Eötvös Loránd Tudományegyetem

Természettudományi Kar

Az automatikus hierarchikus osztályozás elmélete és gyakorlata

Matematikai Elemző BSc

Szakdolgozat

Készítette: Mihucza Gergely

Témavezető Pröhle Tamás

Matematikai Intézet, Valószínűségelméleti és Statisztikai Tanszék

Budapest, Magyarország

2019

Összefoglaló

A dolgozatomban az automatikus osztályozás módszerével foglalkozom. Az automatikus osztályozás alapesetben egy olyan adatmodellt hoz létre, amely egy vagy több magyarázó változó mellett egy magyarázott változóra vonatkozik. A módszer során felhasznált összes változó diszkrét lehetséges értékű, az eredménye pedig a rendelkezésre álló megfigyelések egy hierarchikus csoportokra bontása. Az eredmény formája egy döntési fa, amely alapján a rendelkezésre álló és az esetleges új megfigyelések diszkrét csoportokba sorolhatók. E csoportok olyanok, hogy a célváltozó értékének eloszlása szempontjából azok a megfigyelések amelyek egy csoportba kerülnek egyformának, amelyek pedig különbözőbe kerülnek, azok pedig eltérőknek tekinthetők.

A bontás hierarchikussága azt jelenti, hogy a csoportok egy hierarchikus rendszerben, lépésenként állnak elő. Az eljárás az első lépéseként kiválasztja azt a magyarázóváltozót, amelynek lehetséges értékei szerint a célváltozó feltételes eloszlásai a bontás célfüggvénye szerint egyrészt a keletkező csoportokon belül a leghasonlóbbak, másrészt viszont a csoportok közt tekintve a lehető legkülönbözőbbek. Ezután az eljárás a megfigyeléseket két vagy több csoportra bontja a kiválasztott változó megfelelő értékei szerint. Az eljárás a következő lépéseiben mindig az előző lépések szerint keletkezett csoportokat bontja tovább, ugyanezen elv alapján.

Dolgozatomban egy olyan módszer matematikai hátterével, algoritmikus részleteivel és alkalmazási gyakorlatával foglalkozom, amelynél a csoportosítás alapja a kétdimenziós gyakoriságtáblák elemzése. A felhasznált célfüggvény a χ^2 távolság. A csoportosítás pedig a felhasznált célfüggvény alapján automatikus.

Tartalomjegyzék

1. Homogén csoportokra bontás

A dolgozatomnak ebben a részében az optimális csoportosítás Fisher által vizsgált problémájával foglalkozom. Bemutatom e probléma egy speciális, χ^2 távolságra érvényes változatát.

A módszer lényeges eleme egy-egy magyarázó változó lehetséges értékeinek optimális csoportosítása, amivel először dokumentáltan WD. Fisher foglalkozott. [1]

Egy adott N elemű $\mathbb{R}$-beli halmaz legjobb K részhalmazra való felbontásának azt a felbontást tekintjük, ahol a négyzetösszeg távolság a legkisebb. Nevezetesen a:

$$D = \sum_{i=0}^{N} w_i (a_i - \overline{a}_i)^2$$

minimális. Itt N a csoportosítandó elemek száma, az a_i az i-dik csoportosítandó elemet, w_i az i.-dik elemhez tartozó súlyt és $\overline{a}_i$ az i.-dik elemet tartalmazó részhalmazban szereplő elemek átlagát jelöli.

1.1. A másodfajú Stirling számok

A legegyszerűbb, de nem a leggyorsabb algoritmusa a minimum keresésnek az, ha az összes lehetőséget végignézve találjuk meg a legjobb felbontást. Esetünkben ehhez egy N elemű halmaz összes olyan K osztályba sorolását kell megkeresnünk, amely esetén egyetlen osztály sem üres.

A másodfajú Stirling szám adja meg azon lehetőségek számát, ahányféle képpen egy N elemű halmaz K osztályba sorolható. A másodfajú Stirling szám értéke:

$$S(N,K) = \tfrac{1}{K!} \sum_{J=0}^{K} (-1)^{K-J} \binom{K}{J} J^N$$

Megmutatjuk, hogy tényleg az $S(N,K)$ adja meg a keresett osztályozás számot.

Bizonyítás

Értelmezzünk egy f szürjektív függvényt, ami az n elemű halmaz minden eleméhez hozzárendel egy $1..k$ számot, ami az őt tartalmazó halmaz sorszáma. A k számot $k!$ féleképpen tudjuk úgy hozzárendelni az n elemhez, hogy a halmaz elemei azonosok, csak a hozzájuk rendelt k szám változik. Ezeket az eseteket nem különböztetjük meg.

