogy csak az első $k < n-1$ legközelebbi szomszédot figyeljük, akkor közöttük gyakrabban fordulhat elő egyezés.) Tegyük fel, hogy az előbbi rendezéssel a $\underline{o}_{\gamma_1} > \underline{o}_{\gamma_2} > \cdots > \underline{o}_{\gamma_n}$ sorrendet kaptuk Ekkor a keresési sorrend $(\gamma_1, \gamma_2, ..., \gamma_n)$ lesz. Megbecsülhetjük ehhez a kereséshez tartozó eredményességet is, az alábbi módon. Jelölje z_i a t_i tanulópont legközelebbi szoszédjának találati helyét $(\gamma_1, \gamma_2, ..., \gamma_n)$-ben: $z_i = k$, ha $h_{i\gamma_k} = 1$. Ekkor $eff(\mathcal{S}_d) \approx \frac{1}{n} \sum_{i=1}^{n} z_i$. Az algoritmust értelemszerűen módosíthatjuk arra az esetre, amikor rendelkezésre áll egy $\mathcal{C}_m = \{x_1, x_2, ..., x_m\}$ tesztponthalmaz. o_{ij} ilyenkor azt jelenti, hogy a t_i tanulópont hányszor volt a tesztpontok osztályozásakor a j-edik legközelebbi szomszéd. Most $\sum_{j=1}^{n} o_{ij} = m$. Az eredményességet az előbbiekhez hasonlóan becsülhetjük.

3. Geometriai vagy minimax keresés, $\mathcal{S}_{mm}$

Tegyük fel, hogy már feldolgoztuk a $t_{\gamma_1}, t_{\gamma_2}, ..., t_{\gamma_n}$ tanulópontokat.
Definiáljuk a $g_k(t) = \max_{i=1,...,k} |d(t, t_{\gamma_i}) - d(t_{\gamma_i}, x)|$, $t \in \mathcal{T} \setminus \{t_{\gamma_1}, t_{\gamma_2}, ..., t_{\gamma_n}\}$ függvényt.
$g_k(t)$ tulajdonságai:
i^o $\quad g_k(x) = 0,\ g_k(t) \geq 0;$
ii^o $\quad 0 \leq g_1(t) \leq g_2(t) \leq \cdots \leq g_k(t) \leq d(t, x) \leq d(t_{FN}, x) = d_{\max}$
iii^o $\quad$ Ha $g_k(t) > d_{\min} = \min_{i=1,...,k} d(t_{\gamma_i}, x)$, akkor t biztosan nem lehet x

legközelebbi társa; (Ez a $\mathcal{K}_3$ kizárási feltétel.)

iv^o $g_k(t_{\gamma_i}) = d(t_{\gamma_i}, x)$. Speciálisan, ha a t_{NN}, a legközelebbi társ már $t_{\gamma_1}, t_{\gamma_2}, ..., t_{\gamma_n}$ között van, $g_k(t_{NN}) = d_{\min}$.
v^o $0 \leq g_1(t) \leq g_2(t) \leq \cdots \leq g_n(t) = d(t,x) \Longrightarrow d_{\min} = g_n(t_{NN}) \leq g_n(t) = d(t,x)$.
A minimax keresési stratégiánál mindig az lesz a következő tanulópont, amelyik minimalizálja $g_k(t)$-t:

$$g(t_{\gamma_{k+1}}) = \min_{t \in \mathcal{T} \setminus \{t_{\gamma_1}, t_{\gamma_2}, ..., t_{\gamma_k}\}} \max_{i=1,...,k} |d(t, t_{\gamma_i}) - d(t_{\gamma_i}, x)|.$$

Tehát azzal a tanulóponttal folytatjuk a keresést, amely a $\mathcal{K}_3$ kizárási feltétel szerint legkevésbé zárható ki. Másrészt ez a t tanulópont hasonlít leginkább x-re abban az értelemben, hogy ő is kb. olyan távolságra van a már feldolgozott tanulópontoktól, mint x, így feltehetőleg a metrikus térben hozzá közel fog esni.

4. Keresés kizárással, $\mathcal{S}_e$

Tekintsünk egy $\mathcal{K}$ kizárási feltételt és egy $\mathcal{S}$ keresési stratégiát! A kettő kombinációjából kapjuk az $\mathcal{S}_e$ kevert keresési stratégiát:
1. lépés: $\mathcal{S}$-sel meghatározzuk a keresés első $(\gamma_1, \gamma_2, ..., \gamma_k)$ elemét.
2. lépés: $\mathcal{K}(\gamma_1, \gamma_2, ..., \gamma_k)$ segítségével redukáljuk a $\mathcal{T}$ tanulópont-halmazt azáltal, hogy kizárunk tanulópontokat a vizsgálatból. Ha $\mathcal{T}$ kiürül, megállunk.
3. lépés: $\mathcal{S}$-sel a maradék halmazból meghatározunk egy vagy több újabb $\gamma_{k+1}, \gamma_{k+2}, ...$ indexhez tartozó tanulópontokat, majd visszatérünk a *2.lépésre.*
Példa $\mathcal{S}_e$ kevert startégiára:

i^o $\mathcal{S}_d$ eljárással meghatározzuk γ_1-et: $d^{(1)}_{\min} = d(t, t_{\gamma_1})$, $t^{(1)}_{NN} = t_{\gamma_1}$;
ii^o $\mathcal{K}_3$-al kizárjuk azokat a t tanulópontokat, melyekre

$$\left| d(t, t_{\gamma_1}) - d^{(1)}_{\min} \right| > d^{(1)}_{\min};$$

iii^o $\mathcal{S}_{mm}$-mel dolgozunk tovább: azt a t_{γ_2} tanulópontot vesszük a redukált $\mathcal{T}_r$ tananyagból, melyre $g_1(t_{\gamma_2}) = \min_{t \in T_r} |d(t, t_{\gamma_1}) - d(t_{\gamma_1}, x)|$,

$$d^{(2)}_{\min} = \min\{d(t_{\gamma_1}, x), d(t_{\gamma_2}, x)\}, \; t^{(1)}_{NN} = \arg\min\{d(t_{\gamma_1}, x), d(t_{\gamma_2}, x)\};$$

iv^o $\mathcal{K}_3$-al újabb tanulópontokat kizárunk $\mathcal{T}$-ből, stb.

2.3. megjegyzés. *iii°-nél vehetjük az $\mathcal{S}_d$ keresés által definiált $(\gamma_1, \gamma_2, ..., \gamma_n)$ permutációból azt a legkisebb indexet is, amelyet $\mathcal{K}_3$ még nem zárt ki. Ugyanis, ha már csak kevés tanulópont van hátra, a kizárási feltételek figyelembevétele már lasúbb művelet lehet, mint az összes még ki nem számolt távolság kiszámítása az osztályozandó pont és a tanulópontok között.*

2.6. A kizárásos kereső stratégia alaposssága

Jelölje $x \in \mathcal{X}$ az osztályozandó pontot, $\mathcal{T}_n$ a tanulópont-halmazt, $\mathcal{K}$ a kizárási elvet, $\mathcal{S}_e$ pedig a kizárásos kereső stratégiát. Jelölje $\mathcal{P}_m = \{t_{i_1}, t_{i_2}, ..., t_{i_m}\}$ az m-edik kereső lépés után a már „feldolgozott" tanulópontok halmazát. A feldolgozáson azt értjük, hogy kiszámoltuk a $d\left(t_{i_j}, x\right), j = 1, 2, ..., m$ távolságokat.

$\mathcal{P}_m$-et az $\mathcal{S}_e$ kereső stratégia m-edik lépésbeli *janicsárhalmazának* nevezzük. $\mathcal{P}_m$ felhasználásával alkalmazzuk a $\mathcal{K}$ kizárási elvet a $\mathcal{T}_n \setminus \mathcal{P}_m$ tanulópontok esetében. Az

$\mathcal{E}_m = \{t \in \mathcal{T}_n \setminus \mathcal{P}_m$, $\mathcal{K}$-val kizárható, hogy t az x legközelebbi társa legyen$\}$

halmazt az kereső stratégia m-lépésbeli *hulladékhalmazának* nevezzük. $\mathcal{E}_m$ tehát azokat a pontokat tartalmazza, melyeket nem fogunk a keresés során „feldolgozni", azaz az x-szel való távolságukat kiszámolni, mert biztosan nem lehetnek az x legközelebbi társa.

2.3. definíció. *Az $\mathcal{S}_e$ kereső stratágia alapos, ha nincsenek olyan $u \in \mathcal{T}_n \setminus (\mathcal{P}_m \cup \mathcal{E}_m)$ és $t \in \mathcal{E}_m$ tanulópontok, ahol az teljesülne, hogy a $\mathcal{P}_m \cup \{t\}$ janicsárhalmazzal $\mathcal{K}$ kizárná u-t. Vagyis akkor alapos egy $\mathcal{S}$ kereső stratégia, ha nem zár ki olyan pontot, amellyel a kizárási elv élesíthető lenne.*

2.4. megjegyzés. *A kizárásos kereső stratégia alaposssága és a kizárási elv élessége egymás ellen dolgozik. Ha éles kizárási elvet alkalmazunk könnyen előfordulhat, hogy olyan pontot is kizárunk,- ami bár nem lehet a legközelebbi társ-, de őt feldolgozva sok más tanulópontot kizárhatnánk a segítségével.*

2.9. tétel. *Bármely $\mathcal{S}$ kereső stratégia a $\mathcal{K}_1$ kizárási elvvel kombinálva alapos.*

Bizonyítás. Mivel $\mathcal{K}_1$ a mindenkori legkisebb távolság segítségével zár ki pontokat, és egy már kizárt t tanulópont távolsága x-től nagyobb mint a pillanatnyi $d_{\min}$ minimális távolság, azaz $d_{\min} = \min\limits_{z \in \mathcal{P}_m} d(z, x) = \min\limits_{z \in \mathcal{P}_m \cup \{t\}} d(z, x)$,

ezért a $\mathcal{P}_m$ és $\mathcal{P}_m \cup \{t\}$ janicsárhalmazokkal ugyanazok a tanulópontok zárhatók ki $\mathcal{T}_n \setminus (\mathcal{P}_m \cup \{t\})$-ből. ■

2.10. tétel. *Az $\mathcal{S}_{mm}$ minimax kereső stratégia a $\mathcal{K}_3$ kizárási elvvel kombinálva alapos.*

Bizonyítás. A kereső algoritmus egy megállítási szabállyal müködik. Addig vesszük sorra a $t_{i_1}, t_{i_2}, ..., t_{i_m}, ...$ tanulópontokat kialakítva a $\mathcal{P}_1 \subseteq \mathcal{P}_2 \subseteq \cdots \subseteq \mathcal{P}_m \subseteq \cdots$ janicsárhalmaz-sorozatot, mígnem az m-edik soronkövetkező $t_{i_{m+1}}$ pontra először nem lesz érvényes a megállítási szabály, miszerint

$$\max_{u \in \mathcal{P}_m} \left| d\left(t_{i_{m+1}}, u\right) - d\left(u, x\right) \right| = \min_{t \in \mathcal{T}_n \setminus \mathcal{P}_m} \max_{u \in \mathcal{P}_m} \left| d\left(t_i, u\right) - d\left(u, x\right) \right| > d_{\min}.$$

Ilyenkor $\mathcal{E}_1 = \mathcal{E}_2 = \cdots = \mathcal{E}_{m-1} = \emptyset$ és $\mathcal{E}_m = \mathcal{T}_n \setminus \mathcal{P}_m$. Látható, hogy minden $t \in \mathcal{E}_m$ esetén a $\mathcal{K}_3$ kizárási elv érvényes, azaz t-vel vagy anélkül a $\mathcal{P}_m$ janicsárhalmazzal $\mathcal{K}_3$ minden egyéb pontot kizár, vagyis $\mathcal{S}_{mm}$ alapos. ■

Nem minden kizárásos kereső stratégia alapos. Tekintsük az $\mathcal{S}_r$ véletlen kereső algoritmust a $\mathcal{K}_2$ kizárással kombinálva. A keresés most szekvenciális, azaz minden újabb tanulópont feldolgozása után megvizsgáljuk, hogy a hátralévő tanulópontok közül melyeket lehet kizárni a $\mathcal{K}_2$ kizárási elv segítségével. A 2.4. tétel ellenpéldáját fogjuk most is felhasználni. Tekintsük a síkon a $t_1 = (0,0), t_2 = (3,-1)$ és $t_3 = (\frac{1}{2}, 2), t_4, t_5, ...$ tanulópontokat és az $x = (1.3, -0.3)$ tesztpontot! Tegyük fel, hogy a pontok úgy vannak indexezve, ahogy a véletlen keresés során sorra fognak kerülni. Az Euklideszi-metrikát alkalmazva, egyszerű számolással ellenőrizhetjük, hogy $d(x,t_1) \approx 1.334$, $d(x,t_2) \approx 1.838$, $d(x,t_3) \approx 2.435$, $d(t_1,t_2) \approx 3.162$, $d(t_1,t_3) \approx 2.061$ és $d(t_2,t_3) \approx 3.905$. A keresés első lépése t_1 feldolgozása: $d(x,t_1) \approx 1.334$, $\mathcal{P}_1 = \{t_1\}$. Mivel $2 \cdot d(x,t_1) \approx 2.668 < 3.162 \approx d(t_1,t_2)$, de $2.668 > 2.061 \approx d(t_1,t_3)$, ezért t_2-őt kizárjuk, és t_3 lesz a következő feldolgozandó pont: $\mathcal{E}_1 = \{t_2\}$. Viszont, ha vesszük a $\mathcal{P}_1 \cup \{t_2\}$ janicsárhalmazt, akkor t_3 kizárható a $\mathcal{K}_2$ elvvel, hiszen $\max_{i=1,2} \left(d\left(t_i, t_3\right) - 2 \cdot d\left(x, t_i\right)\right) = d\left(t_2, t_3\right) - 2 \cdot d\left(x, t_2\right) \approx -0.229 < 0$. Tehát az algoritmus nem alapos! Ha egy gyengébb kizárási elvvel dolgoznánk, a keresés alapos lehetne. Ha $\mathcal{K}_2$-őt az alábbiak szerint gyengítjük:

$\mathcal{K}_2^*$: a t tanulópont kizárható, ha $\max_{j=1,...,m} \left(d\left(t_{i_j}, t\right) - 3 \cdot d\left(x, t_{i_j}\right)\right) > 0$,

akkor a 2.5 tétel alapján megmutatható, hogy a fenti keresés már alapos lenne.

Kereső algoritmus megállítási szabállyal

Egy $\mathcal{S}_e$ kereső stratégiát kombinálni lehet egy megállítási szabállyal is. A kizárásokat nem pontonként hajtjuk végre, hanem megállunk amikor már garancia van arra, hogy a hátralévő pontok nem tartalmazhatják a legközelebbi társat. A keresést tehát addig folytatjuk, amíg egy megállítási feltétel nem teljesül. A *megállítási szabály* valamilyen $\mathcal{K}$ kizárási elv alapján van megfogalmazva: nem kell folytatni a keresést, mert az eddig feldolgozott pontok a $\mathcal{K}$ kizárási elv alapján beláthatóan biztosan tartalmazzák már a legközelebbi társat. Ilyenkor a hulladékhalmazok sorozatára az teljesül, hogy $\mathcal{E}_1 = \mathcal{E}_2 = \cdots = \mathcal{E}_{m-1} = \emptyset$ és $\mathcal{E}_m = \mathcal{T}_n \setminus \mathcal{P}_m$. Ebből következőleg a megállítási szabállyal működő keresési stratégiák mindig alaposak. Az alábbiakban bemutatunk két megállítási szabályt alkalmazó kereső stratégiát.

Kereső stratégia a $\mathcal{K}_4$ kizárási elven alapuló megállítással

1^o Az első, t_{i_1} tanulópontot véletlenszerűen választjuk. Legyen

$$t_{i_2} = \arg \max_{t \in \mathcal{T}_n \setminus \{t_{i_1}\}} d\left(t, t_{i_1}\right),$$

$$t_{FN}^{(2)} = \arg \max_{j=1,2} d\left(x, t_{i_j}\right), \; t_{NN}^{(2)} = \arg \min_{j=1,2} d\left(x, t_{i_j}\right),$$

$$d_{\max}^{(2)} = d\left(x, t_{FN}^{(2)}\right), \; d_{min}^{(2)} = d\left(x, t_{NN}^{(2)}\right).$$

2^o Tegyük fel, hogy már feldolgoztunk m tanulópontot: $\mathcal{P}_{(m)} = \{t_{i_1}, t_{i_2}, ..., t_{i_m}\}$,

$$t_{FN}^{(m)} = \arg \max_{j=1,\ldots,m} d\left(x, t_{i_j}\right), \; t_{NN}^{(m)} = \arg \min_{j=1,\ldots,m} d\left(x, t_{i_j}\right),$$

$$d_{\max}^{(m)} = d\left(x, t_{FN}^{(m)}\right), \; d_{\min}^{m} = d\left(x, t_{NN}^{(m)}\right).$$

A kereső algoritmus következő eleme:

$$t_{i_{m+1}} = \arg \max_{t \in \mathcal{T}_n \setminus \mathcal{P}_m} d\left(t, t_{FN}^{(m)}\right).$$

Megállítási szabály Ha , $d\left(t_{i_{m+1}}, t_{FN}^{(m)}\right) < d_{\max}^{(m)} - d_{\min}^{(m)}$. az algoritmus leáll, hiszen minden $t \in \mathcal{T}_n \setminus \mathcal{P}_m$ pont esetén érvényes a $\mathcal{K}_4$ kizárási elv.

Kereső stratégia a $\mathcal{K}_5$ kizárási elven alapuló megállítással

1^o Az első, t_{i_1} tanulópontot véletlenszerűen választjuk, $\mathcal{P}_1 = \{t_{i_1}\}$.

2^o A második,

t_{i_2} tanulópont az lesz, amelyre $t_{i_2} = \arg \max_{u \in \mathcal{T}_n \setminus \{t_{i_1}\}} d(u, t_{i_1})$, $\mathcal{P}_2 = \{t_{i_1}, t_{i_2}\}$.

Legyen

$t_{FN}^{(2)} = \arg \max_{j=1,2} d(x, t_{i_j})$, $t_{NN}^{(2)} = \arg \min_{j=1,2} d(x, t_{i_j})$, $d_{\max}^{(2)} = d\left(x, t_{FN}^{(2)}\right)$, $d_{min}^{(2)} = d\left(x, t_{NN}^{(2)}\right)$.

3^o A harmadik, t_{i_3} tanulópont az lesz, amelyre

$t_{i_3} = \arg \min_{u \in \mathcal{T}_n \setminus \mathcal{P}_2} \left[\max\left\{2 \cdot \left(d_{\max} - d\left(u, t_{NN}^{(2)}\right)\right), d\left(u, t_{NN}^{(2)}\right)\right\}\right]$

- Ha $\max\left\{2 \cdot \left(d_{\max}^{(2)} - d\left(t_{i_3}, t_{NN}^{(2)}\right)\right), d\left(t_{i_3}, t_{NN}^{(2)}\right)\right\} > 2 \cdot d_{\min}^{(2)}$, akkor az algoritmus leáll. A minimális távolság $d_{\min}^{(2)}$, az x legközelebbi szomszédja $t_{NN}^{(2)}$. STOP.

- Ha $\max\left\{2 \cdot \left(d_{\max}^{(2)} - d\left(t_{i_3}, t_{NN}^{(2)}\right)\right), d\left(t_{i_3}, t_{NN}^{(2)}\right)\right\} \leq 2 \cdot d_{\min}^{(2)}$, akkor $\mathcal{P}_3 = \mathcal{P}_2 \cup \{t_{i_3}\}$, $t_{FN}^{(3)} = \arg \max_{u \in \mathcal{P}_3} d(x, u)$, $t_{NN}^{(3)} = \arg \min_{u \in \mathcal{P}_3} d(x, u)$, $d_{\max} = d\left(x, t_{FN}^{(3)}\right)$, $d_{min}^{(3)} = d\left(x, t_{NN}^{(3)}\right)$, és a következő kereső lépésnél folytatjuk.

$\vdots$

m^o A következő, t_{i_m} tanulópont az lesz, amelyre

$t_{i_m} = \arg \min_{u \in \mathcal{T}_n \setminus \mathcal{P}_{m-1}} \left[\max\left\{2 \cdot \left(d_{\max}^{(m-1)} - d\left(u, t_{NN}^{(m-1)}\right)\right), d\left(u, t_{NN}^{(m-1)}\right)\right\}\right]$.

- (**Megállítási szabály**)
Ha $\max\left\{2 \cdot \left(d_{\max}^{(m-1)} - d\left(t_{i_m}, t_{NN}^{(m-1)}\right)\right), d\left(t_{i_m}, t_{NN}^{(m-1)}\right)\right\} > 2 \cdot d_{\min}^{(m-1)}$, akkor az algoritmus leáll. A minimális távolság $d_{\min}^{(m-1)}$, az x legközelebbi szomszédja $t_{NN}^{(m-1)}$. STOP.

- Ha $\max\left\{2 \cdot \left(d_{\max}^{(m-1)} - d\left(t_{i_m}, t_{NN}^{(m-1)}\right)\right), d\left(t_{i_m}, t_{NN}^{(m-1)}\right)\right\} \leq 2 \cdot d_{\min}^{(m-1)}$, akkor $\mathcal{P}_m = \mathcal{P}_{m-1} \cup \{t_{i_m}\}$, $t_{FN}^{(m)} = \arg \max_{u \in \mathcal{P}_m} d(x, u)$, $t_{NN}^{(m)} = \arg \min_{u \in \mathcal{P}_m} d(x, u)$, $d_{\max}^{(m)} = d\left(x, t_{FN}^{(m)}\right)$, $d_{min}^{(m)} = d\left(x, t_{NN}^{(m)}\right)$, és a következő, $m+1$-dik kereső lépésnél folytatjuk, stb.

2.7. Irodalmi példák kizárásos keresési stratégiákra

Ebben a pontban három korábban publikált $\mathcal{S}_e$ típusú kereső algoritmust ismertetünk.

Keresőfás algoritmus kizárással. Az NN-kereső algoritmusok egy része keresőfát alkalmaz (pl. FUKUNAGA és NARENDA [15], KIM és PARK [23].). A $\mathcal{T}_n$ tanulópont-halmaz pontjainak megfelelő csoportosításával, majd a csoportok tovább bontásával egy hierarchiát definiálnak, ami alapja lesz a keresésnek. A keresés folyamán egyre élesedő kizárási feltételek révén a keresőfa egyre több ága zárható ki a további keresésből. Ebben a pontban FUKUNAGA és NARENDA algoritmusának kiterjesztését adjuk meg tetszőleges metrikus térre. Az eredeti algoritmusban a csoportok átlagait a csoportok centrumelemével fogjuk helyettesíteni. A módosított algoritmus így csak az első stációjában fog eltérni az eredetitől.

Adott egy $\mathcal{T}_n = \{t_1, t_2, ..., t_n\} \subset \mathcal{X}$ tanulópont-halmaz az $(\mathcal{X}, d)$ metrikus térben. Bontsuk fel $\mathcal{T}_n$-t l lehetőleg egyenlő elemszámú diszjunkt részhalmazra valamilyen klaszterező eljárással. Jelölje $G_1, G_2, ..., G_l$ ezeket a klasztereket. Ez a partíció alkotja az első generációt. Ezután mindegyik G_i klasztert újból l egyenlő elemszámú klaszterre bontjuk, létrehozva a második generáció partícióit: $G_{11}, G_{12,}..., G_{1l}, G_{21}, ..., G_{2l}, ..., G_{ll}$. A szétbontást addig folytatjuk, amíg az utolsó generáció partíciói mind egyelemű halmazok nem lesznek. Az utolsó generáció sorszámát $g_{\max}$ jelöli. (A tananyag mintaelemszámától függően az utolsó generációban nem biztos, hogy l lesz a partíciók száma, de ez az algoritmus működését nem fogja befolyásolni.) Minden generáció partíciójának minden halmazát egyértelműen azonosítani tudjuk egy $p = (\alpha_1, \alpha_2, ..., \alpha_k)$ indexsorozattal, ahol az α_1 index adja meg az első generáció megfelelő klaszterének sorszámát, α_2 a G_{α_1} halmaz továbbosztásakor a megfelelő részhalmaz sorszámát jelenti, stb. Ezután minden generáció minden G_p klasztere esetén meghatározzuk a halmaz c_p centrumelemét és az R_p takaró sugarát.

Egyszerűen belátható az alábbi két állítás.

2.1. segédtétel. *A $t_i \in G_p$ tanulópont nem lehet az x tesztpont legközelebbi társa, ha $d_{\min} + R_p < d(x, c_p)$, ahol $d_{\min}$ az eddig talált legkisebb távolság x és a tanulópontok között.*

Bizonyítás. $d(x, t_i) \geq d(x, c_p) - d(t_i, c_p) \geq d(x, c_p) - R_p > d_{\min}$.■

2.2. segédtétel. *A $t_i \in G_p$ tanulópont nem lehet az x tesztpont legközelebbi társa, ha $d_{\min} + d(t_i, c_p) < d(x, c_p)$.*

Bizonyítás. Hasonlóan. ■

A *Fukunaga-Narenda* algoritmus második stációja változatlan.

0^o *(Inicializálás)* Legyen a pillanatnyi minimális távolság $d_{\min} := \infty$, a pillanatnyi generáció szintszáma $g := 1$, a feldolgozandó klaszterek indexeinek listája $\Lambda := \emptyset$.

1^o *(A feldolgozandó partíciók beállítása)* Eltároljuk a g generáció klasztereinek sorszámát a Λ listába, és kiszámoljuk az ehhez a listához tartozó p indexhez a $d(x, c_p)$ távolságokat.

2^o *(Az 2.1 segédtételben megfogalmazott kizárási elv érvényesítése)* Vegyük sorra a $p \in \Lambda$ indexeket, és ezek közül töröljük azokat, ahol $d_{\min} + R_p < d(x, c_p)$ teljesül.

3^o *(Visszaléptetés)* Ha Λ kiürül, akkor visszalépünk egy korábbi szintre, vagyis $g := g - 1$. Ha $g = 0$ lenne, akkor megállítjuk az algoritmust. $d_{\min}$ tartalmazza a legközelebbi távolságot, NN pedig a legközelebbi szomszéd indexét. Ha viszont $g \neq 1$, akkor újra a 2^o lépésre térünk. Ha $\Lambda = \emptyset$ akkor a 4^o lépésre ugrunk.

4^o *(A legközelebbi klaszter kiválasztása az adott generációban)* Legyen $p^* \in \Lambda$ az a klaszterindex, melynél $d(x, c_{p^*})$ minimális. Legyen $q := p^*, \Lambda := \Lambda \setminus \{p^*\}$. Ha $g = g_{\max}$, akkor lépjünk 5^o-re, különben $g := g + 1$, és lépjünk 1^o-re.

5^o *(A 2.2 segédtételben megfogalmazott kizárási elv érvényesítése)* Feldolgozzuk a G_q klasztert. Csak azoknál a $t_i \in G_q$ tanulópontoknál számoljuk ki a $d(x, t_i)$ távolságokat, melyekre $d_{\min} + d(t_i, c_q) \geq d(x, c_q)$. Ilyenkor ha még az is teljesül, hogy $d(x, t_i) < d_{\min}$, akkor $d_{\min} := d(x, t_i)$ és $NN := i$. Ha már minden tanulópontot leteszteltünk G_q-ból, akkor a 2^o lépésre térünk vissza.

2.5. megjegyzés. *Gyorsítási lehetőségek az osztályozásnál*

1. Tegyük fel, hogy két osztályunk van C_1 és C_2, $C_1 \cap C_2 = \emptyset$, $C_1 \cup C_2 = \mathcal{T}_n$. Ha az első generáció felbontását úgy végezzük el, hogy $G_1, G_2, ..., G_\alpha \subset C_1$ és $G_{\alpha+1}, G_{\alpha+2}, ..., G_l \subset C_2$ legyen, akkor lehetőség nyílik arra, hogy az algoritmust hamarabb megállítsuk. Ugyanis, ha $g = 1$ esetben pl. $d_{\min} + R_p < d(x, c_p)$ fenáll minden $p > \alpha$ esetén és $NN \leq \alpha$, akkor már megállhatunk. A megállításkor NN

még nem biztos, hogy a legközelebbi szomszéd indexét tartalmazza, de a tartalmazott tanulópont osztálya garantáltan megegyezik a legközelebbi társ osztályával, vagyis x-et pontosan tudjuk már osztályozni.

2. A 2.2 segédtételben megfogalmazott kizárási feltételt élesíteni lehet az 5^o lépésben. A $t_i \in G_q$ kizárandó, ha $d_{\min} < \max |d(x,u) - d(t_i,u)|$ teljesül, ahol a maximumot azon u tanulópontokra kell venni, melyeknél már kiszámoltuk a $d(x,u)$ távolságokat.

A szeparált halmazok felhasználása a keresésnél. A 2.6.-2.8. tételekben megfogalmazott csoportos kizárási feltételeket az alábbiak szerint használhatjuk fel a legközelebbi szomszéd megkeresésére. $\mathcal{T}_n$-et három részre bontjuk fel: az $A_1 (\subset C_1), A_2 (\subset C_2)$ szeparált halmazokra, és az $E = \mathcal{T}_n \setminus (A_1 \cup A_2)$ halmazra. Az A_1 és A_2 halmazokat a 2.4. szakaszban leírt algoritmusok valamelyikével állítottuk elő.

- Először a $d(x,c_1)$ és $d(x,c_2)$ távolságokat számoljuk ki. $d_{\min} := \min\{d(x,c_1), d(x,c_2)\}$ és $class(x) := 1$, ha $d(x,c_1) < d(x,c_2)$, különben $class(x) := 2$. Megnézzük, hogy teljesül-e a 2.6. vagy a 2.11. tételben megfogalmazott kizárási feltétel. Ha igen, A_1 és A_2 közül a megfelelőt töröljük a további keresésből.

- Az E halmaz elemei között a $\mathcal{K}_3$ kizárás figyelembevételével a minimax stratégiával kiválasztjuk a következő tanulópontot. Ha szükséges, $d_{\min}$ és $class(x)$ tartalmát felülírjuk. Ilyenkor megvizsgáljuk a 2.8. tételben ismertetett kizárási feltételt a még nem törölt A_i halmaz esetén. A feltétel teljesülése esetén töröljük a megvizsgált halmazt a további keresésből.

- A nem kizárt A_1 (vagy A_2) halmazok tanulópontjait csak akkor vesszük sorra, ha már E elemei elfogytak. Ezek elemei között is a minimax kereséssel és a $\mathcal{K}_3$ kizárással haladunk.

Keresés cellák csoportos kizárásával. Az $\mathcal{X} = \mathbb{R}^p$ speciális esetben a d Euklideszi metrikával alkalmazható, a csoportos kizárás elvén alapuló algoritmust közölnek DJOUADI és BOUKTACHE [5].

Tegyük fel, hogy $\mathcal{T}_n \subset H$, ahol $H \subset \mathbb{R}^p$ a lehető legkisebb oldalhosszú p-dimenziós hiperkocka. Jelölje h ezen hiperkocka oldalának hosszát! Vezessük be az alábbi jelöléseket: $\mathcal{T}_n = \{\underline{t}_1, \underline{t}_2, ..., \underline{t}_n\}, \underline{t}_i = \left(t_1^{(i)}, t_2^{(i)}, ..., t_p^{(i)}\right)^T$, $\underline{v} =$

$(v_1, v_2, ..., v_p)^T, v_i = \min\limits_{j=1,2,...,n} t_j^{(i)}$, $\underline{u} = (u_1, u_2, ..., u_p)^T, u_i = \max\limits_{j=1,2,...,n} t_j^{(i)}$. Ekkor $h = \max\limits_{i=1,...,p} (u_i - v_i)$.

Hajtsuk végre az $l(\underline{x}) = \frac{1}{h}(\underline{x} - \underline{v})$ lineáris transzformációt $\mathcal{T}_n$ minden elemére! Ezzel mindegyik tanulópontot az egységnyi hiperkockába transzformáltuk. Legyen $\mathcal{T}_n^* = \{\underline{t}_i^* = l(\underline{t}_i), i = 1, 2, ..., n\}$. Osszuk fel az egység hiperkockát $N = k^p$ egybevágó G_i kiskockára (cellára). Jelölje $\underline{g}_i$ közülük a G_i szimmetria-középpontját, $n_i = \#G_i \cap \mathcal{T}_n^*$ pedig az i-edik cellába eső transzformált tanulópontok számát. Legyen továbbá $R_i = \max\limits_{t \in G_i \cap \mathcal{T}_n^*} d\left(\underline{g}_i, \underline{t}\right)$ az i-edik cella tanulópontjait lefedő, $\underline{g}_i$ középpontú gömb sugara.

Egy tetszőleges $\underline{x} \in \mathbb{R}^p$ vektor legközelebbi társát ezek után az alábbi algoritmussal keressük meg.

1^o Legyen $\underline{x}^* = l(\underline{x})$, és határozzuk meg a $D_i = d\left(\underline{g}_i, \underline{x}^*\right)$, $D_{\min} = \min\limits_{i=1,...,N} D_i, i^* = \arg\min\limits_{i=1,...,N} D_i$ mennyiségeket! $(D_{\min} = D_{i^*})$.

2^o Keressük meg $\underline{x}^*$ legközelebbi társát a G_i cellában! Jelölje $d_{\min}$ a minimális távolságot!
- Ha $D_{\min} \leq \frac{\sqrt{p}}{2k}$, azaz $\underline{x}^* \in G_i$, akkor megtaláltuk a legközelebbi társat, STOP.
- Ha $D_{\min} > \frac{\sqrt{p}}{2k}$, akkor még venni kell azon G_j cellában található tanulópontokat is, ahol $D_j < d_{\min} + R_i$ áll fenn. (Viszont minden más cella összes tanulópontját kizárhatjuk a keresésből!) STOP.

2.6. megjegyzés. *A leírásban szereplő k természetes szám azt adta meg, hány részre daraboljuk a hiperkocka oldalait. Ezt az értéket úgy célszerű megválasztani, hogy az $N + \max\limits_{i=1,...,N} n_i$ kifejezés minimális legyen. Ugyanis a fenti kifejezés felső becslést ad a keresés maximális számára, ha az osztályozandó pont benne van H-ban.*

Keresés horoghalmazokkal. A minimax keresés módosítását javasolják LEE és CHAE [24], ún. „horog" (*anchelor*) halmaz konstruálásával. Algoritmusukat *RCNN*-nek (*R*educed-*C*omplexity *N*earest *N*eighbor Classifier) nevezték el. A $\mathcal{T}_n$ tanulópont-halmazt két részre javasolják felosztani: egy m elemszámú A_m horog-halmazra, és a B_{n-m} komplementer halmazra. Egy $x \in \mathcal{X}$ pont legközelebbi társának megkeresése ezek után az alábbi algoritmus alapján történik:

1^o Kiszámítjuk a $d(x,a), a \in A_m$ távolságokat, és ezek segítségével beállítjuk a $d_{\min} := \min_{a \in A_m} d(x,a)$ és $x_{\min} := \arg\min_{a \in A_m} d(x,a) = NN(x, A_m)$ rekeszek tartalmát. ($NN(x, A_m)$ jelöli az x legközelebbi társát az A_m-ben.)

2^o A $b \in B_{n-m}$ tanulópontok közül csak azoknak a távolságát számoljuk ki x-től, melyekre $d^{A_m}_{\mathrm{LOW}}(x,b) = \max_{a \in A_m} |d(x,a) - d(b,a)| < d_{\min}$ teljesül. Jelölje B^* az ilyen tanulópontok halmazát!

3^o Végül $d_{\min} := \min\left\{d_{\min}, \min_{b \in B^*} d(x,a)\right\}, x_{\min} := NN(x, A_m \cup B^*)$.

A keresés során az A_m horog-halmaz elemeitől minden $x \in \mathcal{X}$ esetén kiszámoljuk a távolságokat, míg a B_{n-m} halmaz tanulópontjaitól csak a 2^o pontban látható feltétel teljesülése esetén. „Jó" horog-halmaz esetén ezek száma kevés lesz. Az A_m horog-halmaz megkonstruálására Lee és Chae az alábbi algoritmust javasolják.

1^o $\mathcal{T}_n$ tanulópontjait sorbarendezzük az alább leírt módon:
- A sorban az első, a_1-gyel jelölt tanulópont legyen az $a_1 = \arg\max_{t \in \mathcal{T}_n} d(t, NN(t, \mathcal{T}_n \setminus \{t\}))$.
- Tegyük fel, hogy már előállítottuk az $a_1, a_2, ..., a_k$ sorrendet. A következő a sorban az az $a_{k+1} \in \mathcal{T}_n \setminus \{a_1, a_2, ..., a_k\}$ tanulópont lesz, melyre $a_{k+1} = \arg\max_{t \in \mathcal{T}_n \setminus \{a_1, a_2, ..., a_k\}} P_e(t)$ teljesül, ahol

$$P_e(t) = \sum_{u \in \mathcal{T}_n \setminus \{a_1, a_2, ..., a_k, t\}} \Big(d^{\{a_1, a_2, ..., a_k, t\}}_{\mathrm{LOW}}(u, NN(u, \mathcal{T}_n \setminus \{u\})) - d^{\{a_1, a_2, ..., a_k\}}_{\mathrm{LOW}}(u, NN(u, \mathcal{T}_n \setminus \{u\})),$$

ahol $d^A_{\mathrm{LOW}}(u,t) = \max_{a \in A} |d(u,a) - d(t,a)|$.

2^o Ezután a horog-halmaz elemszámának, m-nek a meghatározása kísérleti úton történik. Egy lehetséges megoldás az lehet, ha felhasználunk egy $U_s = \{u_1, u_2, ..., u_s\}$ tesztponthalmazt. Az $A_1 = \{a_1\}, A_2 = \{a_1, a_2\}, ..., A_m = \{a_1, a_2, ..., a_m\}, ...$ horoghalmazok mindegyikével elvégezzük az *RCNN* osztályozást az U_s halmaz összes elemére. Figyeljük, hogy melyik horog-halmaz esetén kell átlagosan legkevesebb távolságot számítani, és azt választjuk ki véglegesnek. Tehát, ha az A_i horog-halmaz esetén az $u_j \in U_s$ tesztpont meghatározásához $\alpha_j\ (\geq i)$ távolságszámításra volt szükség, akkor legyen $\beta_i = \frac{1}{s} \sum_{j=1}^{s} \alpha_j$. Azt a A_i horoghalmazt választjuk, ahol β_i minimális!

2.7. megjegyzés. *1. A horoghalmaz előállításának első szakaszában, a tanulópontok sorbarendezésekor a megmaradó tanulópontok közül mindig azt veszszük sorra, amely leginkább növeli a döntési halmazban nem szereplő tanulópontok d^A_{LOW} értékeinek összegét. Mivel a tanulópontok kizárása a d^A_{LOW} értékek alapján történik, ezért mindig azzal a tanulóponttal folytajuk a sorrendet (bővítjük a horog-halmazt), amely potenciálisan leginkább diszkriminálja a megmaradó tanulópontokat.*

2. Lee *és* Chae *[24] kézzel írott számjegyek alakzatai esetén végezte el az m elemszám kiválasztását, kihasználva az alkalmazás specialitásait. Futtatásaik során az m elemszám és az f számítási idő között egy racionális törtfüggvény jellegű összefüggést véltek felfedezni. Ezt a kapcsolatot felhasználva az m meghatározásának a folyamata jelentősen lerövidíthető volt. Az $n = 1000$ elemszámú, 24×24-es felbontású (576 dimenziós) tananyaghoz, melynek elemei összesen $L = 80$ osztályba lettek besorolva (mindegyik számjegy 8 alosztályra bomlott), végül is $m = 21$ elemszámú horog-halmazt találtak optimálisnak. Ezt alkalmazva az RCNN osztályozás a feldolgozási időt 38%-kal csökkentette a hagyományos NN-algoritmushoz képest.*

2.8. A kizárásos keresési stratégiák hatékonyságának jellemzése

Adott egy $\mathcal{T}_n$ tanulópont halmaz és egy x tesztpont. Jelölje $pow(x)$ azon tanulópontok minimális számát $\mathcal{T}_n$-ből, melyek x-től vett távolságai ismeretében a többi tanulópont kizárható a további keresésből a $\mathcal{K}$ kizárási feltétellel. Tehát, ha $pow(x) = k$, akkor léteznek olyan $t_{i_1}, t_{i_2}, ..., t_{i_k} \in \mathcal{T}_n$ tanulópontok, hogy a $d\left(x, t_{i_j}\right)$ távolságok és a $\underline{\underline{D}} = \left(d\left(t_\alpha, t_\beta\right)\right)_{\alpha,\beta=1,2,...,n}$ távolságmátrix ismeretében a $\mathcal{K}$ kizárási feltétellel minden $u \in \mathcal{T}_n \setminus \{t_{i_1}, t_{i_2}, ..., t_{i_k}\}$ tanulópontról biztosan állítható, hogy nem lehetnek x legközelebbi szomszédja. Másrészt az is igaz, hogy tetszőleges $t_{\alpha_1}, t_{\alpha_2}, ..., t_{\alpha_l} \in \mathcal{T}_n$ $l \leq k-1$ részhalmaz esetén létezik olyan $u \in \mathcal{T}_n$, ami nem zárható ki $\mathcal{K}$-val. $pow(x)$ tehát a minimális kizáró tanulópont halmaz elemszámát jelenti. Ennél kevesebb távolságszámítással nem lehet megúszni a legközelebbi társ megtalálását az alkalmazott kizárási elv mellett. $pow(x)$ az x tesztpont *ereje*, $\mathcal{J}_0(x) = \{t_{i_1}, t_{i_2}, ..., t_{i_k}\}$ pedig egy *minimális kizáró halmaz*. Tegyük fel, hogy az $\mathcal{S}$ keresési stratégiát alkalmazzuk a $\mathcal{K}$ kizárási feltétellel abban a legrosszabb esetben, amikor az elsőnek kiválasztott tanulópont éppen az x tesztpont legtávolabbi társa $\mathcal{T}_n$-ben. Jelölje $step(x)$ azon $d\left(x, t_{i_j}\right)$ távolságszámítások számát, ami ahhoz szükséges, hogy biztosan állíthassuk azt, hogy már megtaláltuk a legközelebbi társat. Nyilván $step(x) \geq pow(x)$. $step(x)$ az $\mathcal{S}$ keresési stratégia *lépésszáma*, $eff(x) = \frac{pow(x)}{step(x)}$ pedig a *hatékonysága*.

Végezzük el az $x \in \mathcal{X}$ ponthoz az NN-keresést a $\mathcal{K}$ kizárási elvvel. Az m-edik tanulópont feldolgozása után jelöljük $\mathcal{P}_m$-mel a már feldolgozott tanulópontok halmazát, és $\mathcal{E}_m$-mel azon $\mathcal{T}_n \setminus \mathcal{P}_m$ pontok halmazát, melyek a $\mathcal{K}$-val kizárhatók ezek után a keresésből. $\mathcal{P}_m$ az m-edik lépésbeli *janicsárhalmaz*, $\mathcal{E}_m$ pedig a *hulladékhalmaz*.

A $\mathcal{K}$ kizárási elv segítségével a $\mathcal{T}_n$ tananyaghoz egy $\mathcal{G}$ irányított gráf rendelhető, melyenek a tanulópontok lesznek a csúcsai, és az $u, v \in \mathcal{T}_n$ tanulópontok között akkor fut egy irányított él u-ból v-be, ha a $\mathcal{K}$ kizárási elvvel kizárja u v-t. Vagyis, ha u belekerül a janicsárhalmazba, v biztosan a hulladékhalmazba kerül.

2.4. definíció. *Azt mondjuk, hogy az $u \in \mathcal{T}_n$ tanulópont a kizárja $v \in \mathcal{T}_n$-t az $\mathcal{S}_e$ kizárásos kereső algoritmusnál, ha $u \in \mathcal{P}_m$-ből következik $v \in \mathcal{E}_m$. Jelölés: $u \rightarrowtail v$.*

2.11. tétel. *A $\mathcal{K}_1$ vagy $\mathcal{K}_2$ kizárásnál, ha $u \rightarrowtail v$ és $v \rightarrowtail w$, akkor $w \not\rightarrowtail u$.*

Bizonyítás. A 2.4 szerint, ha $u \rightarrowtail v$, azaz $2d(x,u) < d(u,v)$ és $v \rightarrowtail w$, azaz $2d(x,v) < d(w,v)$, akkor $v \not\rightarrowtail u$, azaz $2d(x,v) \geq d(u,v)$ és $w \not\rightarrowtail v$, azaz $2d(x,w) \geq d(w,v)$. Tehát $2d(x,u) < d(u,v) \leq 2d(x,v) < d(w,v) \leq 2d(x,w)$. Ekkor $d(u,w) \leq d(u,x) + d(w,x) \leq 2d(w,x)$ is, azaz w nem zárhatja ki u-t. ■

2.5. definíció. *A $\mathcal{G}$ gráfot a $\mathcal{K}$ kizárási elv (az $\mathcal{S}$ kereső stratégia) kizárási gráfjának nevezzük.*

2.12. tétel. *A $\mathcal{K}_1$ és $\mathcal{K}_2$ kizárási gráfjai nem tartalmazhatnak irányított kört.*

Bizonyítás. Tegyük fel, hogy $t_1 \rightarrowtail t_2 \rightarrowtail \cdots \rightarrowtail t_k$, azaz $2d(x,t_1) < d(t_1,t_2) \leq 2d(x,t_2) < d(t_2,t_3) \leq \cdots \leq 2d(x,t_{k-1}) < d(t_{k-1},t_k)$. Tehát $d(t_k,t_i) \leq d(t_k,x) + d(t_{k-1},x) \leq 2d(t_k,x)$ miatt t_k egyetlen t_i-t sem zár ki. $(i < k)$. ■

Egy $\mathcal{S}_e$ kizárásos kereső stratégia hatékonyságának jellemzéséhez szükségünk van a $pow(x)$ kiszámítását végző algoritmusra. Ehhez elő kell állítani a kereső stratégia minimális elemszámú $\mathcal{J}_0$ janicsárhalmazát, ahol $\mathcal{J}_0 \cup \mathcal{E}_m = \mathcal{T}_n$. A feladat megegyezik a gráfelméletben ismert minimális dominál csúcshalmaz (*dominating set*) keresésének problémájával. Egy adott irányított $\mathcal{G} = (V, E)$ gráfban előállítandó az a minimális elemszámú $D \subset V$ csúcshalmaz, melynek pontjaiból minden $V \setminus D$ pontba fut él. A D megadásának problémája NP-teljes. Az algoritmushoz felhasználjuk a $\mathcal{G}$ gráf $\underline{\underline{A}} = (a_{ij})$ szomszédsági mátrixát, ahol

$$a_{ij} = \begin{cases} 1 & \text{ha } t_i \rightarrowtail t_j \\ 0 & \text{egyébként.} \end{cases}$$

A továbbiakban csak a legerősebb $\mathcal{K}_3$ kizárással foglalkozunk. A 2.8.1. szakaszban néhány az algoritmus leírásához szükséges fogalmat vezetünk be. Maga az algoritmus a 2.8.2. szakaszban olvasható. A 2.8.3. szakaszban a számítógépes kísérletekről számolunk be. A 2.8.4. szakaszban azt a kérdést tárgyaljuk, milyen körülmények között célszerű a kizárásos NN-keresést alkalmazni.

2.8.1. A $\mathcal{K}_3$ kizárási elv janicsárhalmazai

A $\mathcal{K}_3$ kizárási elv segítségével egy konkrét kizárásos NN-keresés esetén értelmezhetjük a *janicsárhalmaz* fogalmát. A janicsárhalmaz lényegében a tananyag minden olyan részhalmaza, melyhez tartozó tanulópontok és az x osztályozandó pont távolságainak ismeretében a maradék tanulópontokról biztosan állíthatók, hogy nem lehet közöttük az x legközelebbi társa. Másképpen megfogalmazva: a $\mathcal{J}$ janicsárhalmaz pontjait figyelembevéve érvényes a $\mathcal{K}_3$ kizárási elv megállítási szabálya, azaz

$$\min_{u\in\mathcal{J}} d(x,u) \leq \min_{t\in\mathcal{T}_n\setminus\mathcal{J}} \max_{u\in\mathcal{J}} |d(x,u) - d(u,t)|$$

Viszont, ha a janicsárhalmazból bármely pontot elhagyjuk, már nem lesz érvényes a fenti megállítási feltétel. Nyílvánvaló, hogy a janicsárhalmazok nem egyértelműen állíthatók elő, de néhány tulajdonság rögtön látható. Az összes janicsárhalmaz metszete nem üres halmaz, mert az x legközelebbi társa biztosan benne van. A metszetet *janicsárhalmaz-magnak* fogjuk nevezni, ami elvileg nem féltétlenül egyelemű halmaz. Az $\mathcal{K}_3$ kizárásos NN-kereső algoritmusok hatékonyságának empirikus ellenőrzéséhez szükségünk van a minimális elemszámú janicsárhalmaz előállítására, melyet *optimális janicsárhalmaznak* fogunk nevezni. Nem lehet kevesebb távolságszámítással az adott beállítások mellett eljutni az x legközelebbi társához, mint ahány az optimális janicsárhalmaz elemszáma. A próbafuttatások során különböző beállítások mellett meghatározzuk a kizárásos NN-keresés lépésszámát (a megállítási feltétel teljesüléséig elvégzendő távolságszámítások számát), és ezt vetjük össze az optimális janicsárhalmaz elemszámával. A pontos definíciókat ebben a pontban fogjuk megadni.