Az n elemhez k számot az n^k féleképpen lehet hozzárendelni, ezzel megengedve az üres halmazokat is figyelmen kívül hagyva a szürjektív kikötést. Ebből kifolyólag az összes esetből ki kell vonni azokat az eseteket, amikor van olyan i eleme k, ami nincs hozzárendelve az n elemű halmaz egyik eleméhez sem. Ezek számát a logikai szita formulával kapjuk meg. A k halmazok számából kiválasztunk elsőnek 1-et amihez nem rendeljün hozzá n egyik elemét sem.

Ennek a száma $\binom{n}{k}$ szorozva ahányféleképpen a fennmaradó (k-1)-et hozzátudjuk rendelni az n halmaz eleméhez. Majd így tovább.

1.1.1. A rekurzió és további összefüggések

Az $S(N, K)$ rekurzív értelmezéséhez egyenként helyezzük el az N elemet a K csoportba. Amikor az utolsó, az N. kerül elhelyezésre, akkor két eset lehetséges. Vagy még csak $K-1$ nem üres csoport van, ekkor az utolsó elemnek egy új csoportba kell kerülnie. Vagypedig már van K meglévő, nem üres csoport, és ekkor az új, utolsó elemnek a K csoport egyikébe kell kerülnie. Tehát:

$$S(N, K) = S(N-1, K-1) + K \cdot S(N-1, K)$$

Az alábbi képletek alkalmasak arra, hogy – szükség esetén – megfelelő rekurzív eljárásokat szerkesszünk egy N elemű halmaz K részalmazra való felbontásainak

felsorolására.

$$\begin{aligned} S(N+1,K+1) &= \sum_{J=K}^{N} \binom{N}{J} \cdot S(J,K) \\ S(N+1,K+1) &= \sum_{J=K}^{N} (K+1)^{N-J} \cdot S(J,K) \\ S(N+K+1,K) &= \sum_{J=0}^{K} J \cdot S(N+J,J) \end{aligned}$$

Az alábbi két egyenlőtlenség a másodfajú Stirling szám nagyságrendjét adja meg.

$$\frac{1}{2} \cdot \binom{n}{k} \cdot k^{n-k} \quad \geq \quad S(n,k) \quad \geq \quad \frac{1}{2} \cdot (k^2+k+2) \cdot k^{n-k-1} - 1$$

Látható, hogy a lehetséges felbontások száma igen gyorsan növekszik. Ez a vizsgált feladatunk körében azt jelenti, hogy nagyobb magyarázó változó szám esetén komoly végrehajtási nehézségek jelentkeznek, ha az optimális felbontást az összes lehetséges felbontás vizsgálatval keressük.

1.2. A felbontás konvexitása

Megmutatjuk, hogy egy optimális felbontás szükségszerűen konvex halmazokat eredményez. Vegyük észre, hogy a minimalizálandó távolság ekvivalens módon a

$$D = \sum_{i=0}^{N} w_i (a_i - \overline{a}_i)^2 = \sum_{i=0}^{N} w_i a_i^2 - \sum_{i=0}^{N} w_i \overline{a}_i^2$$

formába írható. Ezt a formát megkaphatjuk azonos átalakításokkal. De könnyebben következik a Pythagorasz tételből, felhasználva hogy az $(w_1\overline{a}_1, ..., w_N\overline{a}_N)$ vektor a egy merőleges vetülete a $(w_1a_1, ..., w_Na_N)$ vektornak.