2.6. definíció. *Legyen $A \subset \mathcal{T}_n$. Azon $u \in \mathcal{T}_n \setminus A$ pontok száma, amelynél $\min_{a\in A} d(a,x) \leq \max_{a\in A} |d(a,x) - d(a,u)|$ teljesül, az A halmaz kizáró ereje. Jelölés: $pow(A)$.*

2.7. definíció. *Az $A \subset \mathcal{T}_n$ halmaz kizáró halmaz, ha $\#A + pow(A) = n$ azaz, ha az A-hoz tartozó tanulópontok segítségével az összes többi tanulópont kizárható az NN-keresésből. A $\mathcal{J}$ kizáróhalmaz janicsárhalmaz, ha nincs kizáró részhalmaza. A janicsárhalmaz optimális, ha nincs nála kisebb elemszámú janicsárhalmaz.*

2.8. definíció. *A t tanulópont kizáró ereje azon $u\,(\neq t)$ tanulópontok száma, ahol $d_{\min} \leq |d(t,x) - d(t,u)|$. Jelölés: $pow(t)$.*

2.9. definíció. *A $t \notin A$ tanulópont A halmazra vett relatív kizáró ereje azon $u \in \mathcal{T}_n \setminus (A \cup \{t\})$ tanulópontok száma, ahol $d_{\min} \leq |d(t,x) - d(t,u)|$ teljesül, de $d_{\min} > \max\limits_{a \in A} |d(a,x) - d(a,u)|$. Jelölés: $pow(t \mid A)$.*

2.10. definíció. *Azon tanulópontok $\mathcal{J}^*$ halmazát, melyek elemeit semmilyen más tanulóponttal sem lehet kizárni, janicsárhalmaz-magnak nevezzük.*

2.11. definíció. *Az u tanulópont nem erősebb v-nél (v nem gyengébb u-nál), ha minden $t\,(\neq u, v)$ esetén melyre $d_{\min} \leq |d(u,x) - d(t,u)|$ teljesül $d_{\min} \leq |d(v,x) - d(t,v)|$ is fennáll. Másképpen: u tanulópont nem erősebb v-nél ha azokat a pontokat, amiket u kizár, v is kizárja.*

2.8. megjegyzés. - *A legközelebbi társ(ak) mindig elemei $\mathcal{J}^*$-nak.*

- *$\mathcal{J}^*$ minden janicsárhalmaznak részhalmaza.*

2.8.2. Optimális janicsárhalmazt előállító algoritmus

Az alábbi algoritmus adott $\mathcal{T}_n$ tananyaghoz, adott x osztályozandó ponthoz a d metrika esetén megkonstruál egy optimális janicsárhalmazt. A tanulópontok egymástól vett távolságainak mátrixát, és az osztályozandó pontnak a tanulópontoktól vett távolságait felhasználjuk a konstrukció során.

Az algoritmus főbb lépései:

0^o Az $\underline{\underline{A}}$ bináris relációmátrix elkészítése, melynek (i, j) komponense pontosan akkor 1, ha az $d(x, t_i)$ és $d_{\min}$ ismeretében biztosan állítható, hogy t_j nem lehet a legközelebbi társ. Tehát $\underline{\underline{A}}$ i-edik sorában ott áll 1-es, amelyeket t_i képes kizárni a további keresésből. Ennek a mátrixnak a feldolgozása útján állítunk elő egy lehetséges optimális janicsárhalmazt.

1^o A $\mathcal{J}^*$ janicsárhalmaz-mag előállítása. Közben megjelöljük $\underline{\underline{A}}$ azon sorait, melyek $\mathcal{J}^*$ elemeihez tartoznak.

2^o Megjelöljük $\underline{\underline{A}}$ azon oszlopait, melyek olyan tanulóponthoz tartoznak, melyet $\mathcal{J}^*$ pontjai kizárnak.

3^o Megjelöljük $\underline{\underline{A}}$ azon sorait, melyek olyan tanulópontokhoz tartoznak, melyeknél van nem gyengébb tanulópont. (L. 2.11. definíciót!)

4^o Mohó algoritmussal megkonstruálunk egy lehetséges $\mathcal{J}$ janicsárhalmazt. (Nem biztos, hogy optimális lesz.)

5^o Ellenőrizzük, hogy van-e $\mathcal{J}$-nél kisebb elemszámú janicsárhalmaz. (Sorraveszszük a ritkított $\underline{\underline{A}}$ mátrix soraihoz tartozó tanulópontok összes részhalmazait az elemszám növekedése szerint.)

Az algoritmus leírásakor alkalmazott jelölések:

$R := value$ az R rekeszbe töltjük a *value* értéket.

$T(i) := d(x, t_i), i = 1, 2, ..., n$ a tananyag pontjainak az x-től vett távolságait tároló tömb.

$D(i, j) := d(t_i, t_j), i, j = 1, 2, ..., n$ a tanulópontok egymástól vett távolságait tároló kétdimenziós tömb.

t_{NN} az a tanulópont, amelyik x-hez legközelebb van a d metrika szerint (a legközelebbi társ.)

$A(i, j)$ bináris relációmátrix. Értéke 1, ha t_j kizárható a keresésből amennyiben t_{NN} és t_i a janicsárhalmazban van. Különben értéke 0.

$d_{\min} = d(x, t_{NN})$ jelöli a minimális távolságot.

$T_{\min}$ a keresés folyamatában pillanatnyilag legkisebb távolság.

Q a $\mathcal{J}^*$ halmaz elemeinek a száma.

$SZAM$ a $\mathcal{J}^*$ halmaz által kizárt tanulópontok száma.

M a $\mathcal{J}^*$-ot optimális janicsárhalmazzá kibővítő pontok minimális száma.

$DISTANCE(i)$ a $d(x, t_i)$ távolságot kiszámoló eljárásfüggvény.

$i = KEZD...VEG(LSZ) : [_i \cdots]_i$ a $[_i$,$]_i$ zárojelek között olvasható utasítássorozathoz szervezett ciklus.

IF $feltétel$ **THEN** [$parancssorozat$] (**ELSE** [$parancssorozat$]) feltétel szerinti elágazás.

GOTO $label$

$\vdots$

$label$: [$utasítássorozat$] Vezérlésátadás a label címkével jelölt utasítássorozatra.

Az egy sorba leírt utasításokat ; jellel választjuk szét.

Az algoritmus:

1^o A $T, d_{\min}, \underline{\underline{A}}$ számolása:

$d_{\min} := \infty$
$i = 1...n(1)$:
$[_i T(i) := DISTANCE(i)$
$d_{\min} := \min(d_{\min}, T(i))]_i$
$i = 1..n(1)[_i j = 1...n(1)$
$[_j A(i,j) := 0$
IF $|T(i) - D(i,j)| \geq d_{\min}$ **THEN** $A(i,j) := 1]_j]_i$

2^o $\mathcal{J}^*$ előállítása

$Q := 0$
$j = 1...n(1)$
$[_j SUM := 0$
$i = 1...n(1)[_i SUM := SUM + A(i,j)]_i$
IF $SUM = 0$ **THEN** $[SOR(j) := -n - SOR(J); Q := Q + 1]]_j$

$3^o a.$) Az $\underline{\underline{A}}$ relációmátrix oszlopainak ritkítása a $\mathcal{J}^*$ pontjai által kizárt pontokkal

$SZAM := 0$
$i = 1...n(1)$
$[_i j = 1...n(1)$
$[_j OSZL(j) := j$
IF $(SOR(i) < 0) \wedge (A(i,j) = 1) \wedge (OSZL(j) > 0)$ **THEN**
$[OSZL(j) := -n - OSZL(j); SZAM := SZAM + 1]]_j]_i$

3^o *b.*) Az $\underline{\underline{A}}$ mátrix sorösszegeinek számolása

A t_i tanulópontok mindegyikéhez kiszámoljuk a $pow(t_i \mid \mathcal{J}^*)$ értéket, azaz azt a számot, ahány tanulópontot kizárnak a $\mathcal{J}^*$ pontjai által még ki nem zárt pontok közül. A $t_i \notin \mathcal{J}^*$ tanulópont esetén ezt az adatot a $CSUM(i)$ rekesz fogja tárolni.

$i = 1...n(1)$
$[_i SUM := 0$
$j = 1...n(1)$
$[_j$**IF** $(SOR(i) > 0) \wedge (OSZL(j) > 0)$ **THEN** $SUM := SUM + A(i,j)]_j$
$CSUM(i) := SUM]_i$

3^o *c.*) Az $\underline{\underline{A}}$ mátrix sorainak átrendezése a sorösszegek szerint

$i = 2...n-1(+1)$
$[_i j = n...i(-1)$
$[_j$**IF** $CSUM(j) < CSUM(j-1)$ **THEN**
$[S1 := SOR(j); SOR(j) := SOR(j-1); SOR(j-1) := S1;$
$S1 := CSUM(j); CSUM(j) := CSUM(j-1); CSUM(j-1) := S1\ ]]_j]_i$

3^o *d.*) $\underline{\underline{A}}$ sorainak további ritkítása

$\underline{\underline{A}}$-ból elhagyjuk azoknak a tanulópontoknak megfelelő sorokat, melynél van erősebb tanulópont. Tehát az u tanulópontnak megfelelő sort akkor hagyjuk el, ha van olyan v tanulópont, ami kizárja azokat a tanulópontokat, melyeket u is kizárna. M tartalmazza a megtartott sorok számát.

$i = 1...n-1(1)$
$[_i j = i+1...n(1)$
$[_j$**IF** $SOR(i) < i$ **THEN GOTO** *label*1
IF $SOR(j) < 0$ **THEN GOTO** *label*2
$SUM := 0$
$l = 1...n(1)$

$[_l$ **IF** $OSZL(l) > 0$ **THEN** $[S1 := A(j,l) - A(i,l)$; **IF** $S1 = 1$ **THEN** $SUM := 1]_l$
IF $SUM = 0$ **THEN** $SOR(j) := -SOR(j)$
$label2 : S1 := 1]_j$
$label1 : S1 := 1]_i$
$M := 0$
$i = 1...n(1)[_i$**IF** $SOR(i) > 0$ **THEN** $M := M + 1]_i$

4^o Egy lehetséges kizáróhalmaz előállítása, ha $M > 0$.

$SM := 1$
$i = 1...M(1)[_i SORJEL(i) := SOR(i)]_i$
$SORJEL(SM) := -1$
$j = 1...n(1)[_j OSZJEL(j) := OSZL(j)]_j$
$SUM := 0$
$j = 1...n(1)$
$[_j$**IF** $(OSZL(j) > 0) \wedge (A(SOR(1), j) = 1)$
THEN $[OSZJEL(j) := -1; SUM := SUM + 1]]_j$
$label1 : S1 := 1$
$i = 1...M(1)$
$[_i SEGED(i) := 0$
IF $SORJEL(i) > 0$ **THEN**
$[j = 1...n(1)[_j$ **IF** $(OSZJEL(j) > 0) \wedge (A(SOR(i), j) = 1)$
THEN $[SEGED(i) := SEGED(i) + 1]]_j]_i$
$MAX := 0$
$i = 1...M(1)$
$[_i\ IMAX := -1$
IF $(SEGED(i) > MAX) \wedge (SORJEL(i) > 0)$
THEN $[MAX := \max(MAX, SEGED(i)); IMAX := i]]_i$
$SUM := SUM + MAX$
IF $(SUM < n - Q) \wedge (IMAX \neq -1)$
THEN $[SM := SM + 1]$ **ELSE** [**GOTO** $label2$]
$SORJEL(IMAX) := -1$
$j = 1...n(1)$
$[_j$ **IF** $(OSZLJEL(i) > 0) \wedge (A(SOR(IMAX), j) = 1)$
THEN $[OSZJEL(i) := -1]]_j$
GOTO $label1$
$label2 : [M := min(M, SM);$ **STOP**]

5^o A $\mathcal{J}^*$-ot optimálisan kiegészítő $\mathcal{J}^{**}$ halmaz előállítása

Az előző szakaszban meghatározott M darab tanulópontból kiválasztjuk azt a minimális elemszámú $\mathcal{J}^{**}$ ponthalmazt, melyet $\mathcal{J}^{*}$-gal egyesítve egy optimális janicsárhalmazt kapunk. Az algoritmus által feltöltött KIV nevű tömbben az M egy k-adosztályú kombinációjának indexei állnak.

$k = 1...M(1)$
$[_k i = 1...k(1)[_i KIV(i) := i]_i$
$label1:$
$[i = 1...n(1)[_i JEL(i) := OSZL(i)]_i$
$COUNT := 0$
$j = 1...k(1)$
$[_j INDEX := SOR(KIV(j))$
$l = 1...n(1)$
$[_l$**IF** $JEL(l) > 0$ **THEN** $[$**IF** $A(INDEX, JEL(l)) = 1$
THEN $[JEL(l) := -1; COUNT := COUNT + 1]]_l]_j$
IF $COUNT + Q + SZAM + K = n$ **THEN GOTO** $label2$
$LEPT := k$
$i = k...1(-1)$
$[_i$**IF** $KIV(i) > M - k + i - 1$ **THEN** $LEPT := LEPT - 1]$
IF $LEPT = 0$ **THEN GOTO** $label3$
$KIV(LEPT) := KIV(LEPT) + 1$
IF $LEPT < k$
THEN $[j = LEPT + 1...k(1)[_j KIV(j) := KIV(LEPT) + j - LEPT]_j$
GOTO $label1$
$label3 : S1 := 1]_i]_k$
$label2 : [M := k;$ **STOP**$]$.

A **STOP**-ra ugrás pillanatában az optimális janicsárhalmaz elemszáma $Q + k$. $\mathcal{J}^{*}$-hoz tartozik a j indexű tanulópont, ha $SOR(j) < -n$, a kiegészítő $\mathcal{J}^{**}$ halmazhoz tartozik egy i indexű tanulópont, ha $i = SOR(KIV(l))$ áll fenn valamely $l \in \{1, 2, ..., k\}$ esetén.

2.8.3. Számítógépes kísérletek

A kísérlet körülményei

A tananyag $n = 100$ kétdimenziós független $N(0, 1)$ komponensű véletlenszámokkal generált $\mathcal{T}_n = \{\underline{t}_i, i = 1, 2, ..., n\}$ vektorhalmaz volt. Az adatokat az F1. függellék tartalmazza.

A tananyaghoz *három* különböző metrikát definiáltunk:

a.) Euklideszi-metrika: $d_E(\underline{x},\underline{y}) = \sqrt{(x_1-y_1)^2+(x_2-y_2)^2}$;

b.) Csebisev-metrika: $d_{Cs}(\underline{x},\underline{y}) = \max_{i=1,2}|x_i-y_i|$;

c.) Manhattan-metrika: $d_M(\underline{x},\underline{y}) = |x_1-y_1|+|x_2-y_2|$.

A metrikákhoz tartozó távolságmátrixokat SPSS programmal számoltuk ki.

Két pontot választottunk ki osztályozásra: $\underline{x}_1 = (0,0)$ és $\underline{x}_2 = (3,-2)$. Az első pont a tananyag centrumában, a másik a centrumtól meglehetősen távol helyezkedik el.

Kétféle keresési stratégiát alkalmaztunk:

a.) Minimax-keresés $\mathcal{K}_3$-kizárással (minimax);
b.) Véletlen-sorrendű keresés $\mathcal{K}_3$-kizárással (random).

Az a.) esetben az algoritmusokat három különböző pontból indítottuk: egy véletlen tanulópontból (t_1, t_{35}), a legközelebbi társból (t_{NN}) illetve a legtávolabbi társból (t_{FN}). Az első egy tipikus esetet, a második a legszerencsésebb esetet, a harmadik a legszerencsétlenebb esetet jelenti. A b.) esetben két véletlen permutációt generáltunk ($perm1, perm2$), amelyek előírták a keresési sorrendet. A permutációk a F2. függellékben láthatók. Excel programmal az összes lehetséges beállítás mellett elvégeztük az NN-keresést, ami 30 futtatást jelent. Minden beállításhoz meghatároztuk az elvileg szükséges minimális lépésszámot az optimális janicsárhalmazok megkonstruálása révén. A 2.1. sz. táblázatban foglaljuk össze a futási tapasztalatokat. A táblázat első oszlopa a futtatás sorszámát, a második oszlopában a metrikát, a harmadikban az osztályozandó pontot, a negyedik oszlopban a beállított keresési stratégiát olvashatjuk ki. Az ötödik oszlop tartalmazza, hogy hányadik távolságszámítás után állt le az algoritmus. A leállás pillanatában biztosan lehet tudni, hogy melyik az osztályozandó pont legközelebbi társa. A hatodik oszlop tartalmazza az NN-keresés minimális lépésszámát. A hetedik oszlopban az olvasható, hányszorosa a lépésszám a minimális lépésszámnak. A nyolcadik oszlopban az olvasható, hogy a tanulópontok hány százalékának kellett az osztályozandó ponttól vett távolságát kiszámítani. A futtatások részletes adatai és az Excel program számításai a F3. függellékben illetve a weben a *www.szit.bme.hu/ kela* címen olvashatók.

No.	Metrika	oszt. pont	stratégia	Lépésszám	Min. lépéssz.	arány1	%
1.	d_E	$(0,0)$	minimax, t_{35}	5	2	2.5	5%
2.	d_E	$(0,0)$	minimax, t_{NN}	5	2	2.5	5%
3.	d_E	$(0,0)$	minimax, t_{FN}	4	2	2	4%
4.	d_E	$(0,0)$	random, $perm1$	3	2	1.5	3%
5.	d_E	$(0,0)$	random, $perm2$	9	2	4.5	9%
6.	d_E	$(3,-2)$	minimax, t_1	4	3	1.33	4%
7.	d_E	$(3,-2)$	minimax, t_{NN}	4	3	1.33	4%
8.	d_E	$(3,-2)$	minimax, t_{FN}	4	3	1.33	4%
9.	d_E	$(3,-2)$	random, $perm1$	3	3	1	3%
10.	d_E	$(3,-2)$	random, $perm2$	9	3	3	9%
11.	d_{Cs}	$(0,0)$	minimax, t_1	4	3	1.33	4%
12.	d_{Cs}	$(0,0)$	minimax, t_{NN}	4	3	1.33	4%
13.	d_{Cs}	$(0,0)$	minimax, t_{FN}	3	3	1	3%
14.	d_{Cs}	$(0,0)$	random, $perm1$	5	3	1.66	5%
15.	d_{Cs}	$(0,0)$	random, $perm2$	8	3	2.66	8%
16.	d_{Cs}	$(3,-2)$	minimax, t_1	6	3	2	6%
17.	d_{Cs}	$(3,-2)$	minimax, t_{NN}	6	3	2	6%
18.	d_{Cs}	$(3,-2)$	minimax, t_{FN}	3	3	1	3%
19.	d_{Cs}	$(3,-2)$	random, $perm1$	10	3	3.33	10%
20.	d_{Cs}	$(3,-2)$	random, $perm2$	7	3	2.33	7%
21.	d_M	$(0,0)$	minimax, t_1	3	2	1.5	3%
22.	d_M	$(0,0)$	minimax, t_{NN}	3	2	1.5	3%
23.	d_M	$(0,0)$	minimax, t_{FN}	3	2	1.5	3%
24.	d_M	$(0,0)$	random, $perm1$	5	2	2.5	5%
25.	d_M	$(0,0)$	random, $perm2$	7	2	3.5	7%
26.	d_M	$(3,-2)$	minimax, t_1	4	3	1.33	4%
27.	d_M	$(3,-2)$	minimax, t_{NN}	5	3	1.66	5%
28.	d_M	$(3,-2)$	minimax, t_{FN}	4	3	1.33	4%
29.	d_M	$(3,-2)$	random, $perm1$	8	3	2.66	8%
30.	d_M	$(3,-2)$	random, $perm2$	7	3	2.33	7%

2.1. sz. TÁBLÁZAT. A kísérleti futtatások összefoglaló táblázata

A tapasztalatok összesítése

1. Mindegyik beállítás mellett kevesebb mint a pontok 10%-t kellett feldolgozni a kizárásos NN-kereséseknél.

2. Az esetek többségében t_{FN}-ből indítva ért leghamarabb révbe a minimax keresés. (Nem ez a legrosszabb eset!)

3. A metrikák változtatása nincs észrevehető változással a lépésszámra.

4. A minimax kereső stratégia gyorsabbnak tűnik a kizárásos random keresésnél. Az alábbi táblázat hat beállítás esetén az *átlagos lépésszámokat* mutatja a két keresési stratégiánál:

Beállítás	**minimax**	**random**
$d_E, x = (0,0)$	4,66	6
$d_E, x = (3,-2)$	4	6
$d_{Cs}, x = (0,0)$	3,66	6,5
$d_{Cs}, x = (3,-2)$	5	8,5
$d_M, x = (0,0)$	3	6
$d_M, x = (3,-2)$	4,33	7,5

5. A harminc futtatás mindegyikénél a janicsárhalmaz-magban csak a legközelebbi társnak megfelelő tanulópont került. Ez elvileg nem mindig kell, hogy így legyen. Ha több azonos távolságra lévő tanulópont van, illetve a távolságuk kicsi, $\mathcal{J}^*$ többelemű is lehetne. Igaz ennek valószínűsége alkalmazások esetén elhagyagolható.

2.8.4. NN-kereső algoritmusok működési idejeinek összehasonlítása

Először ismertetjük a két összehasonlítandó NN-kereső algoritmust, majd a műveleti idejükre vonatkozó képleteket adjuk meg. Az első algoritmus a hagyományos, szekvenciális kereső algoritmus, a második a $\mathcal{K}_3$ kizáráson alapuló minimax-kerső algoritmus.

A leírásban szereplő változók, konstansok és eljárásfüggvények definícióival kezdünk.

Változók

$T_{\min}$ a már kiszámított távolságok között a legkisebb értéke;
i ciklusváltozó;
K a már végrehajtott távolságszámítások száma;
$UJ, LABEL$ egész értékű tömb, ami még a fel nem dolgozott tanulópontok indexeit tárolja;
$LEPT$ segédváltozók;
$NEXT$ a feldolgozásban következő tanulópont indexe;
NN a feldolgozott tanulópontok között a -hez legközelebbinek az indexe;
S a még fel nem dolgozott tanulópontok minimax keresőértékeit tartalmazó tömb;
$S_{\min}$ a minimális minimax-keresőérték;
T a legutóbb kiszámított távolságok tömbje;

Konstansok

D a tanulópontok egymástól vett távolságainak mátrixa;
n a tanulópontok száma;

Eljárásfüggvények

$DISTANCE(i)$ az i indexű tanulópont és x távolságát meghatározó függvény.

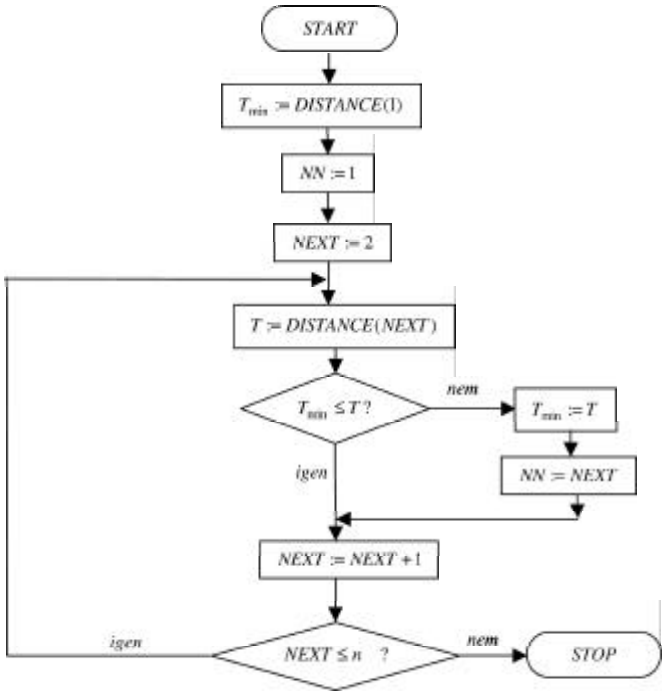
A szekvenciális keresés algoritmusa

$T := DISTANCE(1)$
$T_{\min} := T; NN := 1; NEXT := 2;$
$label1:$
$T := DISTANCE(NEXT)$
IF $T_{\min} > T$ **THEN** $[T_{\min} := T; NN := NEXT];$
$NEXT := NEXT + 1$
IF $NEXT \leq n$ **THEN GOTO** $label1$
STOP

A **STOP** pillanatában NN tartalmazza a legközelebbi társ indexét, $T_{\min}$ pedig a minimális távolságot.

A $\mathcal{K}_3$ kizárásos minimax NN-kereső algoritmus

$NEXT := 1; NN := 1; K := 1$
$i = 1...n-1(1)[_i\ LABEL(i) := i+1; S(i+1) := 0]_i$
$T := DISTANCE(1); T_{\min} := T; S_{\min} := \infty$
$label2:$
$i = 1...n-K(1)$
$[_i\ S(LABEL(i)) := max(S(LABEL(i)), |T - D(NEXT, LABEL(i)|]_i$
IF $S_{\min} > S(LABEL(i))$ **THEN**
$[S_{\min} := S(LABEL(i)); UJ := LABEL(i); LEPT := i]$
$NEXT := UJ$
IF $S_{\min} > T_{\min}$ **THEN GOTO** $label1$
$T := DISTANCE(NEXT); K := K + 1$
$T_{\min} := \min(T_{\min}, T)$
IF $T_{\min} = T$ **THEN** $NN := NEXT$
$i := LEPT..n-K(1)[_i\ LABEL(i) := LABEL(i+1)]_i$
GOTO $label2$



Az szekvenciális kereső algoritmus blokkdiagramja.

*label*1 : **STOP**

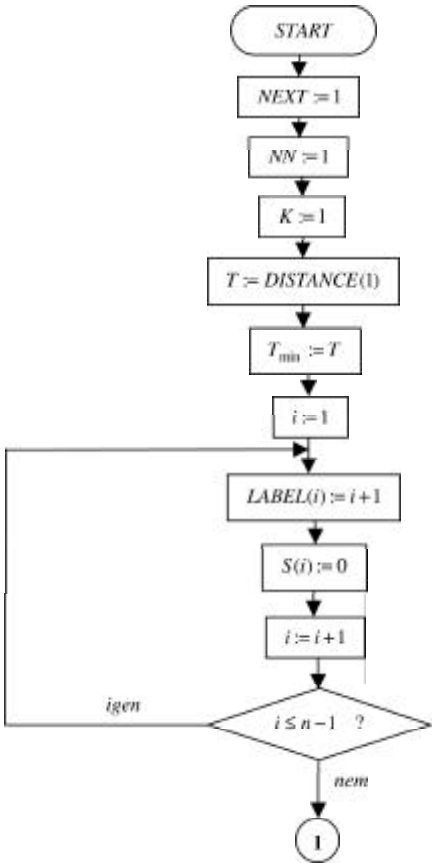
A **STOP** pillanatában NN tartalmazza a legközelebbi társ indexét, $T_{\min}$ pedig a minimális távolságot. Az algoritmus K lépésben jutott el a legközelebbi társhoz.

A két kereső algoritmus működési idejének összehasonlítása

Ebben a pontban a $\mathcal{K}_3$ kizárással működő minimax NN-kereső algoritmus futási idejét hasonlítjuk össze a hagyományos, szekvenciális NN-kereső algoritmussal, amikor a tanulópontok száma n.

Jelölések:

a két valós szám különbsége abszolút értékének kiszámításához szükséges gépidő;



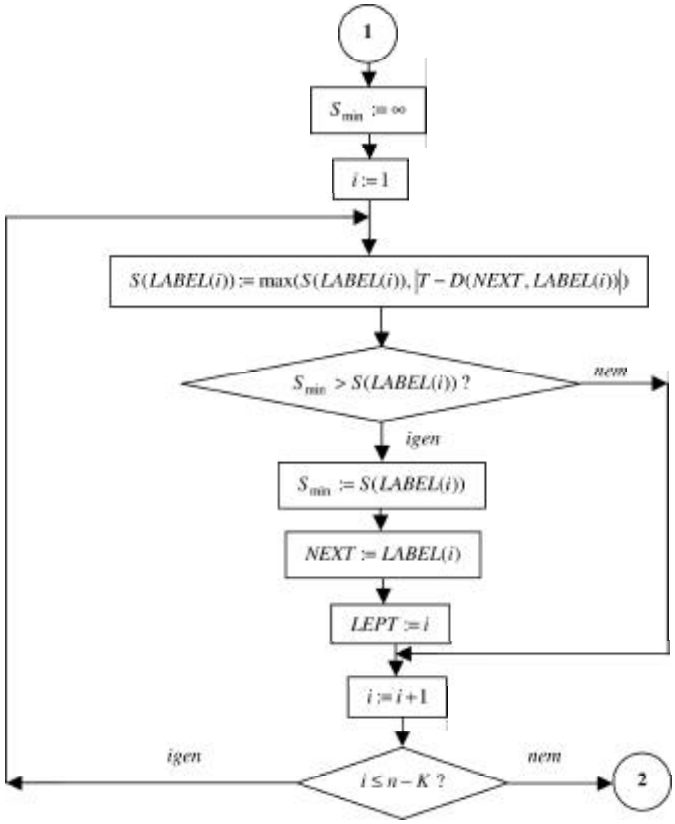
A $\mathcal{K}_3$ kizárásos minimax NN-kereső algoritmus blokkdiagramja (1. rész)

b azt az időtartamot jelöli, ami két valós szám közül a kisebbik meghatározásához szükséges;

c azt a gépidőt jelöli, ami két metrikus pont távolságának kiszámításához szükséges;

Nyílván $a \approx b$ és a, b sokkal kisebb c-nél. Pl., ha 100-dimenziós vektorok távolságait akarjuk kiszámítani a $d_M\left(\underline{x}, \underline{y}\right) = \sum_{i=1}^{100} |x_i - y_i|$ *Manhattan*-metrikával, akkor $c = 100a + 9\varepsilon$, ahol ε az összeadás műveleti igénye.

Az algoritmusok összidejének becsléseivel foglalkozunk, nem számolunk olyan



A $\mathcal{K}_3$ kizárásos minimax NN-kereső algoritmus blokkdiagramja (2. rész)

elhanyagolható időtartamokkal, mint az értékadás, egész számok közötti műveletek, egész számok összehasonlítása, irányításvezérlés (GOTO) stb.

A szekvenciális NN-keresés T_s futási ideje:

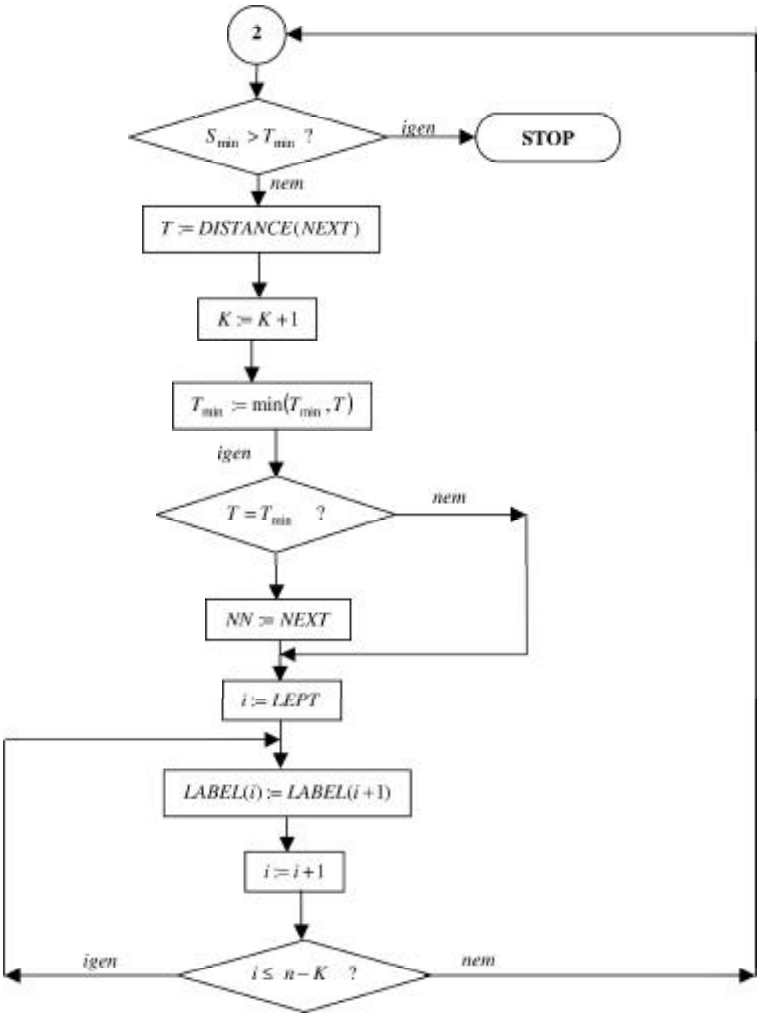
$T_s \approx n \cdot c + (n-1) \cdot b$

A $\mathcal{K}_3$ kizárással működő minimax NN-kereső algoritmus T_{mm} futási ideje:

$T_{mm} \approx K \cdot c + K \cdot \left[n - \frac{K+1}{2}\right] \cdot (a + 2b) + K \cdot b,$

ahol K az elvégzett távolságszámítások száma.

A két becslés közül csak az utóbbi szorul magyarázatra. Az első távolság kiszámítása után $n-1$ tanulóponthoz kell a minimax keresőértéket kiszámolni. A kiszámolt keresőértékek minimumának meghatározásához $n-1$ összehasonlítás szükséges. További $2b$ időtartam kell az $S_{\min} > D_{\min}$ megállítási feltétel el-



A $\mathcal{K}_3$ kizárásos minimax NN-kereső algoritmus blokkdiagramja (3. rész)

lenőrzéséhez és a $D_{\min}$ pillanatnyi minimális távolság aktualizálásához. Így az első lépésben az össz műveleti idő:

$$c + (n-1) \cdot (a+b) + (n-1) \cdot b + 2b = c + (n-1)(a+2b) + 2b.$$

A második távolság kiszámítása után $n-2$ tanulópont marad, így a keresőlépés, a megállítási szabály ellenőrzése és a minimális távolság aktualizálása $c+(n-2)(a+2b)+2b$ időtartam alatt történik meg. Folytatva megállapítható, hogy K lépés alatt összesen $K \cdot c + \sum_{i=1}^{K} (n-i) \cdot [a+2b] + 2K \cdot b = K \cdot c + K\left[n - \frac{K+1}{2}\right] \cdot (a+2b) + 2Kb$ idő telik el.

T_s és T_{mm} összehasonlítása

A továbbiakban feltételezzük, hogy $a=b$ és $c=\alpha \cdot a$. Ekkor $T_s \approx a\left[(\alpha+1) \cdot n - 1\right]$ és $T_{mm} = a\left[(\alpha+2) \cdot K + 3K \cdot n - 1.5 \cdot K \cdot (K+1)\right]$. Akkor vesz kevesebb időt igénybe a kizárásos minimax kereső NN-algoritmus, ha $T_{mm} < T_s$ azaz $(\alpha+1) \cdot n - 1 > (\alpha+2) \cdot K + 3K \cdot n - 1.5 \cdot K \cdot (K+1)$ azaz,

$(*) \qquad n > \frac{1+(\alpha+2)K-1.5K(K+1)}{\alpha+1-3K} = \gamma(\alpha, K)$

feltéve, hogy $\alpha > 3K-1$. Jelölje $\gamma(\alpha, K)$ a $(*)$ jobboldalán álló racionális tört kifejezést. A 2.2. sz. táblázatban megadjuk $\gamma(\alpha, K)$ értékét néhány α és K esetében. A táblázatból kiolvasható például, hogyha c százszorosa a-nak ($\alpha > 100$), akkor ha tíz keresésen belül eljutunk a legközelebbi társhoz a minimax algoritmussal ($K \leq 10$), akkor ez már kevesebb idő alatt zajlik le, mintha a szekvenciálissal próbálkoztunk volna, ha a tananyag elemszáma legalább 13 (l. a táblázat 9. sorát!). A $\gamma(\alpha, K)$ adat tehát a tananyag n mintaelemszámára ad meg alsó korlátot.

α	K	$\gamma(\alpha, K)$
10	1	1,25
10	2	3,2
10	3	9,5
100	1	1,02
100	2	2,06
100	3	3,14
100	4	4,26
100	5	5,42
100	10	12,05
100	15	20,91
100	20	34,41
100	25	60,61
100	30	151,45

α	K	$\gamma(\alpha, K)$
1000	10	10,15
1000	20	20,62
1000	30	31,47
1000	40	42,70
1000	50	54,38
1000	100	121,33
1000	150	211,12
1000	200	349,38
1000	250	623,02
1000	300	1535,16

2.2. sz. Táblázat. A tananyag elemszámára (n) vonatkozó alsókorlátok táblázata

3. fejezet

Metrikák automatikus skálázása

3.1. A probléma megfogalmazása

A gyakorlati alkalmazásokban sokszor felvetődik az alábbi osztályozási feladat. Adott egy objektumhalmaz és kategóriák egy véges halmaza. Az objektumokról tudjuk, hogy ezek véges sok kategória (osztály) között oszlanak szét. Hogyan lehet az objektumok megmérésével a kategóriákra következtetni? Például jellemezni lehet-e a választópolgárokat az iskolai végzettségre, korra, foglalkozásra, jövedelemre stb. vonatkozó mérési adatokkal úgy, hogy megbízhatóan következtetni lehessen arra, hogy melyik pártra fognak szavazni? Milyen adatokat gyűjtsön össze az orvos a páciensről ahhoz, hogy felismerje a betegséget? Mekkora felbontással kell digitalizálni a kézzel írt jelet ahhoz, hogy az azonosítható legyen a megfelelő betűvel az abc-ben? Az objektumokhoz hozzárendelt X adatot *alakzatnak* nevezik, a kategóriák Y azonosítóját pedig *osztálynak*. Az objektumok kiválasztása általában véletlenszerű, ezért sztochasztikus modellt alkalmaznak. Az alkalmazások nagy részében az osztályok száma kettő, ezért a matematikai modellt ebben a speciális esetben adjuk meg.

Tegyük fel, hogy adott az $(\Omega, \mathcal{F}, \mathbf{P})$ valószínűségi mezőn az (X, Y) véletlen elem-pár, ahol X az $(\mathcal{X}, d)$ metrikus téren veszi fel az értékeit, Y pedig bináris. A feladat az, hogy X segítségével jól becsüljük Y-t. Ehhez megadunk egy $\phi : \mathcal{X} \to \{0, 1\}$ *döntésfüggvényt*, melyet az $L(\phi) = \mathbf{P}(\phi(X) \neq Y)$ *hibavalószínűséggel* jellemzünk. Arra törekszünk, $L(\phi)$ minél kisebb legyen. Amennyiben ismert az (X, Y) pár együttes eloszlása, meghatározható egy optimális ϕ^* döntésfüggvény, melyre $\phi^* = \arg \min_{\phi:\mathcal{X}\to\{0,1\}} L(\phi)$ teljesül. A ϕ^* a *Bayes*-döntésfüggvény, $L(\phi^*) = L^*$ pedig a Bayes-hibavalószínűség. Az alkalmazásokban nem ismerjük az (X, Y)

együttes eloszlását, viszont rendelkezésünkre áll egy véges elemszámú statisztikai minta, melyet *tananyagnak* nevezünk: $\mathcal{D}_m = \{(X_1, Y_1), (X_2, Y_2), ..., (X_m, Y_m)\}$. Egy tanulóalgoritmus $\mathcal{D}_m$ segítségével előállít egy $\phi_m : \mathcal{X} \times \mathcal{D}_m \to \{0, 1\}$ döntésfüggvényt, melyet az $L(\phi_m) = \mathbf{P}(\phi_m(X) \neq Y \mid \mathcal{D}_m)$ feltételes hibavalószínűséggel jellemzünk. Olyan ϕ_m megkonstruálása a cél amelyről belátható, hogy $\lim_{m\to\infty} \mathbf{E}L(\phi_m) = L^*$. Az ilyen döntésfüggvényeket előállítani tudó tanulóalgoritmusokat (gyengén) *konzisztensnek* nevezzük.

A legközelebbi társ módszer az egyik legismertebb és legrégibb tanulóalgoritmus. Az X alakzatot ahhoz a kategóriához sorol, amelyik kategóriához az X-hez legközelebbi X_i tanulópont is tartozik $\mathcal{D}_m$-ből, vagyis $d(X, X_i) < d(X, X_j)$ minden $(X_j, Y_j) \in \mathcal{D}_m$-re. A legközelebbi k társ módszereknél X-et abba a kategóriába osztályozzuk, amibe a $\mathcal{D}_m$-beli k legközelebbi társ többsége is tartozik. A legközelebbi társ módszereknek igen nagy irodalma van, itt most DEVROYE, GYÖRFI, LUGOSI [8], COVER és HART [4], STONE [32], DUDA és HART [9] valamint FUKUNAGA [16] összefoglaló művére hivatkozunk. COVER és HART [4] foglalkoztak először a legközelebbi társ módszer aszimptotikus tulajdonságaival. Bebizonyították, hogyha az $(\mathcal{X}, d)$ metrikus térben az x osztályozandó pont 1 valószínűséggel az osztályok feltételes sűrűségfüggvényének folytonossági pontja, akkor az osztályozási hiba nagysága aszimptotikusan a Bayes-hiba egyszerese és kétszerese közé fog esni függetlenül a d metrika megválasztásától. Ezt az eredményt többen általánosították arra az esetre is, amikor az X alakzat eloszlására semmilyen kikötés sincsen. (L. pl. DEVROYE [7].) Szeparábilis metrikus térben a legközelebbi szomszéd módszer konzisztens döntésfüggvények sorozatát fogja előállítani, ha a Bayes-hiba 0. A konvergencia sebessége persze függ d megválasztásától, ezért gyakorlati alkalmazásokban -ahol mindig véges elemszámú $\mathcal{D}_m$ tananyag áll rendelkezésre- nem közömbös milyen metrika alapján osztályozunk.

SHORT és FUKUNAGA [31] az $\mathcal{X} = \mathbb{R}^p$ vektortérben foglalkozott az optimális metrika megválasztásával. A szerzők az $\underline{x}$ osztályozandó vektor függvényében javasolják az Euklideszi-metrika alábbi módosítását:

$$d(\underline{x}, \underline{x}_i) = \left|\underline{A}^T(\underline{x}) \cdot (\underline{x} - \underline{x}_i)\right|, \text{ ahol}$$

$$\underline{A}(\underline{x}) = \underline{M}_1(\underline{x}) - \underline{M}(\underline{x}) \text{ és } \underline{M}_1(\underline{x}) = \frac{1}{k_1} \sum_{i=1}^{k} (\underline{x} - \underline{x}_{\gamma_i}) \cdot I_{\left\{\underline{x}_{\gamma_i} \text{ az 1 osztályhoz tartozik}\right\}},$$

$$\underline{M}(\underline{x}) = \frac{1}{k} \sum_{i=1}^{k} (\underline{x} - \underline{x}_{\gamma_i}), k_1 = \sum_{i=1}^{k} I_{\left\{\underline{x}_{\gamma_i} \text{ az 1 osztályhoz tartozik}\right\}}, \text{ és } \underline{x}_{\gamma_1}, \underline{x}_{\gamma_2}, ..., \underline{x}_{\gamma_k}$$

az $\underline{x}$ legközelebbi k társa a $\mathcal{D}_m$ tananyagban. Az alkalmazott $\underline{A}(\underline{x})$ súlyvektor nem más, mint az osztályozás négyzetes hibájához tartozó gradiens-vektor statisztikai becslése.

BROWN és KOPLOWITZ [1] egészen más elvek alapján javasolja a metrika kiválasztást. Módszerükkel nem reprezentatív tananyagok esetén lehet az osztályozási torzulást kiküszöbölni. A javasolt metrika $d^*(\underline{x},\underline{x}_i) = A_j d(\underline{x},\underline{x}_i)$ alakú, ahol $\underline{x}_i$ a j-edik osztályhoz tartozik, és A_j alkalmasan megválasztott súly.

Az $\mathcal{X} = \mathbb{R}^p$ térben, az Euklideszi metrika asszimetrikus súlyozásával osztályoz RICCI és AVESANI [27]. Az oszályozást a $d(\underline{x},\underline{y}) = \left(\sum_{j=1}^{p} w_j(\underline{x},\underline{y})(x_j - y_j)^2\right)^{\frac{1}{2}}$ metrikával végzik, ahol $w_j(\underline{x},\underline{y}) = \begin{cases} w_j^0(\underline{x}) & \text{, ha } y_j < x_j \\ w_j^1(\underline{x}) & \text{, ha } y_j \geq x_j \end{cases}$, $w_j^0(\underline{x}), w_j^1(\underline{x}) \in [0,1]$. A képletben $\underline{x}$ a $\mathcal{T}_n$ tananyag egy eleme, $\underline{y} \in \mathbb{R}^p$ tetszőleges. Látható, hogy a metrika $\underline{w}$ súlyvektora függ egyrészt a tananyag pontjaitól is, de a tananyag pontjainak az osztályozandó ponttal való kapcsolatától is.

A $w_j(\underline{x},\underline{y})$ súlyokat egy előkészítő feldolgozás alkalmával állítják be egy $\mathcal{U}_m$ tesztpont-halmaz segítségével, hogy az osztályozás a lehető legpontosabb legyen. A javasolt algoritmus az alábbi:

1^o Az egyenletes súlytényezőkből indulunk ki: $w_j^0(\underline{x}) = w_j^1(\underline{x}) = \frac{1}{n}, \underline{x} \in \mathcal{T}_n$.

2^o Legyen $count := 0$. Osztályozzuk $\mathcal{U}_m$ elemeit a
$d(\underline{x},\underline{y}) = \left(\sum_{j=1}^{p} w_j(\underline{x},\underline{y})(x_j - y_j)^2\right)^{\frac{1}{2}}$ metrika segítségével az NN-módszerrel.
Jelölje $\underline{x}^* = NN(\underline{y},\mathcal{T}_n)$ az $\underline{y}$ legközelebbi társát! Minden $\underline{y} \in \mathcal{U}_m$ osztályozása után módosítjuk a súlytényezőket:
- Ha $class(\underline{y}) = class(\underline{x}^*)$, akkor $count := count + 1$ és

$$R_j^0(\underline{w}(\underline{x}^*,\underline{y})) = \begin{cases} w_j^0(\underline{x}^*) - \alpha w_j^0(\underline{x}^*)(x_j^* - y_j) & \text{, ha } y_j < x_j^* \\ w_j^0(\underline{x}^*) & \text{, egyébként} \end{cases},$$

$$R_j^1(\underline{w}(\underline{x}^*,\underline{y})) = \begin{cases} w_j^1(\underline{x}^*) & \text{, ha } y_j < x_j^* \\ w_j^1(\underline{x}^*) - \alpha w_j^1(\underline{x}^*)(y_j - x_j^*) & \text{, egyébként} \end{cases}$$

- Ha $class(\underline{y}) \neq class(\underline{x}^*)$, akkor

$$P_j^0(\underline{w}(\underline{x}^*,\underline{y})) = \begin{cases} w_j^0(\underline{x}^*) + \frac{\beta}{2}\left(1 - \left|2w_j^0(\underline{x}^*) - 1\right|\right)(x_j^* - y_j) & \text{, ha } y_j < x_j^* \\ w_j^0(\underline{x}^*) & \text{, egyébként} \end{cases}$$

$$P_j^1(\underline{w}(\underline{x}^*,\underline{y})) = \begin{cases} w_j^1(\underline{x}^*) & \text{, ha } y_j < x_j^* \\ w_j^1(\underline{x}^*) + \frac{\beta}{2}\left(1 - \left|2w_j^1(\underline{x}^*) - 1\right|\right)(y_j - x_j^*) & \text{, egyébként} \end{cases}$$

3^o Az algoritmus akkor áll meg, amikor egymás után két iterációban is csökken *count* értéke a ciklus végén, azaz romlani kezd az osztályozás pontossága. Különben új ciklust indítva visszatérünk a 2^o-ra.

3.1. megjegyzés. *1. Az $\alpha, \beta \in [0,1]$ paraméterek a metrikasúlyok változásának sebességét szabályozzák.*

Ebben a fejezetben annak a lehetőségét vizsgáljuk, hogyan lehet adott $\mathcal{X}$ alaphalmazhoz metrikafüggvények egy Δ halmazából az optimálisat kiválasztani, amivel egy teszthalmazon az osztályozási hiba relatív gyakoriságát minimalizáljuk. A leírás során az $\mathcal{X}$ alaphalmaz a p-dimenziós $\mathbb{R}^p$ vektortér lesz. Ennél általánosabban csak az 3.5. szakaszban fogjuk a problémát tárgyalni. Egy $d(\underline{x},\underline{u}) = \sum_{i=1}^{p} \rho(x_i,u_i)$ szerkezetű bázismetrikából indulunk ki, ahol ρ egy metrika $\mathbb{R}$-en. Ilyen pl. az *Euklideszi-*, *Minkowsky-*, *City-blokk* vagy a *Canberra-*metrika.

A bázismetrikával generáljuk a

$$\Delta = \left\{ d_{\underline{A}}; d_{\underline{A}}(\underline{x},\underline{u}) = \sum_{i=1}^{p} A_i \rho(x_i,u_i), \underline{A} \geq \underline{0} \right\}$$

halmazt. Δ minden eleme metrika $\mathbb{R}^p$-n. A célkitűzésünk az, hogy ebből válasszuk ki az optimálisat.

Az n elemszámú tananyagot egy m elemszámú $\mathcal{D}_m^*$ és egy l elemszámú $\mathcal{D}_l^{**}$ részhalmazra bontjuk fel. $(n = m + l)$. A továbbiakban $\mathcal{D}_m^*$ játsza a tananyag szerepét az osztályozáskor. $\mathcal{D}_l^{**}$-et tesztanyagnak fogjuk nevezni. Jelölje a tananyagot

$$\mathcal{D}_m^* = \{(\underline{x}_1,y_1),(\underline{x}_2,y_2),...,(\underline{x}_m,y_m)\}, \text{ ahol}$$

$\underline{x}_i$ az i-edik *tanulópont*, y_i az i-edik *tanítás.*

A tesztanyag

$$\mathcal{D}_l^{**} = \left\{ \left(\underline{x}_{m+1},y_{m+1}\right), \left(\underline{x}_{m+2},y_{m+2}\right), ..., \left(\underline{x}_{m+l},y_{m+l}\right) \right\}$$

lesz, ahol $\underline{x}_{m+j}$ az j-edik *tesztpont.*

Jelölje a Δ metrika-családhoz tartozó súlyozott metrikás legközelebbi társ döntésfüggvények halmazát C_m! A döntésfüggvényeket a kapcsolatos $\underline{A}$ skálázóvektorral fogjuk indexelni.

A $\phi_{\underline{A}}$ döntésfüggvényhez tartozó, a $\mathcal{D}_m^*$ feltételre vett feltételes döntési hiba valószínűségét $L\left(\phi_{\underline{A}}\right) = \mathbf{P}\left(\phi_{\underline{A}}(\underline{X}) \neq Y \mid \mathcal{D}_m\right)$ jelöli. A $\phi_{\underline{A}}$ döntésfüggvény az $L\left(\phi_{\underline{A}}\right) - \inf_{\phi \in C_m} L(\phi)$ különbséggel jellemezhető.

Jelölje $g_m^* \in C_m$ azt a döntésfüggvényt, amelynél

$$\hat{L}_{m,l}(g_m^*) = \min_{\phi_{\underline{A}} \in C_m} \hat{L}_{m,l}\left(\phi_{\underline{A}}\right), \text{ ahol}$$

$\hat{L}_{m,l}\left(\phi_{\underline{A}}\right) = \frac{1}{l}\sum_{i=1}^{l} I_{\left\{\phi_{\underline{A}}\left(\underline{x}_{m+i}\right)\neq y_{m+i}\right\}}$ az $L\left(\phi_{\underline{A}}\right)$ döntési hibavalószínűség empirikus becslése.

3.2. A skálázott metrikás osztályozás konvergencia sebessége

Ennek a fejezetnek a legfontosabb eredményét az 3.1. tételben foglaljuk össze. Ismeretes (L. pl. DEVROYE [7]), hogyha az L^* Bayes-hiba 0, a legközelebbi szomszéd döntésfüggvényeinek sorozata konzisztens tetszőleges olyan d metrika esetén, ahol $(\mathcal{X}, d)$ metrikus tér szeparábilis. Tehát ebben a kedvező esetben minden $d \in \Delta$ metrika esetén biztos a legközelebbi társ döntésfüggvényeinek konzisztenciája. Továbbá igaz az is, hogy $\mathbf{E}\left(L\left(\phi_m\right)\right) \to L_{NN}$ minden $\phi_m \in C_m$ és tetszőleges eloszlás esetén, ahol L_{NN} jelöli a legközelebbi társ döntési hibavalószínűségét. Nem mindegy azonban, hogy a fenti konvergencia milyen sebességgel történik meg, azaz a tananyag végessége miatt nem közömbös a d metrika megválasztása. A 3.3. szakaszban adjuk meg azt az algoritmust, amely kiválasztja Δ-ból az empirikus hibát minimalizáló elemet. Ehhez a metrikához tartozó a $g_m^* \in C_m$ döntésfüggvényre igaz az alábbi tétel:

3.1. tétel. *Az $\hat{L}_{m,l}\left(\phi\right)$ empirikus hibavalószínűséget minimalizáló g_m^* döntésfüggvényre minden $\varepsilon > 0$ esetén fennáll, hogy*

$$\mathbf{P}\left(L\left(g_m^*\right) - \inf_{\phi \in C_m} L\left(\phi\right) > \varepsilon \mid \mathcal{D}_m^*\right) \leq 4e^8 \left(l^2 \binom{m}{2}\right)^p e^{-\frac{l\varepsilon^2}{2}}.$$

Az 3.1 tétel bizonyításához szükségünk lesz a döntésfüggvény-osztály struktúráját jellemző fogalmakra és állításokra, melyeket a DEVROYE-GYÖRFI-LUGOSI [8] könyvből vettünk át.

3.1. definíció. *Legyen $\mathcal{A}$ $\mathbb{R}^p$-beli mérhető halmazok egy osztálya. Tekintsük a $\{\underline{z}_1, \underline{z}_2, ..., \underline{z}_n\}$ p-dimenziós vektorhalmazt! Jelölje $\mathcal{N}_{\mathcal{A}}\left(\underline{z}_1, \underline{z}_2, ..., \underline{z}_n\right)$ azoknak a részhalmazoknak a számát, amelyek $\{\underline{z}_1, \underline{z}_2, ..., \underline{z}_n\} \cap A$, $A \in \mathcal{A}$ alakúak! Ekkor az $\mathcal{A}$ halmazosztály n-edik darabolási együtthatóján az*
$s\left(\mathcal{A}, n\right) = \max_{\forall\left\{\underline{z}_1, \underline{z}_2, ..., \underline{z}_n\right\}} \mathcal{N}_{\mathcal{A}}\left(\underline{z}_1, \underline{z}_2, ..., \underline{z}_n\right)$ *számot értjük.*

Tehát maximum $s(\mathcal{A},n)$ különböző részhalmaz lehet egy n-elemű $\mathbb{R}^p$-beli pontrendszerből $\mathcal{A}$-beli részhalmazokban. Szokás azt is mondani, hogy $\mathcal{A}$ maximum $s(\mathcal{A},n)$ részhalmazt „csíp ki" $\{\underline{z}_1,\underline{z}_2,...,\underline{z}_n\}$-ből. Nyilván $s(\mathcal{A},n)\leq 2^n$. Ha valamely $\{\underline{z}_1,\underline{z}_2,...,\underline{z}_n\}$ pontrendszer esetén $\mathcal{N}_{\mathcal{A}}(\underline{z}_1,\underline{z}_2,...,\underline{z}_n)=2^n$, akkor $\mathcal{A}$ „összedarabolja" $\{\underline{z}_1,\underline{z}_2,...,\underline{z}_n\}$-et. Ha viszont $s(\mathcal{A},n)<2^n$, akkor bármely $\{\underline{z}_1,\underline{z}_2,...,\underline{z}_n\}$ pontrendszer esetén van legalább egy olyan $\left\{\underline{z}_{i_1},\underline{z}_{i_2},...,\underline{z}_{i_k}\right\}$ részhalmaz, amely egyik $A\in\mathcal{A}$-nak sem részhalmaza. Az is világos, hogy ha $s(\mathcal{A},k)<2^k$ valamely k-ra, akkor $s(\mathcal{A},n)<2^n$ is igaz minden $n>k$-ra.

3.2. definíció. *Legyen $\mathcal{A}$ $\mathbb{R}^p$-beli mérhető halmazok egy legalább kételemű családja. Azt a legnagyobb k egész számot, melyre $s(\mathcal{A},k)=2^k$ az $\mathcal{A}$ halmazcsalád Vapnik-Chervonenkis dimenziójának (VC-dimenzió) nevezzük és $\mathcal{V}_{\mathcal{A}}$-val jelöljük. Ha $s(\mathcal{A},n)=2^n$ minden n-re, akkor definíció szerint $\mathcal{V}_{\mathcal{A}}=\infty$.*

(Tehát, ha $\mathcal{V}_{\mathcal{A}}<\infty$ és $n>\mathcal{V}_{\mathcal{A}}$, akkor szükségképpen $s(\mathcal{A},n)<2^n$.)

3.3. definíció. *Legyen C tetszőleges $\phi:\ \mathbb{R}^p\to\{0,1\}$ döntésfüggvények halmaza. Tekintsük $p+1$-dimenziós vektorok alábbi $\mathcal{A}_C$ halmazosztályát:*

$$\mathcal{A}_C=\{A;\ A=\{\{\underline{x};\ \phi(\underline{x})=1\}\times\{0\}\}\cup\{\{\underline{x};\ \phi(\underline{x})=0\}\times\{1\}\},\ \phi\in C\}.$$

Ezen $\mathcal{A}_C$ halmazosztály n-edik darabolási együtthatója lesz a C döntésfüggvények osztályának n-edik darabolási együtthatója: $S(C,n)=s(\mathcal{A}_C,n)$. C Vapnik-Chervonenkis dimenziója pedig $\mathcal{V}_C=\max\limits_{S(C,n)=2^n} n$ lesz.

Használjuk az 3.1. szakaszban bevezetett jelöléseket! Az n elemszámú tananyagot egy m elemszámú $\mathcal{D}_m^*$ és egy l elemszámú $\mathcal{D}_l^{**}$ részhalmazokra bontottunk. $\phi_{\underline{A}}$ jelöli a $d_{\underline{A}}(\underline{x},\underline{u})=\sum\limits_{i=1}^{p}A_i\rho(x_i,u_i)$ távolságfüggvényhez és $\mathcal{D}_m^*$-hez tartozó legközelebbi szomszéd elven működő döntésfüggvényt, $C_m=\left\{\phi_{\underline{A}};\underline{A}\geq\underline{0}\right\}$.

A C_m függvényrendszer l-edik darabolási együtthatójára vonatkozó egyenlőtlenséget fogunk bebizonyítani. Ehhez két segédtételre lesz szükségünk.

Az első segédtételben megfogalmazott állítás bizonyítása megtalálható pl. COVER [25] cikkében is. Az itt közölt bizonyítás jelöléseit később használni fogjuk. Legyen $\underline{n}\in\mathbb{R}^p$ tetszőleges. Ekkor a $H=\left\{\underline{x}\in\mathbb{R}^p;\underline{n}^T\underline{x}=0\right\}$ hipersíkkal az $\mathbb{R}^p$ teret két részre partícionálhatjuk: az $A_1=\left\{\underline{x}\in\mathbb{R}^p;\underline{n}^T\underline{x}>0\right\}$ és $A_2=\left\{\underline{x}\in\mathbb{R}^p;\underline{n}^T\underline{x}\leq 0\right\}$ partíciókra. Ha k darab különböző normálvektort tekintünk, akkor hipersíkokkal kialakítható partíciók számát jelölje $C(k,p)$.

3.1. segédtétel. $C(k,p) = 2\sum_{j=0}^{p-1}\binom{k-1}{j}$.

Bizonyítás. Nyilván $C(k,p) = C(k-1,p) + C(k-1,p-1)$, mert a k-adik hipersíkot az előző $k-1$ db hipersík éppen $C(k-1,p-1)$ partícióra bontja szét. Ugyanis p-dimenziós hipersíkok metszete is hipersík lesz, de a $p-1$ dimenziós altérben. Egyetlen hipersík 2 részre bontja a teret, tehát $C(1,p) = \begin{cases} 2 & \text{ha } p \geq 1 \\ 0 & \text{ha } p \leq 0 \end{cases}$. Ekkor $C(k,p) = C(k-1,p) + C(k-1,p-1) =$
$C(k-2,p) + 2C(k-2,p-1) + C(k-2,p-2) =$
$= C(k-3,p)+3C(k-3,p-1)+3C(k-3,p-3)+C(k-3,p-3) = \cdots =$
$= \sum_{j=0}^{s}\binom{s}{j}C(k-s,p-j) = \cdots = \sum_{j=0}^{k-1}\binom{k-1}{j}C(1,p-j) = 2\sum_{j=0}^{p-1}\binom{k-1}{j}$. ■

3.2. segédtétel. $C(k,p) \leq \sum_{i=0}^{p}\binom{k}{i} \leq k^p,\ k \geq p \geq 2$.

Bizonyítás.

$$C(k,p) = 2\sum_{j=1}^{p-1}\binom{k-1}{j} = \binom{k-1}{0} + \left[\binom{k-1}{0} + \binom{k-1}{1}\right] + \left[\binom{k-1}{1} + \binom{k-1}{2}\right] + \cdots$$
$$+ \left[\binom{k-1}{p-2} + \binom{k-1}{p-1}\right] + \binom{k-1}{p-1} = \binom{k}{0} + \binom{k}{1} + \cdots + \binom{k}{p-1} + \binom{k-1}{p-1} \leq \sum_{i=0}^{p}\binom{k}{i}.$$

Teljes indukcióval bizonyítunk. A $p = 2$-nél tetszőleges $k \geq 2$-ra igaz az állítás, hiszen $\sum_{i=0}^{2}\binom{k}{i} = 1 + k + \frac{k(k-1)}{2} = k^2 + \left(\frac{k+2-k^2}{2}\right) \leq k^2$. Ezután legyen $p \geq 2$ tetszőleges! A $k = p$ esetére nyílván igaz:

$$\sum_{i=0}^{p}\binom{p}{i} = 2^p \leq p^p, \text{ ha } p \geq 2.$$

Tegyük fel, hogy az egyenlőtlenség igaz valamely $k \geq p$-re: $\sum_{i=0}^{p}\binom{k}{i} \leq k^p$. Ekkor

$$\sum_{i=0}^{p}\binom{k+1}{i} = 2\sum_{i=0}^{p-1}\binom{k}{i} + \binom{k}{p} < 2k^{p-1} + k^p \leq pk^{p-1} + k^p \leq (k+1)^p. \blacksquare$$

Vegyük ki az i-edik tesztpontot, $\underline{x}_{m+i}$-t T_l-ből! Számoljuk ki minden (j,k) indexkombinációval az $\underline{N}_{jk}^{(i)} = \underline{t}(\underline{x}_j, \underline{x}_{m+i}) - \underline{t}(\underline{x}_k, \underline{x}_{m+i})$ vektorokat, ahol

$\underline{t}(\underline{x},\underline{u}) = (\rho(x_1,u_1),\rho(x_2,u_2),...,\rho(x_p,u_p))^T$ jelöli az $\underline{x},\underline{u} \in \mathbb{R}^p$ vektorok távolságvektorát. A rögzített $\underline{x}_{m+i}$-hez $\mathcal{D}_m$ tanulópontjaival $\binom{m}{2}$ ilyen vektor készíthető el, tehát ha mind az l tesztpontot sorravesszük, akkor $l \cdot \binom{m}{2}$ darab $\underline{N}_{jk}^{(i)}$ különbség-vektorunk lesz. Tekintsük most azokat a hipersíkokat $\mathbb{R}^p$ -ben, melyeknek az $\underline{N}_{jk}^{(i)}$ vektorok a normálvektorai! A hipersíkok egyenletei $\underline{N}_{jk}^{(i)T}\underline{A} = 0$ alakban adhatók meg, $\underline{A} \in \mathbb{R}^p$.

Ezen előkészületek után becslést tudunk adni a C_m halmaz l-edik darabolási együtthatójára.

3.3. segédtétel. $S(C_m,l) \leq \left(l\binom{m}{2}\right)^p$.

Bizonyítás. Jelölje $\mathcal{T}_l = \left\{\underline{x}_{m+1},\underline{x}_{m+2},...,\underline{x}_{m+l}\right\}$ a tesztpontok halmazát és legyen $\mathcal{F}_m = \left\{\underline{u} \in \mathbb{R}^p; \underline{u} = \left(\phi\left(\underline{x}_{m+1}\right),\phi\left(\underline{x}_{m+2}\right),...,\phi\left(\underline{x}_{m+l}\right)\right)^T,\ \phi \in C_m\right\}$. Először azt bizonyítjuk be, hogy $\mathcal{F}_m$ elemszáma nem lehet több, mint $\left(l\binom{m}{2}\right)^p$. Tekintsük minden (j,s) indexkombinációval az $\underline{N}_{j,s}^{(i)} = \underline{t}\left(\underline{x}_j,\underline{x}_{m+i}\right) - \underline{t}\left(\underline{x}_s,\underline{x}_{m+i}\right)$ vektorokat, ahol $\underline{t}(\underline{x},\underline{u}) = (\rho(x_1,u_1),\rho(x_2,u_2),...,\rho(x_p,u_p))^T$ jelöli az $\underline{x},\underline{u} \in \mathbb{R}^p$ vektorok távolságvektorát. (L. a **??** definíciót!) A képletben $\underline{x}_j$ és $\underline{x}_s$ a $\mathcal{D}_m$ tananyag két tetszőleges pontja. Ílymódon összesen $l\binom{m}{2}$ db ilyen $\underline{N}_{j,s}^{(i)} \in \mathbb{R}^p$ vektor készíthető el. Vegyük most az összes olyan hipersíkot $\mathbb{R}^p$-ben, melyeknek normálvektorai a kiszámolt $\underline{N}_{js}^{(i)}$ vektorok: $H_{j,s}^{(i)} = \left\{\underline{u} \in \mathbb{R}^p; \left(\underline{N}_{js}^{(i)}\right)^T\underline{u} = 0\right\}$. A 3.2. segédtétel szerint ez az összesen $k = l\binom{m}{2}$ db hipersík a teljes teret nem több mint $\left(l\binom{m}{2}\right)^p$ partícióra bontja. Ha most $\underline{a}_1$ és $\underline{a}_2$ nemnulla vektorok ugyanabból a partícióból valók, a hozzájuk tartozó $\phi_{\underline{a}_1},\phi_{\underline{a}_2} \in C_m$ döntésfüggvényekre fenn fog állni, hogy azonosan osztályozzák a $\mathcal{T}_l$ teszthalmaz pontjait, azaz $\phi_{\underline{a}_1}\left(\underline{x}_{m+i}\right) = \phi_{\underline{a}_2}\left(\underline{x}_{m+i}\right),\ i = 1,2,...,l$. A $\mathcal{F}_m$ halmaz elemszáma nem lehet tehát több, mint a partíciók száma, azaz $\left(l\binom{m}{2}\right)^p$.

Vegyünk most tetszőleges $\underline{z}_1,\underline{z}_2,...,\underline{z}_l \in \mathbb{R}^p$ tanulópontokat, és hozzátartozóan $v_1,v_2,...,v_l \in \{0,1\}$ tanításokat! Ezekkel a pontokkal fogjuk eljátszatni a tesztpontok szerepét, azaz tegyük fel egy pillanatra, hogy $\underline{z}_i = \underline{x}_{m+i}$ és $v_i = y_{m+i}$ $(i = 1,2,...,l)$. A korábban elmondott módon létrehozzuk $\mathbb{R}^p$ partícióit, melyek száma nem haladhatja meg $\left(l\binom{m}{2}\right)^p$-t. A C_m függvényosztályhoz tartozó $\mathcal{A}_{C_m}$ halmazrendszer most

$$\mathcal{A}_{C_m} = \left\{A_{\underline{a}};\ A_{\underline{a}} = \left\{\left\{\underline{x};\ \phi_{\underline{a}}(\underline{x}) = 1\right\} \times \{0\}\right\} \cup \left\{\left\{\underline{x};\ \phi_{\underline{a}}(\underline{x}) = 0\right\} \times \{1\}\right\},\ \underline{a} \in \mathbb{R}^p\right\}.$$

$S(C_m,l)$ azt a maximális részhalmaz számot jelenti, amit $\mathcal{A}_{C_m}$ kicsippent $Z = \{\underline{z}_1 \times v_1,\underline{z}_2 \times v_2,...,\underline{z}_l \times v_l\}$ részhalmazai közül. Ha most $\underline{a}_1$ és $\underline{a}_2$ nemnulla vektorok ugyanabból a partícióból valók, és

$A_{\underline{a}_i} = \left\{\left\{\underline{x};\ \phi_{\underline{a}_i}(\underline{x}) = 1\right\} \times \{0\}\right\} \cup \left\{\left\{\underline{x};\ \phi_{\underline{a}_i}(\underline{x}) = 0\right\} \times \{1\}\right\},\ i = 1,2,$

akkor $A_{\underline{a}_1} \cap Z = A_{\underline{a}_2} \cap Z$ lesz. Vagyis Z-ből nem lehet több részhalmazt kicsípni, mint ahány partíció alakult ki benn a hipersíkok révén, azaz $S(C_m, l) \leq \left(l\binom{m}{2}\right)^p$. ■

Ezek után rátérünk az 3.1 tétel bizonyítására.

Bizonyítás. Becslést adunk a

$$\mathbf{P}\left(L(\phi^*) - \inf_{\phi \in C_m} L(\phi) > \varepsilon \mid \mathcal{D}_m^*\right)$$

feltételes valószínűségre, ahol

$$L(\phi) = \mathbf{P}\left(\phi(\underline{X}) \neq Y \mid \mathcal{D}_m^*\right).$$

A Devroye-Györfi-Lugosi [7] könyv 8.2 lemmája szerint:

$$L(\phi^*) - \inf_{\phi \in C_m} L(\phi) \leq 2 \sup_{\phi \in C_m} \left|\hat{L}_{m,l}(\phi) - L(\phi)\right|.$$

A Devroye-Györfi-Lugosi [7] könyv 12.8 tétele szerint:

$$\mathbf{P}\left(\sup_{\phi \in C_m} \left|\hat{L}_{m,l}(\phi) - L(\phi)\right| > \varepsilon \mid \mathcal{D}_m^*\right) \leq 4e^8 S\left(C_m, l^2\right) e^{-2l\varepsilon^2}.$$

Ezekből az egyenlőtlenségekből közvetlenül következik, hogy

$$\mathbf{P}\left(L(\phi^*) - \inf_{\phi \in C_m} L(\phi) > \varepsilon \mid \mathcal{D}_m^*\right) \leq 4e^8 S\left(C_m, l^2\right) e^{-\frac{l\varepsilon^2}{2}}.$$

Végül felhasználva az l^2-edik darabolási együtthatóra vonatkozó becslést:

$$\mathbf{P}\left(L(\phi^*) - \inf_{\phi \in C_m} L(\phi) > \varepsilon \mid \mathcal{D}_m^*\right) \leq 4e^8 \left(l^2\binom{m}{2}\right)^p e^{-\frac{l\varepsilon^2}{2}}.$$

A fenti becslés már eloszlásmentes, a felső korlát nem függ a véletlentől. Ezzel a 3.1. tételt bebizonyítottuk. ■

3.2. megjegyzés. *A 3.1. tétel állításának jobboldala*

$$c_0 \exp\left(-l \cdot \left(c_1 + c_2 \frac{\ln l}{l} + c_3 \frac{\ln m}{l} + c_4 \frac{\ln(m-1)}{l}\right)\right)$$

alakú, ahol $c_0, ..., c_4$ *alkalmasan megválasztott konstansok. A* $l \to \infty$ *határátmenet során a kifejezés akkor lesz minimális, ha közben* $\frac{\log m}{l} \to 0$ *is fennáll. Ez utóbbi feltétel nyújt gyakorlati útmutatást arra vonatkozóan, hogy milyen arányban osszuk fel az adatainkat tananyag és teszthalmaz részekre. Tekintsük az* $f(x,y) = \frac{\ln(y-x)}{x} = \frac{\ln\left(\frac{y}{x}-1\right)}{x} + \frac{\ln x}{x}$ *függvényt! Megmutatható, hogy* $x = \frac{y}{\ln y}$ *választással* $y \to \infty$ *esetén* $x \to \infty$ *és* $f\left(\frac{y}{\ln y}, y\right) \to 0$. *Tehát az* $l \approx \frac{n}{\ln n}$, $m \approx n - \frac{n}{\ln n}$ *választás esetén a becslő kifejezés minimalizálható az* $n \to \infty$ *átmenetkor, azaz viszonylag kis elemszámú tesztanyag leválasztásával pontossá tehető a metrikasúlyozásos osztályozás. A 3.1. sz. táblázat azt mutatja, hogy* n *növekedtével hogyan alakul az* l *és* m *értéke, valamint az* $\frac{l}{m}$ *és* $\frac{l}{n}$ *arány.*

n	l	m	$\frac{l}{m}\%$	$\frac{l}{n}\%$
100	22	78	27.74	21.71
200	38	162	23.26	18.87
300	53	247	21.26	17.53
400	67	333	20.3	16.69
500	80	420	19.18	16.09
600	94	506	18.53	15.63
700	107	593	18.01	15.26
800	120	680	17.59	14.96
900	132	768	17.23	14.70
1000	145	855	16.93	14.48
2000	263	1737	15.15	13.16
3000	375	2625	14.27	12.50
4000	482	3518	13.71	12.06
5000	587	4413	13.30	11.74
10000	1086	8914	12.18	10.86
100000	8686	91314	9.51	8.69

3.1. sz. táblázat Tananyag, tesztanyag felosztás

3.3. A legjobban skálázott metrika kiválasztásának előkészítő fázisa

Ebben a szakaszban azt fogjuk tárgyalni, hogy hogyan lehet a g_m^* döntésfüggvényhez tartozó $\underline{A}^*$ skálázó vektort előállítani, azaz hogyan lehet kiválasztani az optimális metrikát a Δ metrika-családból.

A leírásnál szükségünk van néhány jelölésre. A tananyag két osztályra osztható:

$$M_0 = \{\underline{x}_i;\ \underline{x}_i \in \mathcal{D}_m^*,\ y_i = 0\} \text{ és } M_1 = \left\{\underline{x}_j;\ \underline{x}_j \in \mathcal{D}_m^*,\ y_j = 1\right\}.$$

Hasonlóképpen a teszthalmazt is két részre oszthatjuk:

$$O_0 = \{\underline{x}_i;\ \underline{x}_i \in \mathcal{D}_l^{**},\ y_i = 0\} \text{ és } O_1 = \left\{\underline{x}_j;\ \underline{x}_j \in \mathcal{D}_l^*,\ y_j = 1\right\}.$$

A dolgozatban használni fogjuk a vektorok és mátrixok között értelmezhető $\geq, \leq, <$ és $>$ féligrendezési relációkat. Az $\underline{x} \geq \underline{u}$ és $\underline{\underline{A}} \geq \underline{\underline{B}}$ $(\leq, >, <)$féligrendezési relációk akkor állnak fenn, ha minden koordinátában fennállnak.

3.4. definíció. $\underline{t}(\underline{x}, \underline{u}) = (\rho(x_1, u_1), \rho(x_2, u_2), ..., \rho(x_p, u_p))^T$, *az* $\underline{x}$ *és* $\underline{u}$ *távolságvektora.*

3.3. megjegyzés. $\underline{t}(\underline{x}, \underline{u}) \geq \underline{0}$, *és nyilvánvalóan igaz, hogyha* $\underline{t}(\underline{x}, \underline{u}) \leq \underline{t}(\underline{w}, \underline{v})$, *akkor tetszőleges* $\underline{A} \geq \underline{0}$ *skálázó vektorral* $d_{\underline{A}}(\underline{x}, \underline{u}) \leq d_{\underline{A}}(\underline{w}, \underline{v})$ *is fennáll.*

3.5. definíció. *A* $\underline{z} \in O_0$ *tesztpont távolságvektora az* M_1*-től:*

$$\underline{t}(M_1, \underline{z}) = \underline{t}^*(\underline{z}) = \left(\min_{\underline{u} \in M_1} \rho(z_1, u_1), \min_{\underline{u} \in M_1} \rho(z_2, u_2), ..., \min_{\underline{u} \in M_1} \rho(z_p, u_p)\right)^T$$

3.4. megjegyzés. *Hasonlóan lehet értelmezni a* $\underline{z} \in O_1$ *esetén a* $\underline{t}(M_0, \underline{z})$ *távolságvektort is.*

Az optimális $\underline{A}^* \geq \underline{0}$ skálázó vektort úgy fogjuk előállítani, hogy felírjuk azt egy $\underline{\underline{G}}\underline{A} \leq \underline{0}$, $\underline{A} \geq \underline{0}$, $\underline{A} \neq \underline{0}$ homogén lineáris egyenlőtlenség-rendszer megoldásaként, ahol a $\underline{\underline{G}}$ mátrix minden sorában van legalább egy negatív komponens. A $\mathcal{D}^{**}$ teszthalmaz minden eleméhez megkísérlünk előállítani egy vagy több sorvektort a $\underline{\underline{G}}$ mátrixban. Amelyik tesztpontnál ez nem sikerül, azokat rosszul fogja osztályozni még az optimális g_m^* döntésfüggvény is, de ezeket minden más $\phi_{\underline{A}} \in C_m$ döntésfüggvény is rosszul osztályozná. Az ismertetendő algoritmus két fázisban oldja meg a feladatot. Az első fázisban megadjuk a $\underline{\underline{G}}$ mátrixot, a másodikban pedig meghatározzuk a g_m^*-hoz tartozó $\underline{A}^* \geq \underline{0}$ skálázó vektort. Ebben a pontban az algoritmus első fázisát ismertetjük.

Felhasználjuk, hogy $\underline{z} \in O_0$-t tekintve, az alábbi három eset valamelyike biztosan fennáll:

1.eset: Van olyan $\underline{x} \in M_0$, melyre $\underline{t}(\underline{x}, \underline{z}) \leq \underline{t}^*(\underline{z})$ áll, és legalább egy koordinátánál szigorú a reláció. Ilyenkor tetszőleges $\underline{A}$ skálázó vektort használva $\underline{z}$-t jól fogjuk osztályozni, hiszen $\underline{A}^T \underline{t}(\underline{x}, \underline{z}) < \underline{A}^T \underline{t}^*(\underline{z}) < \underline{A}^T \underline{t}(\underline{u}, \underline{z})$, minden $\underline{u} \in M_1$-re, azaz $\phi_{\underline{A}}(\underline{z}) = 0$ lesz.

2. eset: Minden $\underline{x} \in M_0$ esetén $\underline{t}(\underline{x}, \underline{z}) > \underline{t}^*(\underline{z})$. Ekkor további két alesetet különböztetünk meg:

*2/***-a-** Létezik $\underline{u} \in M_1$, hogy minden $\underline{x} \in M_0$-re $\underline{t}(\underline{x}, \underline{z}) > \underline{t}(\underline{u}, \underline{z})$ is. Ilyenkor semmilyen $\underline{A}$ skálázó vektornál sem tudjuk $\underline{z}$-t jól osztályozni, hiszen $d_{\underline{A}}(\underline{u}, \underline{z}) \leq d_{\underline{A}}(\underline{x}, \underline{z})$, azaz $\phi_{\underline{A}}(\underline{z}) = 1$.

*2/***-b-** Minden $\underline{u} \in M_1$-hez létezik $\underline{x} \in M_0$, hogy $\underline{t}(\underline{x}, \underline{z}) \not> \underline{t}(\underline{u}, \underline{z})$. Ilyenkor az $(\underline{x}, \underline{u})$ párhoz kiszámítjuk a $\underline{g} = \underline{t}(\underline{x}, \underline{z}) - \underline{t}(\underline{u}, \underline{z})$-t, és felvesszük azt a $\underline{\underline{G}}$ mátrix sorvektorai közé. Ekkor majd $\underline{A}^*$-gal teljesülni fog, hogy minden $\underline{u} \in M_1$-hoz létezik $\underline{x} \in M_0 : d_{\underline{A}^*}(\underline{u}, \underline{z}) \geq d_{\underline{A}^*}(\underline{x}, \underline{z})$, azaz $\underline{z}$-t jól fogjuk osztályozni, mert $\phi_{\underline{A}^*}(\underline{z}) = 0$.

3. eset: Van olyan $\underline{x} \in M_0$, melyre $\underline{t}(\underline{x}, \underline{z}) \not> \underline{t}^*(\underline{z})$, de $\underline{t}(\underline{x}, \underline{z}) \not< \underline{t}^*(\underline{z})$ is.
Az ilyen $\underline{x} \in M_0$-ekre először maximalizáljuk a $\kappa(\underline{x}) = \sum_{i=1}^{p} I_{\{\rho(x_i, z_i) < t_i^*(\underline{z})\}}$ függvényt, majd a $\kappa(\underline{x})$-et maximalizáló $\underline{x}$-ekre minimalizáljuk a $\zeta(\underline{x}) = \sum_{i=1}^{p} \rho(x_i, z_i)$ függvényt. Jelöljük az optimumot felvevő tanulópontot $\underline{x}^*$-gal!
Ilyenkor $\underline{g} = \underline{t}(\underline{x}^*, \underline{z}) - \underline{t}^*(\underline{z})$-t felvesszük azt a $\underline{\underline{G}}$ mátrix sorvektorai közé, és $\underline{A}^{*T} \underline{t}(\underline{x}^*, \underline{z}) < \underline{A}^{*T} \underline{t}^*(\underline{z})$ miatt $d_{\underline{A}^*}(\underline{u}, \underline{z}) > d_{\underline{A}^*}(\underline{x}^*, \underline{z})$ lesz minden $\underline{u} \in M_1$-re, azaz $\underline{z}$-t jól fogjuk osztályozni.

Az algoritmus az első fázisban a fenti három eset figyelembevételével előállít egy $\underline{\underline{G}}$ mátrixot. A második fázisban megoldjuk a $\underline{\underline{G}}\underline{A} \leq \underline{0}$, $\underline{A} \geq \underline{0}$, $\underline{A} \neq \underline{0}$ lineáris programozási feladatot, melynek bármely $\underline{A}^*$ megoldása alkalmas skálázó vektor lesz a g_m^* döntésfüggvényhez. $\underline{G}$ sorainak s számára igaz az alábbi

egyenlőtlenség: $0 \leq s \leq \#M_1 \cdot \#O_0 + \#M_0 \cdot \#O_1$, ahol pl. $\#M_1 = \sum_{i=1}^{m} I_{\{y_i=1\}}$, az M_1 halmaz számossága. Az algoritmus az első fázisban meghatároz egy h^* értéket, amely azon tesztpontok számát adja meg melyekre a *2.***-a-** aleseténél leírt feltétel igaz. Ezeket egyik $\phi_{\underline{A}} \in C_m$ döntésfüggvénnyel sem lehet jól osztályozni, még g_m^*-gal sem. Igaz lesz továbbá az alábbi egyenlőtlenség:

$$\epsilon^* = \frac{h^*}{l} = \hat{L}_{m,l}\left(g_m^*\right) \leq \hat{L}_{m,l}\left(\phi_{\underline{A}}\right),$$

vagyis C_m-en g_m^* minimalizálja az $\hat{L}_{m,l}$ empirikus osztályozási hibát.

3.5. megjegyzés. *Ha az osztályok száma kettőnél nagyobb, azaz $\phi : \mathcal{X} \to \{1, 2, ..., L\}$ alakú, akkor az O_1 tesztosztályon az épp vizsgált tesztpont osztályától eltérő osztályhoz nem tartozó tesztpontok halmazát kell érteni. Ilymódon a leírt algoritmus kiterjeszthető az általánosabb esetre is! Második lépésben elhagyjuk az elsőnek kiválasztott osztály tanulópontjait. Az így redukált halmazban vesszük a második osztályt és ennek a tesztpontjai fogják játszani O_0 szerepét, a komplemens halmaz pedig O_1-ét. Az első lépésben megkezdett $\underline{\underline{G}}$ mátrix sorainak feltöltését az új szereposztásban folytatjuk, amíg az összes osztály feldolgozásra nem került.*

A korábban tárgyalt három eset figyelembevételével megadjuk az $\underline{A}^*$-ot előállító algoritmus első fázisát, azaz előállítjuk a $\underline{\underline{G}}$ mátrixot.

Az algoritmus leírásában használni fogjuk a := jelet, amelynek jelentése: a jobboldal tartalmával felülírjuk a baloldal tartalmát.

Inicializálás: $h^* := 0$.

i^0 Vegyük egyenként a T_l halmaz tesztpontjait! (1. ciklus). A leírásban jelölje $\underline{z} \in T_l$ az éppen soron következő tesztpontot! (Ha már minden tesztpontot feldolgoztunk, az iv^0 pontra ugrunk) Tegyük fel, hogy $\underline{z} \in O_0$. Legyen $s, b := 0$! Számoljuk ki $\underline{t}\left(M_1, \underline{z}\right) = \underline{t}^*\left(\underline{z}\right)$-t!

ii^0 Sorravesszük az M_0 osztály $\underline{x}$ tanulópontjait, ezzel elindítjuk a 2. (belső) ciklust.

ii^0*/-a-* Vegyük a következő $\underline{x} \in M_0$ tanulópontot, és számoljuk ki a $\underline{t}\left(\underline{x}, \underline{z}\right)$ távolságvektort! Ha már nincs több pont M_0-ban, az ii^0*/-e-* pontra ugrunk.

ii^0/-b- Megvizsgáljuk a $\underline{t}(\underline{x},\underline{z}) \leq \underline{t}^*(\underline{z})$ feltételt. Ha teljesül, akkor kilépünk a 2. ciklusból és az 1. ciklus végére lépünk, véve a következő $\underline{z}$ tesztpontot, azaz visszalépünk i^0-re. Ha nem teljesül, a ii^0/-c- pontra ugrunk.

ii^0/-c- Megvizsgáljuk a $\underline{t}(\underline{x},\underline{z}) \not> \underline{t}^*(\underline{z})$ relációt. Ha igaz, megvizsgáljuk, hogy $\underline{x}$ maximalizálja-e a $\kappa(\underline{x})$ és $\zeta(\underline{x})$ függvényeket. Ha igen, $\underline{x}^*$-ot $\underline{x}$-el felülírjuk, $s := s+1$. Visszatérünk az ii^0/-a- ponthoz.

ii^0/-d- Ha nem igaz, akkor Visszatérünk az ii^0/-a- ponthoz.

ii^0/-e- Megvizsgáljuk az $s = 0$ feltételt. Ha igaz, elindítunk egy 3. ciklust, azaz ráugrunk a iii^0-pontra. (Ilyenkor a fent tárgyalt 2. eset fordult elő!) Ha $s \geq 1$, akkor kiszámoljuk a $\underline{g} = \underline{t}(\underline{x}^*,\underline{z}) - \underline{t}^*(\underline{z})$-t és felvesszük a $\underline{\underline{G}}$ mátrix sorvektorai közé, $b := b+1$. (Vö. a 3. esetnél leírtakkal.) Ráugrunk i^0-ra.

iii^0 Sorravesszük az M_1 halmaz elemeit. Vegyük $\underline{u} \in M_1$-et! Elindítunk egy belső ciklust, ahol sorravesszük az $\underline{x} \in M_0$ tanulópontokat. Megvizsgáljuk, hogy fennáll-e $\underline{t}(\underline{x},\underline{z}) \not> \underline{t}(\underline{u},\underline{z})$.

Ha valamely $\underline{x}$-re ez teljesül, kiugrunk a belső ciklusból, kiszámoljuk $\underline{g} = \underline{t}(\underline{x},\underline{z}) - \underline{t}(\underline{u},\underline{z})$-t és tároljuk azt $\underline{\underline{G}}$-ben, $b := b+1$. Visszatérünk i^0-ra.

Ha a belső ciklus végigpörög és minden $\underline{x} \in M_0$-re $\underline{t}(\underline{x},\underline{z}) > \underline{t}(\underline{u},\underline{z})$ adódott, 1-gyel megnöveljük az h^* számláló tartalmát és csak ekkor térünk vissza i^0-ra.

iv^0 A feldolgozás befejeződött. A $\underline{\underline{G}}$ mátrix sorainak száma b, oszlopainak száma p. h^* tartalma a T_l-ben rosszul osztályozható pontok minimális száma.

3.4. Homogén lineáris egyenlőtlenség-rendszer nemnegatív megoldásának megkeresése

Belátható, hogy egy homogén lineáris egyenlőtlenségrendszer nemnegatív megoldásai egy véges konvex kúp elemeiként adhatók meg. Ebben a pontban ismertetendő eredményeket FARKAS GYULA klasszikus cikke [13] alapján közöljük.

Legyen $\underline{\underline{G}} \in \mathbb{R}^{s\times p}$ egy tetszőleges olyan mátrix amelynek minden sorában van negatív komponens, azaz létezik $g_{ij} < 0$ $(i = 1,2,...,s)$. Hogyan konstruáljunk meg olyan $\underline{x} \geq 0, \underline{x} \neq \underline{0}$ vektort, melyre $\underline{\underline{G}}\underline{x} \leq \underline{0}$ lesz?

Az alábbi tétel *Farkas Gyulától* származik. A konstruktív bizonyításban használt jelöléseket az optimálisan skálázott metrikát meghatározó algoritmus második fázisának leírásakor fel fogjuk használni.

3.2. tétel. *(Farkas) A* $\underline{Gx} \leq \underline{0}$, $\underline{x} \geq \underline{0}$, $\underline{x} \neq \underline{0}$ *feladat megoldható, és minden megoldás előáll* $\underline{x} = \underline{\underline{H}}_1\underline{\underline{H}}_2 \cdots \underline{\underline{H}}_s\underline{y}$, $\underline{y} \geq \underline{0}$, $\underline{y} \neq \underline{0}$ *alakban, ahol a* $\underline{\underline{H}}_1, \underline{\underline{H}}_2, \cdots, \underline{\underline{H}}_s \geq \underline{\underline{0}}$ *a bizonyítás során ismertetendő algoritmus által a* $\underline{\underline{G}}$ *komponenseiből megkonstruált mátrixok.*

Bizonyítás. *1. lépés:* Jelölje $\underline{g}_i$ a $\underline{\underline{G}}$ mátrix i-edik sorvektorát! Legyen $\underline{g}^{(0)} = \underline{g}_1 = (\gamma_1, \gamma_2, ..., \gamma_p)^T$. Tegyük fel az egyszerűség kedvéért, hogy $\underline{g}^{(0)}$ első r db komponense 0, az ezeket követő k db komponense negatív, az utolsó l db komponense pedig pozitív ($r + k + l = p$). Ezután definiáljuk a

$$\underline{\underline{H}}_1 = \left(\underline{e}_1, ..., \underline{e}_{r+k}, \underline{h}_1, ..., \underline{h}_{k \cdot l}\right) \in \mathbb{R}^{p \times (r+k+k \cdot l)}$$

mátrixot, ahol $\underline{e}_i$ az i-edik p-dimenziós egységvektor,

$$\underline{h}_1 = \gamma_{r+k+1}\underline{e}_{r+1} - \gamma_{r+1}\underline{e}_{r+k+1}, ..., \underline{h}_{k \cdot l} = \gamma_p\underline{e}_{r+k} - \gamma_{r+k}\underline{e}_p.$$

Tehát $\underline{\underline{H}}_1$ első $r + k$ oszlopa egységvektor, az utolsó $k \cdot l$ oszlopvektora pedig $\gamma_i\underline{e}_j - \gamma_j\underline{e}_i$ alakú, ahol $\gamma_i > 0$ és $\gamma_j < 0$. A konstrukcióból nyilvánvaló, hogy

$$\underline{\underline{H}}_1 \geq \underline{\underline{0}} \text{ és } \underline{g}^{(0)T}\underline{\underline{H}}_1 \left(\underbrace{0, ...0}_{r \text{ db}}, \gamma_{r+1}, \gamma_{r+2}, ..., \gamma_{r+k}, \underbrace{0, ...0}_{k \cdot l \text{ db}}\right) \leq 0.$$

2. lépés: Legyen

$$\underline{\underline{G}}_1 = \begin{pmatrix} \underline{g}_2^T \\ \vdots \\ \underline{g}_s^T \end{pmatrix} \cdot \underline{\underline{H}}_1.$$

Jelölje $\underline{g}^{(1)}$ a $\underline{\underline{G}}_1$ mátrix első sorának vektorát! Ismételjük meg $\underline{g}^{(1)}$-gyel az *1.* lépésben $\underline{g}^{(0)}$-val leírt eljárást, előállítva most a $\underline{\underline{H}}_2 \geq \underline{\underline{0}}$ mátrixot. A $\underline{\underline{H}}_2$ mátrix sorainak száma $r + k + kl$ lesz, az oszlopainak száma attól függ, mennyi 0, pozitív és negatív komponense van a $\underline{g}^{(1)}$-nek. Ezután legyen

$$\underline{\underline{G}}_2 = \begin{pmatrix} \underline{g}_3^T \\ \vdots \\ \underline{g}_s^T \end{pmatrix} \cdot H_2,$$

ennek $\underline{g}^{(2)}$ az első sorvektora.... A fenti eljárást ismételgetve $s-1$ lépésben eljutunk a $\underline{\underline{G}}_{s-1} = \underline{g}_s^T \underline{\underline{H}}_{s-1} = \underline{g}^{(s-1)}$ mátrixhoz, ami már sorvektor. Végül jelölje a $\underline{g}^{(s-1)}$-hez elkészített mátrixot $\underline{\underline{H}}_s$!

3. lépés: Legyen most $\underline{y} \geq \underline{0}$, $\underline{y} \neq \underline{0}$ tetszőleges!

$$\underline{\underline{G}}\,\underline{\underline{H}} = \underline{\underline{G}}\,\underline{\underline{H}}_1\underline{\underline{H}}_2 \cdots \underline{\underline{H}}_s = \begin{pmatrix} \underline{g}^{(0)T}\underline{\underline{H}}_1\underline{\underline{H}}_2 \cdots \underline{\underline{H}}_s \\ \underline{g}^{(1)T}\underline{\underline{H}}_2\underline{\underline{H}}_3 \cdots \underline{\underline{H}}_s \\ \vdots \\ \underline{g}^{(s-1)T}\underline{\underline{H}}_s \end{pmatrix} \leq \underline{0},$$

hiszen $\underline{g}^{(i)T}\underline{\underline{H}}_{i+1} \leq \underline{0}^T$ és $\underline{\underline{H}}_{i+2}\underline{\underline{H}}_{i+3} \cdots \underline{\underline{H}}_s \geq \underline{0}$ miatt a mátrix i-edik sorvektora nempozitív komponensű vektor lesz. Ekkor viszont minden $\underline{y} \geq \underline{0}$ vektorral $\underline{x} = \underline{\underline{H}}\,\underline{y}$-ra teljesül $\underline{\underline{G}}\,\underline{x} = \underline{\underline{G}}\,\underline{\underline{H}}\,\underline{y} \leq \underline{0}$! ■

3.6. megjegyzés. *a.) Minden $\underline{g}_i$ sorvektorhoz olyan $\underline{\underline{H}}_i$ mátrixot konstruálunk, hogy a $\underline{g}_i^T\underline{\underline{H}}_i$ sorvektor nemnulla komponensei éppen $\underline{g}_i$ negatív komponensei legyenek.*

b.) Az algorimus minden újabb sorvektor bevonásakor tovább tud lépni, hiszen $\underline{\underline{H}}_{i-1} \geq \underline{0}$ és $\underline{g}_i$-nek van negatív komponense a feltételek szerint. Így nyilván $\underline{g}_i^T\underline{\underline{H}}_{i-1} \leq \underline{0}^T$ lesz.

A *Farkas*-tétel alapján lehet az optimálisan skálázott metrikát kiválasztó algoritmus második fázisát leírni.

Inicializálás: $\underline{\underline{H}}_0, \underline{\underline{P}} := \underline{\underline{E}} \in \mathbb{R}^{p\times p}$, ahol $\underline{\underline{E}}$ a $p \times p$ -es egységmátrix.

i^0 Egyetlen ciklusban sorravesszük a $\underline{\underline{G}}$ sorvektorait. A leírásban az i-edik sorvektort $\underline{g}_i$-vel fogjuk jelölni. Ebből kiszámítjuk $\underline{w}_i^T = \underline{g}_i^T\underline{\underline{H}}_{i-1}$-et. Jelölje $\theta(1), \theta(2), ..., \theta(r)$ a $\underline{w}_i = (w_{i1}, w_{i2}, ..., w_{ip})^T$ vektor zérus komponenseinek indexét, $\omega(1), \omega(2), ..., \omega(k)$ a negatív, $\mu(1), \mu(2), ..., \mu(l)$ pedig a pozitív komponenseinek indexeit! Ekkor $\underline{w}_i$-komponenseiből előállítjuk a $\underline{\underline{H}}_i$ mátrixot, amelynek $h^{(i)}_{\alpha,\beta}$ komponenseire teljesül, hogy:

$$h_{\alpha,\beta}^{(i)} = \begin{cases} 1 & \text{ha} \quad \begin{cases} (1 \le \beta \le r \text{ és } \alpha = \theta(\beta)) \text{ vagy} \\ (r+1 \le \beta \le r+k \text{ és } \alpha = \theta(\beta - r)) \end{cases} \\ w_{i\gamma} & \text{ha} \quad \begin{cases} \text{valamely } j = 1,2,...,k\text{-ra:} \\ r+k+1+(j-1)\cdot l \le \beta \le r+k+1+j\cdot l \\ \alpha = \omega(j), \gamma = \mu(\beta - r - k) \end{cases} \\ -w_{i\gamma} & \text{ha} \quad \begin{cases} \text{valamely } j = 1,2,...,k\text{-ra:} \\ r+k+1+(j-1)\cdot l \le \beta \le r+k+1+j\cdot l \\ \alpha = \mu(\beta - r - k), \gamma = \omega(j) \end{cases} \\ 0 & \text{egyébként} \end{cases}$$

Végül: $\underline{\underline{P}} := \underline{\underline{P}} \cdot \underline{\underline{H}}_i$.

ii^0 Minden $\underline{x} \ge \underline{0}$ esetén a $\underline{\underline{P}}\underline{x}$ vektor egy optimális skálázó-vektort állít elő. Legyen pl. $\underline{A}^* = \underline{\underline{P}}\,\underline{1}$, ahol $\underline{1} = (1,1,...,1)^T$.

3.5. Metrikák keverése

Ebben a pontban általánosabban foglalkozunk az optimális metrika megválasztásával. Az alaptér most tetszőleges $\mathcal{X} \neq \emptyset$ halmaz lehet. Legyenek $d_1, d_2, ..., d_p$ különböző metrikák $\mathcal{X}$-hez. Ezekkel, mint bázismetrikákkal képezzük a

$$\Delta = \left\{ d_{\underline{A}};\ d_{\underline{A}}(x,u) = \sum_{i=1}^{p} A_i d_i(x,u),\ \underline{A} \ge \underline{0} \right\}$$

metrika-halmazt. Feladatunk az, hogy $\mathcal{D}_m^*$ és $\mathcal{D}_l^{**}$ segítségével olyan $\underline{0} \le \underline{A}^* = \underline{A}^*(\mathcal{D}_m^*, \mathcal{D}_l^{**})$ skálázó vektort megadjunk, amivel $d_{\underline{A}^*}$ minimalizálja az

$$\hat{L}_{m,l}(\phi_{\underline{A}}) = \frac{1}{l} \sum_{i=1}^{l} I_{\{\phi_{\underline{A}}(x_{m+i}) \neq y_{m+i}\}}$$

hibabecslést. Legyen most

$$\underline{t}(x,u) = (d_1(x,u), d_2(x,u), ..., d_p(x,u))^T$$

a távolságvektor és $z \in M_0$ esetén

$$\underline{t}\left(M_1, z\right) = \underline{t}^*\left(z\right) = \left(\min_{u \in M_1} d_1\left(z, u\right), \min_{u \in M_1} d_2\left(z, u\right), ..., \min_{u \in M_1} d_p\left(z, u\right) \right)^T .$$

Ezekkel minden megismételhető, amit az optimális súlyvektorok előállításáról a 3.3. és 3.4. szakaszban elmondtunk. Ezzel a technikával jó metrikákból keverhetünk ki még jobb metrikát. Ugyanis, ha $W_i \subset T_l$ jelöli a teszthalmaz azon részét, melyet a d_i metrikával a legközelebbi társ módszerrel jól osztályozzuk, akkor a $d_{\underline{A}^*}$ metrikával a jól osztályozható tesztpontok halmaza $\bigcup_{i=1}^{p} W_i$ lesz.

Összefoglalás

A dolgozat a legközelebbi társ módszerrel foglalkozik. Az az alapfeladat, hogy egy $x \in \mathcal{X}$ metrikus ponthoz megtaláljuk egy $\mathcal{T}_n \subset \mathcal{X}$ véges elemszámú halmaz legközelebbi elemét. A témakört speciális szempontból tárgyaljuk: hogyan lehet a legkevesebb távolságszámítással a feladatot megoldani? Milyen előkészítő lépésekkel lehet segíteni ezt a célt? Hogyan célszerű a metrikát megválasztani?

A legfontosabb célkitűzés tehát a keresés költségeinek leszorítása úgy, hogy közben az osztályozási pontosság ne növekedjék jelentősen.

A dolgozat három fejezetre tagozódik.

Az **első fejezet** témája a *tananyag szerkesztése.*

Ha a $\mathcal{D}_n$ tananyag olyan $\mathcal{D}_m^*$ részhalmazát állítjuk elő, mellyel a $\mathcal{D}_n$ tanulópontjai $\mathcal{D}_m^*$-gal pontosan osztályozhatók lesznek, *ritkításról* beszélünk. Összefoglaljuk az irodalomban javasolt ritkító algoritmusokat. Újdonságként megadjuk *Tomek* idegen-pontpárokat szerkesztő algoritmusának általános metrikus térben is alkalmazható változatát.

Ha az n elemszámú $\mathcal{D}_n$ tananyaggal azonos szerkezetű, kisebb m elemszámú új $\mathcal{P}_m^*$ tananyagot állítunk elő, mellyel $\mathcal{D}_n$ pontjai pontosan osztályozhatók lesznek, p*rototípushalmaz-szerkesztés*ről van szó. Összefoglaljuk az irodalmi ajánlásokat, és megadunk egy vadonatúj eljárást metrikus térben. (L. Prototípushalmaz szerkesztése segéd ponthalmazzal az 1.2.3. pontban).

Ha az osztályozás pontosságát úgy kívánjuk javítani, hogy osztályai közül néhányat egybevonunk, néhányat feldarabolunk, *osztály-átdefiniálásról* beszélünk. A tananyag átszerkesztésének ez a szempontja ennek a dolgozatnak az eredménye. (L. 1.3. pontot!)

A pontosságot úgy is lehet javítani, ha előzetesen detektáljuk az osztályok szélsőséges tulajdonságú tanulópontjait, az osztályok „fekete bárányait", a *deviáns* (vagy outlier) tanulópontokat, melyek az osztályozások során a hibás besorolások többségét okozhatják. A deviáns tanulópontok definícióit az 1.4. pontban adjuk meg, ami szintén ennek a dolgozatnak a terméke. Ha a deviáns pontokat másképpen vesszük figyelembe a döntésben, akkor azt mondjuk, hogy a tananyagot *megszűrtük.*

A dolgozat **második fejezetének** témája a tanulópontok gyors megkeresése.

A metrikus térben a háromszög-egyenlőtlenségeken alapuló kizárási feltételeket fogalmazhatunk meg, melyeket beépítve a keresés folyamatába gyorsabban eljuthatunk a legközelebbi társhoz.

A dolgozatban öt kizárási feltételt fogalmazunk meg, melyek a következők:

2.1. TÉTEL: A $t \in \mathcal{T}_n \setminus \{t_{i_1}, t_{i_2}, ..., t_{i_k}\}$ tanulópont biztosan nem legközelebbi szomszédja x-nek, ha az alábbi kizárási feltételek valamelyike igaz rá:

$$\mathcal{K}_1: \qquad 2d_{\min}^{(k)} < d\left(t, t_{NN}^{(k)}\right);$$

$$\mathcal{K}_2: \qquad 0 < \max_{j=1,\dots,k} \left(d\left(t, t_{i_j}\right) - 2d\left(x, t_{i_j}\right)\right);$$

$$\mathcal{K}_3: \qquad d_{\min}^{(k)} < \max_{j=1,\dots,k} \left|d\left(t, t_{i_j}\right) - d\left(x, t_{i_j}\right)\right|;$$

$$\mathcal{K}_4: \qquad d_{\max}^{(k)} - d_{\min}^{(k)} > d\left(t_{FN}^{(k)}, t\right);$$

$$\mathcal{K}_5: \qquad 2d_{\min}^{(k)} < d\left(t, t_{NN}^{(k)}\right) \text{ vagy } d_{\max}^{(k)} - d_{\min}^{(k)} > d\left(t_{FN}^{(k)}, t\right);$$

ahol $d_{\min}^{(k)} = \min_{j=1,\dots,k} d(x, t_{i_j}), d_{\max}^{(k)} = \max_{j=1,\dots,k} d(x, t_{i_j}),$

$\mathrm{t}_{NN}^{(k)} = \arg \min_{j=1,\dots,k} d(x, t_{i_j}), t_{FN}^{(k)} = \arg \max_{j=1,\dots,k} d(x, t_{i_j}).$

A felsorolt kizárási feltételek közül csak $\mathcal{K}_3$-nak vannak előzményei az irodalomban, a többi új eredmény.

A 2.2 és 2.3 tételek megadják a kizárási feltételek között meglévő erőkapcsolatot:

2.2. TÉTEL: $\mathcal{K}_i \Longrightarrow \mathcal{K}_j$, ha $(i, j) \in \{(1,2), (2,3), (4,3), (5,3)\}$, ahol a $\Longrightarrow$arra utal, ha $\mathcal{K}_i$ kizár egy $x \in \mathcal{X}$ pontot, akkor $\mathcal{K}_j$ is kizárja azt.

2.3. TÉTEL: $\mathcal{K}_i \not\Longrightarrow \mathcal{K}_j$, ha
$(i, j) \in \{(3,2), (3,1), (2,1), (3,4), (3,5), (2,4), (4,2), (1,4), (4,1)\}$.

A 2.4 és 2.5 tételekben a $\mathcal{K}_1$ kizárási elv két olyan tulajdonságát tárgyaljuk, ami a kizárásos kereső stratégiák megalkotásánál játszanak szerepet:

2.4. TÉTEL: A $\mathcal{K}_1$ kizárási feltétel antiszimmetrikus és nem tranzitív. (Ha a kizárja b-t, akkor b nem zárhatja ki a-t. Viszont létezhetnek olyan a, b, c tanulópontok, hogy bár a kizárja b-t, b c-t, de a nem zárja ki c-t.)

2.5. TÉTEL: Ha $3d(x,a) < d(a,b)$ és $3d(x,b) < d(b,c)$, akkor $3d(x,a) < d(a,c)$.

A kizárási feltételek függenek a már feldolgozott tanulópontoktól. fontos tehát, hogy jó sorrendben dolgozzuk fel a tanulópontokat. A feldolgozás sorrendjét meghatározó elvet *keresési stratégiá*nak nevezzük. A lehetséges keresési stratégiákat a 2.5. pontban tárgyaljuk. A kísérleti futtatások alapján a minimax kereső stratégia tűnik a legjobbnak.

Nemcsak pontonként, hanem *csoportonkénti kizárási elv* is érvényesíthető, ha előzőleg megfelelően struktúráljuk a $\mathcal{T}_n$ tanulópont halmazt. A *lefedő gömb* fogalma segítségével definiáljuk a *szeparálhatóság* fogalmát.

2.1. DEFINÍCIÓ Legyen $A \subset \mathcal{T}_n$. A $G(c,R) = \{x;\ x \in \mathcal{X},\ d(c,x) \leq R\}$ c centrumú, R sugarú zárt gömb az A-t *lefedő gömb*, ha $c \in A$ és $A \subset G(c,R)$. Az A-t lefedő zárt gömbök között azt a $G_A(c_A, R_A)$-val jelöltet fogjuk *legkisebb lefedő zárt gömb*nek nevezni, amelynél sugár minimális. A c_A az A halmaz *centruma*, R_A pedig a *takaró-sugár*.

2.2. DEFINÍCIÓ: Legyenek $A_1, A_2 \subset \mathcal{T}_n$. Jelölje rendre c_1, c_2 a centrumaikat, és R_1, R_2 a takaró-sugaraikat. Az A_1, A_2 halmazok *szeparálhatóak*, ha $d(c_1,c_2) > R_1 + R_2 + \max\{R_1, R_2\}$.

Szeparált halmazok esetén lehetőség nyílik a kereséskor csoportos kizárásra.

2.6. TÉTEL: Legyenek $A_1, A_2 \subset \mathcal{T}_n$ szeparálhatóak. Ha $x \in \mathcal{X}$-re teljesül, hogy $d(x,c_1) < R_1$, akkor $\min\limits_{z \in A_1} d(x,z) < \min\limits_{y \in A_2} d(x,y)$.

2.7. TÉTEL: Legyenek $A_1, A_2 \subset \mathcal{T}_n$ szeparálhatóak. Ha $x \in \mathcal{X}$-re teljesül, hogy $d(x,c_1) + R_2 < d(x,c_2)$, akkor $t_{NN}(x) \notin A_2$.

2.8. TÉTEL: Tegyük fel, hogy már kiszámoltuk a $d(x,t_1), d(x,t_2), ..., d(x,t_k)$ távolságokat. Legyen $d_{\min}^{(k)} = \min\limits_{i=1,...,k} d(x,t_i)$, és $A \subset \mathcal{T}_n \setminus \{t_1, ..., t_k\}$.
Ha $d_{\min}^{(k)} + R_A < d(x,c_A)$, akkor $t_{NN}(x) \notin A$.

Az irodalmi példák mellett a 2.4. pontban bemutatunk néhány csoportos kizárással dolgozó NN-kereső algoritmust.

A 2.7. pontban foglalkozunk a kizárásos kereső stratégiák hatékonyságának jellemzésével. Megadunk egy olyan algoritmust, amely a tanulópontokat úgy rendezi át adott x osztályozandó pont ismeretében, hogy a legközelebbi társ megkeresésének lépésszáma minimális legyen. Ehhez a minimális lépésszámhoz lehet összehasonlítani egy tesztelendő NN-kereső stratégia lépésszámát. Ezt az összevetést meg is tesszük két kizárásos NN-kereső stratégia esetében, amikor is 30 számítógéppel szimulált esetben elvégezzük különböző beállítások mellett a keresést. (L. 2.7. pontot és a függelléket!)

A **harmadik fejezet** tárgya a metrikák optimális átskálázása. Alkalmas súlyok segítségével hogyan lehet pontosabb osztályozást eredményező metrikát kalibrálni? Az átkalibrált metrikával hogyan változik meg az NN-keresés sebessége? Ezeket a kérdéseket tisztázzuk ebben a fejezetben. A legfontosabb eredményt a 3.1. tételben mondjuk ki.

3.1. Tétel: Az $\hat{L}_{m,l}(\phi) = \frac{1}{l}\sum_{i=1}^{l} I_{\{\phi(\underline{x}_{m+i})\neq y_{m+i}\}}$ empirikus hibavalószínűséget minimalizáló $g_m^* = \arg\min_{\phi_{\underline{A}}\in C_m} \hat{L}_{m,l}\left(\phi_{\underline{A}}\right)$ döntésfüggvényre minden $\varepsilon > 0$ esetén fennáll, hogy

$$\mathbf{P}\left(L(g_m^*) - \inf_{\phi\in C_m} L(\phi) > \varepsilon \mid \mathcal{D}_m\right) \leq 4e^8\left(l^2\binom{m}{2}\right)^p e^{-\frac{l\varepsilon^2}{2}}, \text{ ahol}$$

$L(\phi) = \mathbf{P}(\phi(\underline{x}) \neq y \mid \mathcal{D}_m)$ és C_m jelöli a skálázott metrikákhoz tartozó NN-döntésfüggvények halmazát.

A fenti tétel segítségével meg lehet adni azt, hogy a súlyvektor előállításakor a tananyag-tesztanyag elemszám $m : l$ aránya hogyan választandó meg.

Az optimális átskálázáshoz alkalmazott súlyvektor meghatározása egy lineáris programozási feladat megoldásával történik. A feladat felállításához szükséges együtthatómátrix komponenseinek előállítása ennek a dolgozatnak az eredménye. A felállított egyenlőtlenségrendszer megoldása *Farkas Gyula* klasszikus módszerével történik.

A fejezet végén megmutatjuk, hogy a súlyvektort előállító algoritmus alkalmazható metrikák keveréséhez felhasználható együtthatórendszer előállítására is.

Summary

This essay deals with Nearest Neighbour Decision Rule. The aim is here to detect the nearest element in the finite $\mathcal{T}_n \subset \mathcal{X}$ set to a given $x \in \mathcal{X}$ metrical point. We discuss this topic from a special point of view: how possible to solve the searching problem with minimal distance calculation? Which types of the preprocessing steps are needed to reach this aim? How should be choosen the suitable metric? The most important purpose is here to reduce the cost meanwhile the accuracy of the classification does not increase significantly.

The essay consists of three chapters. The topic of the first chapter is the *edition of the training* set $\mathcal{D}_n$. If we create a subset $\mathcal{D}_m^* \subset \mathcal{D}_n$ which classifies correctly the training points in $\mathcal{D}_n \setminus \mathcal{D}_m^*$, then we *reduce the training set.* First we give a summary of the reducing methods proposed in the literature. Then we give the extension of the *Tomek*'s Foreign Point Pair method in metric space.

If we create a new $\mathcal{P}_m$ training set with the same structure of the original $\mathcal{D}_n$ training set, where the points in $\mathcal{D}_n$ classifiable correctly by the points of $\mathcal{P}_m$ and m far less then n, we *create a prototype set* $\mathcal{P}_m$. We give an overlook of the published solutions of this problem. We propose a new construction method of the prototype set in metric space. (See 1.2.3. subsection.)

The third type of the training set edition is the *redefinition of the classes.* This solution of the edition is a new result of this paper. If we merge into a common class more classes where the training points are too close together, and we split up the classes into new classes where the training points disperse significantly in the space, we can improve the accuracy of the classification. Some applications need such type of edition. (See 1.3. section).

We can enhance the accuracy of the classification by detection of the deviant points in the training set. There are several possibilities to give the definition of the deviant points, which are given in the 1.4. section. The deviant points in a class of the training set have extremely different statistical properties as the other (typical) training points in the same class. If we filter the training set from such deviant elements we can reduce the number of the false classifications.

The topic of the second chapter is the fast search of the nearest neighbour. On the base of the triangular inequality we can state excluding assumptions.

Suppose, that we know the distances $d(x,t_{i_j}), j=1,2,...,k$ yet. The point $t \in \mathcal{T}_n \setminus \{t_{i_1}, t_{i_2}, ..., t_{i_k}\}$ will not be the nearest neighbour of the query point $x \in \mathcal{X}$, if at least one of the next five excluding assumptions holds:

$$\mathcal{K}_1: \qquad 2d_{\min}^{(k)} < d\left(t, t_{NN}^{(k)}\right);$$

$$\mathcal{K}_2: \qquad 0 < \max_{j=1,...,k} \left(d\left(t,t_{i_j}\right) - 2d\left(x,t_{i_j}\right)\right);$$

$$\mathcal{K}_3: \qquad d_{\min}^{(k)} < \max_{j=1,...,k} \left|d\left(t,t_{i_j}\right) - d\left(x,t_{i_j}\right)\right|;$$

$$\mathcal{K}_4: \qquad d_{\max}^{(k)} - d_{\min}^{(k)} > d\left(t_{FN}^{(k)}, t\right);$$

$$\mathcal{K}_5: \qquad 2d_{\min}^{(k)} < d\left(t, t_{NN}^{(k)}\right) \text{ or } d_{\max}^{(k)} - d_{\min}^{(k)} > d\left(t_{FN}^{(k)}, t\right);$$

where $d_{\min}^{(k)} = \min\limits_{j=1,...,k} d(x,t_{i_j}), d_{\max}^{(k)} = \max\limits_{j=1,...,k} d(x,t_{i_j})$,

$t_{NN}^{(k)} = \arg \min\limits_{j=1,...,k} d(x,t_{i_j}), t_{FN}^{(k)} = \arg \max\limits_{j=1,...,k} d(x,t_{i_j})$.

The next two theorems show the power connections among the excluding assumptions:

2.2. THEOREM: $\mathcal{K}_i \Longrightarrow \mathcal{K}_j$, if $(i,j) \in \{(1,2),(2,3),(4,3),(5,3)\}$, where the arrow $\Longrightarrow$means that if $\mathcal{K}_i$ excludes an $x \in \mathcal{X}$ point, then $\mathcal{K}_j$ also excludes it.

2.3. THEOREM: $\mathcal{K}_i \not\Longrightarrow \mathcal{K}_j$, if
$(i,j) \in \{(3,2),(3,1),(2,1),(3,4),(3,5),(2,4),(4,2),(1,4),(4,1)\}$.

The excluding assumption $\mathcal{K}_3$ is the strongest among them. In the next two theorems we characterize the simplest excluding assumption $\mathcal{K}_1$.

2.4. THEOREM: The excluding assumption $\mathcal{K}_1$ is unsymmetrical and non transitive. (If a excludes b, then b doesn't exclude a. But exist such a,b,c points in the metric space, where a excludes b, b excludes c, but a doesn't exclude c.)

2.5. THEOREM: If $3d(x,a) < d(a,b)$ and $3d(x,b) < d(b,c)$, then $3d(x,a) < d(a,c)$.

The excluding assumptions depend on the order of the training points. The searching strategies determine the order of the searching. The essay deals with the searching strategies in the subsection 2.5. Based on the computer program running the so called *"minimax"* searching strategy seems to be the best one. Here the next training point $t \in \mathcal{T}_n$ will be the one which

$t = \arg \min_{u \in \mathcal{T}_n \setminus \{t_1,...,t_k\}} \max_{i=1,...,k} |d(x,t_i) - d(u,t_i)|$.

After suitable restructuring of the training set we can execute grouped exclusion. We can omit whole subsets of $\mathcal{T}_n$ from the searching process. Using the concepts of the minimal covering sphere and separable sets we can create such type of excluding searching algorithms.

2.1. Definition: Let A be a subset of $\mathcal{T}_n$. The sphere $G(c,R) = \{x; x \in \mathcal{X}, d(c,x) \leq R\}$ *covers* the set A if $c \in A$ and $A \subset G(c,R)$. A covering sphere is minimal if does not exist such covering sphere which has smaller radii. The centre element of the minimal covering sphere is the centrum of A and the radii of this sphere is the covering radii of A. *Notations:* c_A and R_A.

2.2. Definition: The subsets $A, B \subset \mathcal{T}_n$ are separable, if $d(c_A, c_B) > R_A + R_B + \max\{R_A, R_B\}$.

The next three theorems shows how possible to use the concept of the separable sets for the grouped exclusion.

2.6. Theorem: Let $A, B \subset \mathcal{T}_n$ be separable sets. If $d(x,c_A) < R_A$, then $\min_{z \in A} d(x,z) < \min_{z \in B} d(x,y)$.

2.7. Theorem: Let $A, B \subset \mathcal{T}_n$ be separable sets. If $d(x,c_A) + R_B < d(x,c_B)$ then $t_{NN}(x) \notin B$, where $t_{NN}(x)$ the nearest neighbour of x in $\mathcal{T}_n$.

2.8. Theorem: Suppose that we have calculated the distances $d(x,t_i), i = 1,...,k$, and $d^{(k)}_{\min} = \min_{i=1,...,k} d(x,t_i)$. If $A \subset \mathcal{T}_n \setminus \{t_1, t_2, ..., t_k\}$ and $d^{(k)}_{\min} + R_A < d(x,c_A)$, then $t_{NN}(x) \notin A$.

In the essay we describe some NN-searching processes which execute grouped exclusion. In the subsection 2.7. we give the principle of the characterizing of the searching strategies in the point of view of effectiveness. We delineate an algorithm which produce the *minimal spacer set* in $\mathcal{T}_n$ if the query point x is given.

2.3. Definition: $B \subset \mathcal{T}_n$ is a *spacer set* of x in $\mathcal{T}_n$ if after calculating the distances $d(x,t), t \in B$ the applied excluding assumption $\mathcal{K}$ will exclude every

other training point from the searching. B_0 is a *minimal spacer set* if no other spacer set B exsists which has fewer elements then B_0.

We can compare the number of steps of an searching strategy with the number of the elements in B_0. That's way we can characterize the effectiveness of the searching strategy empirically. We demonstrate it by 30 computer simulations of several NN-searching algorithms.

In the third chapter we deal with the optimal scaling of the metric function. How possible to set the metric function by suitable weights in order to get more accurate classification? How will change the speed of the search with the new metric? The most important result of this part is presented in the 3.1 theorem. Split $\mathcal{T}_n$ into subsets $\mathcal{A}_m$ and $\mathcal{B}_l$. (m and l show the number of elements in $\mathcal{A}_m$ and $\mathcal{B}_l$.) After creating a weight vector $\underline{w}$ elaborating the subset $\mathcal{A}_m$., let's denote by $\phi_{\underline{w}}$ the NN-decision function which uses the normalized metric with weight $\underline{w}$.

$\hat{L}_{m,l}(\phi) = \frac{1}{l}\sum_{i=1}^{l} I_{\left\{\phi\left(\underline{x}_{m+i}\right)\neq y_{m+i}\right\}}$ is the empirical error of classification. $g_m^* = \arg\min_{\phi_{\underline{w}}\in C_m} \hat{L}_{m,l}\left(\phi_{\underline{w}}\right)$.

3.1. THEOREM: For every $\varepsilon > 0$

$$\mathbf{P}\left(L\left(g_m^*\right) - \inf_{\phi\in C_m} L(\phi) > \varepsilon \mid \mathcal{A}_m\right) \leq 4e^8\left(l^2\binom{m}{2}\right)^p e^{-\frac{l\varepsilon^2}{2}},$$

where $L(\phi) = \mathbf{P}\left(\phi\left(\underline{x}\right) \neq y \mid \mathcal{A}_m\right)$ the theoretical error of classification.

We can give the optimal $m : l$ splitting ratio using the statement of the previous theorem. Otherwise the determination of the optimal weight vector $\underline{w}^*$ is made by solving a classical linear programming problem. Originally it was solved by *Gyula Farkas*. The proposed method also applicable at the metric mixture. We can determine the optimal mixing weights to the given metrics.

FÜGGELLÉKEK

F1. Függellék

A kísérleti futtatásoknál felhasznált tanulóponthalmaz.

100 *db független* $N(0,1)$ *eloszlásból származó kétdimenziós vektor.*

n	x	y	n	x	y	n	x	y	n	x	y
1	-0,91	1,74	26	1,36	-0,22	51	-1,19	0,57	76	0,87	0,56
2	-0,04	-0,7	27	-0,21	0,27	52	-2,39	-0,77	77	-0,48	-0,51
3	-0,28	-1,28	28	-0,32	0,53	53	0,79	-0,22	78	-0,9	0,97
4	-0,36	0,43	29	0,7	-2,62	54	1,08	-1,17	79	-1,99	2,36
5	-1,86	0,24	30	-0,98	-0,36	55	0,41	-1,27	80	0,76	-1,87
6	-1,77	-1,01	31	0	0,24	56	-0,97	-1,77	81	-0,41	0,62
7	-0,32	1,06	32	1,06	0,12	57	-0,45	-0,77	82	-0,83	-1,2
8	1,63	-0,18	33	-1,38	0,87	58	2,35	-0,51	83	-0,87	-0,8
9	-0,19	-0,56	34	-0,28	-1,56	59	-0,67	0,14	84	-2,04	-0,04
10	-0,32	1,45	35	-0,78	-0,16	60	-0,49	0,73	85	-0,48	0,13
11	-0,34	0,68	36	-0,31	0,29	61	1,06	1,03	86	-1,96	-0,3
12	-1,16	0,02	37	-1,14	0,41	62	0,97	-0,46	87	-1,72	-1,35
13	1,43	0,84	38	1,15	1,4	63	-1,11	-1,16	88	0,37	-0,6
14	-0,87	1,52	39	1,35	-1,16	64	1,76	0,76	89	0,71	1,24
15	-0,45	-0,24	40	-1,22	0,56	65	0,48	0,95	90	-0,56	0,07
16	-1,79	-1,63	41	-0,03	0,23	66	-0,66	-1,46	91	0,38	2,17
17	-1,52	-0,58	42	0,4	0,04	67	-0,11	0,68	92	1,68	-0,44
18	1,01	0,79	43	0,03	1,52	68	1,14	-0,1	93	0,08	-1,14
19	0,74	-0,48	44	0,41	-0,14	69	1,04	-0,35	94	0,74	-0,78
20	0,58	-1,54	45	0,13	-0,47	70	-2,13	0,34	95	-1,33	0,98
21	0,01	-0,47	46	-0,32	-1,5	71	1,71	1,42	96	-1,04	1,55
22	-0,88	0,22	47	0,83	0,33	72	-0,44	-1,4	97	-1,03	0,07
23	1,52	-1,26	48	0,07	-3,06	73	0,46	1,4	98	-0,23	-0,73
24	-0,27	-0,52	49	0,06	0,41	74	-0,08	1,18	99	0,26	0,66
25	-0,01	0,43	50	0,31	0,49	75	-1,4	2,05	100	-1,26	-0,44

F2. Függellék

A véletlen sorrendű keresésnél használt permutációk.

INDEX	1	2	3	4	5	6	7	8	9	10
PERM1	24	56	49	41	6	8	45	96	53	46
PERM2	97	26	12	68	60	20	87	45	29	93
INDEX	11	12	13	14	15	16	17	18	19	20
PERM1	42	17	94	27	38	7	10	85	78	75
PERM2	77	50	81	95	42	5	28	80	33	7
INDEX	21	22	23	24	25	26	27	28	29	30
PERM1	60	26	95	50	58	93	52	43	76	22
PERM2	34	57	14	30	67	44	61	70	2	39
INDEX	31	32	33	34	35	36	37	38	39	40
PERM1	59	87	12	48	30	47	18	91	92	15
PERM2	59	54	82	6	46	62	65	90	18	71
INDEX	41	42	43	44	45	46	47	48	49	50
PERM1	57	70	61	71	65	44	82	63	62	67
PERM2	58	51	94	47	35	8	63	1	66	69
INDEX	51	52	53	54	55	56	57	58	59	60
PERM1	16	1	81	89	72	23	37	100	31	34
PERM2	73	24	43	16	13	4	23	32	56	78
INDEX	61	62	63	64	65	66	67	68	69	70
PERM1	88	84	19	99	74	32	54	90	86	2
PERM2	86	36	17	79	83	9	76	48	40	64
INDEX	71	72	73	74	75	76	77	78	79	80
PERM1	98	39	73	55	11	83	36	25	4	80
PERM2	92	10	91	88	98	72	31	84	100	3
INDEX	81	82	83	84	85	86	87	88	89	90
PERM1	40	29	28	3	35	5	9	68	77	33
PERM2	74	15	21	49	55	41	11	27	89	52
INDEX	91	92	93	94	95	96	97	98	99	100
PERM1	69	97	64	79	13	20	21	51	66	14
PERM2	99	37	19	22	85	96	53	25	75	38

F3. Függelék

Futtatási jegyzőkönyv

Optimális janicsárhalmazok (zárójelben a maghoz tartozó pontok állnak)

Metrika	Osztályozandó pont	Optimális janicsárhalmaz	Elemszám
d_E	$(0,0)$	$\{(t_{41}), t_2\}$	2
d_E	$(3,-2)$	$\{(t_{58}), t_{33}, t_{34}\}$	3
d_{Cs}	$(0,0)$	$\{(t_{41}), t_{63}, t_5\}$	3
d_{Cs}	$(3,-2)$	$\{(t_{23}), t_5, t_{38}\}$	3
d_M	$(0,0)$	$\{(t_{31}), t_1\}$	2
d_M	$(3,-2)$	$\{(t_{58}), t_1, t_{29}\}$	3

1. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{35}	1,966	0,061	
2. t_{27}	0,796	0,176	
3. t_{41}	0,228	0,213	NN!
4. t_{31}	0,237	0,219	
5. t_{49}	0,413	0,321	STOP!

2. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{41}	0,228	0,023	NN
2. t_{49}	0,413	0,059	
3. t_{36}	0,425	0,19	
4. t_{28}	0,62	0,225	
5. t_{85}	0,493	0,231	STOP!!!

3. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{79}	3,087	0,066	FN
2. t_{52}	2,512	0,18	
3. t_{31}	0,237	0,20474	
4. t_{41}	0,228	0,257	NN, STOP!!!

4. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{24}	0,579982	0,579982	↓
2. t_{49}	0,412736	0,412736	↓
3. t_{91}	0,228467	0,228467	↓ NN, STOP!!

5. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{97}	1,035154	1,035154	
2. t_{12}	1,160535	1,035154	
3. t_{60}	0,485789	0,875075	↓
4. t_{45}	0,485789	0,485789	↓
5. t_{77}	0,69996	0,485789	
6. t_{42}	0,401752	0,401752	↓
7. t_{36}	0,42498	0,401752	
8. t_{31}	0,23734	0,23734	↓
9. t_{41}	0,228467	0,228467	↓ NN, STOP!!

6. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_1	5,412157	0,509091	
2. t_{48}	3,120573	1,453571	
3. t_{58}	1,628378	1,554772	NN
4. t_{23}	1,659257	1,628378	STOP!!

7. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{58}	1,628378	0,001281	NN
2. t_{94}	2,3979	0,399145	
3. t_{72}	3,651948	0,624278	
4. t_{38}	3,870639	1,628378	STOP!!

8. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{79}	6,626131	0,825804	FN
2. t_{48}	3,120573	1,427317	
3. t_{58}	1,628378	1,588908	NN
4. t_{23}	1,659257	1,628378	STOP'!!

9. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{24}	3,587986	3,587986	
2. t_{56}	3,976665	3,587986	
3. t_{49}	3,800941	3,587986	
4. t_{41}	3,762712	3,587986	
5. t_{7}	4,869298	3,587986	
6. t_{9}	2,274819	2,274819	↓
7. t_{58}	1,628378	1,628378	↓, NN, STOP!!

10. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{97}	4,531853	4,531853	
2. t_{26}	2,418049	2,418049	↓
3. t_{58}	2,657723	2,418049	
4. t_{20}	2,458774	2,418049	
5. t_{29}	2,385336	2,385336	↓
6. t_{80}	2,240754	2,240754	↓
7. t_{61}	3,601019	2,240754	
8. t_{39}	1,849078	1,849078	↓
9. t_{58}	1,628378	1,628378	↓, NN, STOP!!

11. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{1}	1,741875	0,005177	
2. t_{47}	0,82648	0,122389	
3. t_{89}	1,23194	0,225976	
4. t_{45}	0,225976	0,353056	NN, STOP!!

12. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{45}	0,225976	0,006744	NN
2. t_2	0,701567	0,138765	
3. t_{48}	0,406591	0,222782	
4. t_{29}	2,615479	1,076715	STOP!!

13. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{48}	3,06301	0,021424	FN
2. t_{12}	1,160337	0,225976	
3. t_{41}	0,225976		NN, STOP!!

14. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{24}	0,515207	0,515207	↓
2. t_{49}	0,408427	0,408427	↓
3. t_{41}	0,225976	0,225976	↓ NN
4. t_{42}	0,399786	0,225976	
5. t_{44}	0,406591	0,225976	STOP!!

15. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{97}	1,032979	1,032979	
2. t_{12}	1,160337	1,032979	↓
3. t_{60}	0,728186	0,728186	↓
4. t_{45}	0,468847	0,468847	↓
5. t_{42}	0,399786	0,399786	↓
6. t_{36}	0,314549	0,314549	↓
7. t_{31}	0,237316	0,237316	↓
8. t_{41}	0,225976	0,225976	↓ NN, STOP!!

16. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_1	3,910219	0,299288	
2. t_{80}	2,236926	0,503752	
3. t_{56}	3,90097	0,653501	
4. t_{58}	1,491489	1,238886	
5. t_{64}	2,758505	1,483284	
6. t_{23}	1,483284		NN, STOP!!

17. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{23}	1,483284	0,012332	NN
2. t_{21}	2,985788	0,393568	
3. t_{48}	2,933937	0,653508	
4. t_{58}	1,491489	1,238886	
5. t_{64}	2,758508	1,3318357	
6. t_{74}	3,183571	1,55755	STOP!!

18. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{70}	5,125766	1,238886	FN
2. t_{64}	2,758505	1,483284	
3. t_{23}	1,483284		STOP!!

19. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{24}	3,266348	3,266348	
2. t_{56}	3,90097	3,266348	
3. t_{49}	2,940515	2,940515	↓
4. t_{41}	3,03365	2,940515	
5. t_{8}	1,818938	1,818938	↓
6. t_{96}	4,036381	1,818938	
7. t_{26}	1,780187	1,780187	↓
8. t_{58}	1,491489	1,491489	↓
9. t_{71}	3,416466	1,491489	
10. t_{23}	1,483284	1,483284	↓ NN, STOP!!

20. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{97}	4,032979	4,032979	
2. t_{26}	1,780187	1,780187	↓
3. t_{29}	2,304563	1,780187	
4. t_{39}	1,645318	1,645318	↓
5. t_{58}	1,491489	1,491489	↓
6. t_{23}	1,483284	1,483284	↓ NN
7. t_{64}	2,758508	1,483284	STOP!!

21. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{1}	2,652094	0,26263	
2. t_{61}	2,093901	0,04076	
3. t_{31}	0,24076		NN, STOP!!

22. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{31}	0,24076	0,003437	NN
2. t_{27}	0,484956	0,073934	
3. t_{49}	0,467912	0,24076	STOP!!

23. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{79}	4,349748	0,02663	FN
2. t_{61}	2,093901	0,24076	
3. t_{31}	0,24076		STOP!!

24. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{24}	0,78155	0,78155	
2. t_{49}	0,467912	0,467912	↓
3. t_{41}	0,259626	0,259626	↓
4. t_{27}	0,484956	0,259626	
5. t_{31}	0,24076	0,24076	↓ NN STOP!!

25. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{97}	1,100048	1,100048	
2. t_{12}	1,181762	1,100048	
3. t_{60}	1,213471	1,100048	
4. t_{45}	0,59602	0,59602	↓
5. t_{77}	0,989149	0,59602	
6. t_{42}	0,439476	0,439476	↓
7. t_{31}	0,24076	0,24076	↓ NN, STOP!!

26. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{1}	4,335646	1,689084	
2. t_{29}	2,920042	1,1870926	
3. t_{48}	3,996947	2,144997	
4. t_{58}	2,144997		NN, STOP!!!

27. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{58}	2,144997	0,082883	NN
2. t_{88}	4,027606	0,677156	
3. t_{71}	4,706615	1,689084	
4. t_{29}	2,920042	1,891767	
5. t_{12}	6,181762	2,144997	STOP!!

28. futás:

A feldolgozás sorrendje	A kiszámított távolság	A minimax keresőérték	Megjegyzés
1. t_{79}	9,34748	1,689084	FN
2. t_{29}	2,920042	1,870926	
3. t_{48}	3,996947	2,144997	
4. t_{58}	2,144997		NN, STOP!!!

29. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{24}	4,751141	4,751141	
2. t_{56}	4,198559	4,198559	↓
3. t_{49}	5,348942	4,198559	
4. t_{6}	5,759642	4,198559	
5. t_{8}	3,18505	3,18505	↓
6. t_{96}	7,585359	3,18505	
7. t_{58}	2,144997	2,144997	↓ NN
8. t_{29}	2,920042	2,144997	STOP!!

30. futás:

A feldolgozás sorrendje	A kiszámított távolság	Pillanatnyi min távolság	Megjegyzés
1. t_{97}	6,100048	6,100048	
2. t_{26}	3,416615	3,416615	↓
3. t_{20}	2,872296	2,872296	
4. t_{29}	2,920042	2,872296	
5. t_{80}	2,367869	2,367869	↓
6. t_{71}	4,706615	2,367869	
7. t_{58}	2,144997	2,144997	↓ NN, STOP!!

Irodalomjegyzék

[1] Brown, T.A. and Koplowitz, J.: The Weighted Nearest Neighbor Rule for Class Dependent Sample Sizes, *IEEE Transactions on Information Theory,* **5** (1979) 617-620.

[2] Chang, C.L.: *Finding Prototypes for Nearest Neighbour Classifiers,* IEEE Transactions On Computers, Nov. 1974. Volc-23, No 11, pp **1179-1184.**

[3] Cover, T.: Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition., *IEEE Transactions on Electronic Computers, Vol.14.,* (1965) 326-334.

[4] Cover, T. and Hart, P.: Nearest Neighbor Pattern Classification, *IEEE Transactions on Information Theory*, **13** (1967) **21-27.**

[5] Djouadi, A.,Bouktache, E.: A Fast Algorithm for the Nearest-neighbor Classifier, *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.19.,* (1997) 277-282.

[6] Devijever, P.A., Kittler, J.: *Pattern Recognition: A Statistical Approach,* Prentice/Hall International, 1981.

[7] Devroye,L.: On the Inequality of Cover and Hart in Nearest Neighbor Discrimination, *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. Pami-3.,* (1981) 75-78.

[8] Devroye,L., Györfi,L., Lugosi,G.: *A Probabilistic Theory of Pattern Recognition*, (Springer-Verlag, New-York), (1996)

[9] Duda, R.O. and Hart, P.: Pattern Classification and Sceene Analysis, (Wiley, New York), (1974) ch 4.

[10] Dudani, S.A.: *The Distance-Weighted k-Nearest-Neighbor Rule*, IEEE Transactions on Systems Man and Cybernetics,1976,4,pp. **325.-327.**

[11] FARAGÓ, A.-LINDER,T.-PIKLER, T.-LUGOSI,G.: *A legközelebbi szomszéd osztályozási módszer algoritmikus problémáiról*, Hiradástechnika, XXXIX. évf. 8.,1988, 337.-341. old.

[12] FARAGÓ, A.-LINDER,T., LUGOSI,G.: *Fast Nearest Neighbor Search in Dissimilarity Spaces*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 15., No. 9., september 1993.,pp. 1.-5.

[13] FARKAS, J.: Über die Theorie der einfachen Ungleichugen, *Journal für die reine und angewandte Mathematik* **124** (1902) 1-27.

[14] FRIEDMANN, J.H., BASKETT, F., SHUSTEK,L.J.: An Algorithm for Finding Nearest Neighbors, *IEEE Transactions on Computers, Vol 24,* (1975), 1000-1006.

[15] FUKUNAGA, K.-NARENDRA,P.M.: *A Branch and Bound Algorithm for Computing k-Nearest Neighbors*, IEEE Transactions on Computer, july 1975,pp 750-753.

[16] FUKUNAGA, K.: Introduction to Statistical Pattern Recognition, (Acedemic Press, New York), (1976) ch 6.

[17] GÁBOR, GY.: Az NN alakfelismerési módszer, *Doktori értekezés,* (1975)

[18] GATES, G.W.: *The reduced Nearest Neighbour Rule* IEEE Trans.Inform.Theory (Corresp.), May 1972,pp **431-433.**

[19] HART, P.E.: *The Condensed Nearest Neighbour Rule,* IEEE Trans.Inform.Theory (Corresp.),May 1968. vol. IT-14,pp **515-516.**

[20] HATTORI, K., TAKAHASHI, M.: A New Edited k-Nearest Neighbor Rule in the Pattern Classification Problem, *Pattern recognition, Vol. 33,* (2000), **521-528.**

[21] HUANG, Y.S., CHIANG, C.C., SHIEH, J.W., GRIMSON, E.: Prototype Optimization for Nearest-Neighbor Classification, *Pattern recognition, Vol 35,* (2002), **1237-1245.**

[22] KETSKEMÉTY L.: *Statistical methods for Preparing the Training Set,* ZAMM, 70 (1990) 6, pp. **733.-735.**

[23] KIM, B.S., PARK, S.B..: Fast k Nearest Neighbor Finding Algorithm Based on the Ordered Partition, *IEEE Transactionson Pattern Analysis and Machine Intelligence, Vol. Pami-8,* (1986) 761-766.

[24] LEE, E-W.,CHE, S-I: Fast Design of Reduced-Complexity Nearest Neighbor Classifiers Using Triangular Inequality, *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.20.,* (1998) 562-566.

[25] PARADES, R.,VIDAL,E.: A Class-dependent Weighted Dissimilarity Measure for Nearest Neighbor Classification Problems, *Pattern Recognition Letters, Vol 21,* (2000), 1027-1036.

[26] RAMASUBRAMANIAN,V., KULDIF K. PALIWAL: Fast Nearest-neighbor Search Algorithms Based on Approximation-elimination Search, *Pattern Recognition, Vol 33,* (2000),1497-1510.

[27] RICCI, F.,AVESANI, P.: Data Compression and Local Metrics for Nearest-Neighbor Classication, *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.21.,* (1999) 380-384.

[28] RITTER, G.L., WOODRUF,H.B.,LOWRY, S.R., ISENHOW,T.L.: *An Algorithm for a Selective Nearest Neighbour Decision Rule*, IEEE Trans. On Inf.Theory,Nov. 1975, pp **665-669.**

[29] SALZBERG, S.,DELCHER, A.L., HEATH, D.,KASIF, S.: Best-case Results for Nearest-Neighbor Learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.17.,* (1995) 599-608.

[30] SETHI, I.K.: *A Fast Algorithm for Recognizing Nearest Neighbors*, IEEE Transactions on System, Man and Cybernetics, Vol. SMC 11., No. 3., March 1981, pp. 245-248.

[31] SHORT, R.D. AND FUKUNAGA, K.: The Optimal Distance Measure for Nearest Neighbor Classification, *IEEE Transactions on Information Theory,* **5** (1981) 622-627.

[32] STONE, C.: Consistent Nonparametric Regression, *Annals of Statistics*, **8** (1977) **1348-1360.**

[33] TOMEK, I.: *Two modifications of CNN*, IEEE Transactions On Systems, Man And Cybernetics,Nov.1976., pp **769-772.**

[34] ULLMAN, J.R.: *Automatic Selection of Reference Data for Use in a Nearest Neighbour* Method of Pattern Classification, IEEE Trans.Inform.Theory,July 1974, pp **541-543.**

[35] VIDAL, E.: *An Algorithm Finding Nearest Neighbors in (approximately) Constant Average Time*, Pattern Recognition Letters, Vol. 4., 1986, pp. 145-157.

[36] WILSON, D.L.: *Asymptotic Properties of Nearest Neighbor Rules Using Edited Data*, IEEE Transactions on Systems Man and Cybernetics, 1972,2,pp. **408.-421.**

[37] ZHANG,H., SUN, G.: Optimal Reference Subset for Nearest Neighbor Classification by Tabu Search, *Pattern recognition, Vol 35,* (2001), 1481-1490.

A szerző egyéb publikációi

[38] KETSKEMÉTY L.: Nearest Neighbor Classifying with Excluding Assumption, XXI. Seminar of Stability Problems of stochastic Models, Eger, 2001. 28 januar-3 februar.

[39] KETSKEMÉTY L.: A legközelebbi szomszéd gyors megkeresése, Alkalmazott Matematikai Lapok, **21**, (2003)

[40] KETSKEMÉTY L.: A legközelebbi szomszéd módszer hatékonysága automatikus skálázás esetén, Alkalmazott Matematikai Lapok, **21**,(2003)

[41] KETSKEMÉTY L.: Digitális műholdképek számítógépes klaszterezése, egyetemi oktori disszertáció, 1984, KLTE.

[42] GULYÁS O., KETSKEMÉTY L., KORÁNDI M.: Többsávos műholdképek számítógépes klaszterezése, Időjárás,88.vf., No.3.,(1984), 161.-173.

[43] ASZMUSZ,V.V., VADÁSZ V., KARASZEV, A.B., KETSKEMÉTY L.: Adaptivnaja avtomatizirovannaja szisztema inventarizacii szelszkohozjajsztvennüh kultur po koszmicseszkim szmjimkam, Iszledovanija Zemli iz Koszmosza, No.6, (1987), 79-88

[44] ASZMUSZ,V.V., VADÁSZ V., KARASZEV, A.B., KETSKEMÉTY L., SZPIRIMIDONOV,J.G.: Programnüj komplex klaszterizacii mnogozonalnüh dannüh Iszledovanija Zemli iz Koszmosza, No 3, (1988), 86-93

[45] TÄNCZER T., KETSKEMÉTY L., LÉVAI G.: Kísérlet a borultsági viszonyok műholdas meghatározására, Időjárás, 92.évf.,4.sz.,(1988), 242-250

[46] TÄNCZER T., KETSKEMÉTY L., LÉVAI G. Attempt for Estimation of Cloud Amounts whithin Satellite Pixels in Visible Spectrum, Adv.Space Res.,Vol.9.,No. 7., (1989), 181-184.

[47] KETSKEMÉTY L.: Statistical methods for Preparing the Training Set ZAMM, 70 (1990) 6, 733-735

[48] GULYÁS, O., BARTA GY., SZÁSZ, G., KETSKEMÉTY, L., PRÖHLE, K.: Alakfelismerési módszerek alkalmazása a minőségbiztosításban, Tanulmány (az OMFB támogatásával), (1995)

[49] GULYÁS, O., BARTA GY., SZÁSZ, G., KETSKEMÉTY, L., PRÖHLE, K.: Alakfelismerési programrendszer minőségbiztosítási alkalmazásokhoz, Programdokumentáció (az OMFB támogatásával), (1996)

A tananyag előfeldolgozásai a legközelebbi társ módszerhez

Értekezés a doktori (PhD) fokozat megszerzése érdekében a matematika tudományában.

írta: Ketskeméty László okleveles matematikus, matematika tanár

Készült a Debreceni Egyetem Matematika doktori programja (Valószínűségelmélet és matematikai statisztika alprogramja) keretében

Témavezető: Dr. Fazekas István

Elfogadásra javaslom: 2003.

A jelölt a doktori szigorlatot 2002. május 24-én eredményesen letette.

A bizottság elnöke: Dr. Pap Gyula

.....

Az értekezést bírálóként elfogadásra javaslom:

Dr. ....

Dr. ....

Dr. ....

A jelölt az értekezést 200 . ...-n sikeresen megvédte:

A bírálóbizottság elnöke: Dr. ....

A bírálóbizotság tagjai:

Dr. ....

Dr. ....

Dr. ....

Debrecen – 2003