Tegyük fel, hogy az aktuális felbontás nem konvex. Azaz létezik három olyan

$$a_i < a_k < a_j$$

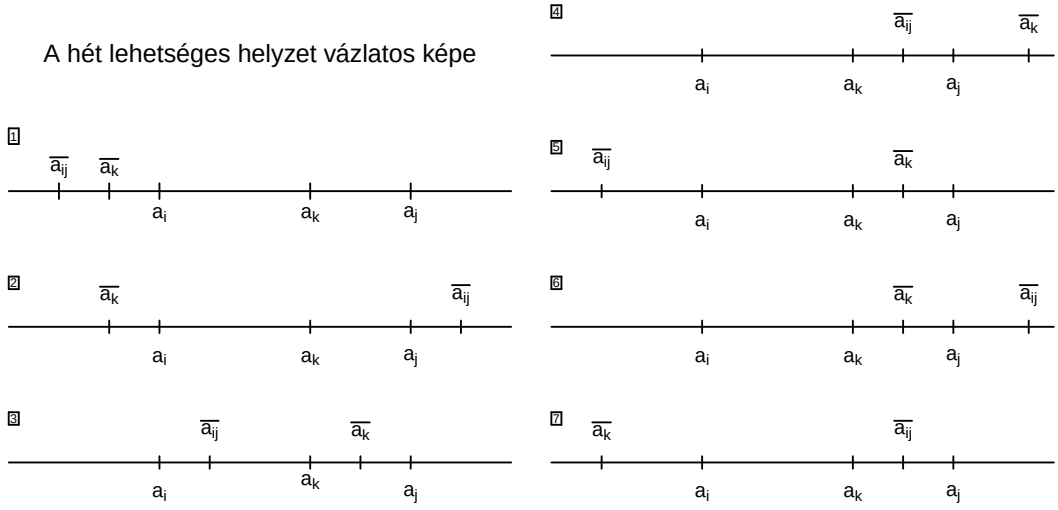
elem, ahol az a_i és az a_j azonos, az a_k pedig egy másik csoporthoz tartozik. Ekkor a három közül legalább az egyik távolabb van a saját halmazának az átlagától mint a másik halmaz átlagától. Azaz, ha $\overline{a}_{ij}$ annak a csoportnak az átlaga amelybe az a_i és az a_j tartozik és $\overline{a}_k$ annak a csoportnak az átlaga amelyhez az a_k, akkor az alábbi három egyenlőtlenség közül

$$\begin{aligned} |a_k - \overline{a}_k| &\geq |a_k - \overline{a}_{ij} \\ |a_i - \overline{a}_{ij}| &\geq |a_i - \overline{a}_k \\ |a_j - \overline{a}_{ij}| &\geq |a_j - \overline{a}_k \end{aligned}$$

legalább egy teljesül.

Ugyanis a a_i, a_j és az a_k szempontjából hét különböző típusú eset lehetséges aszerint, hogy a két átlag az a_i-hez és az a_j-hez viszonyítva hogyan helyezkedik el. 1.) Ha mindkét átlag az (a_i, a_j) intervallumon kívül, azonos félegyenesen helyezkedik el, akkor mindhárom ponthoz ugyanaz az átlag van közelebb. 2.) Ha mindkét átlag az (a_i, a_j) intervallumon kívül van, például úgy, hogy az $\overline{a}_k < a_i < a_k < a_j < \overline{a}_{ij}$ teljesül, akkor ahhoz, hogy teljesüljön, hogy az a_k-hoz az $\overline{a}_k$ átlag, az a_i-hez pedig az $\overline{a}_{ij}$ átlag legyen a közelebbi egyrészt annak, hogy $a_i - \overline{a}_k < \overline{a}_{ij} - a_k$ másrészt pedig annak, hogy $\overline{a}_{ij} - a_k < a_i - \overline{a}_k$ kell teljesülnie. Ez pedig ellentmond egymásnak. Tehát az a_k és az a_i közül az egyik nem a saját átlagához van a legközelebb. 3.) Ha mindkét átlag az (a_i, a_j) intervallumon belüli, akkor az intervallum két végpontjához nem lehet ugyanaz az átlag a legközelebbi. 4.) Ha az $\overline{a}_k$ az (a_i, a_j) intervallumon belüli és az $\overline{a}_{ij}$ kívüli úgy, hogy $a_j < \overline{a}_{ij}$, akkor szükségszerű, hogy a a_i-hez az $\overline{a}_k$ közelebb legyen mint az $\overline{a}_{ij}$. 5.) Ha az előbbi esetben $\overline{a}_{ij}$ úgy van kívül, hogy $\overline{a}_{ij} < a_i$ akkor meg a_j van közelebb $\overline{a}_k$-hoz mint $\overline{a}_{ij}$-hez. 6.) Ha az $\overline{a}_{ij}$ az (a_i, a_j) intervallumon belüli úgy, hogy a (a_k, a_j) intervallumba esik és az $\overline{a}_k$ kívüli úgy, hogy $a_j < \overline{a}_k$, akkor

szükségszerű, hogy az a_k az $\overline{a}_{ij}$-hez legyen közelebb mint az $\overline{a}_k$-hoz. 7.) Ha az előbbi esetben az $\overline{a}_k$ úgy kívüli, hogy $\overline{a}_k < a_i$, akkor ahhoz, hogy az a_k-hoz a két átlag közül az $\overline{a}_k$ közelebb legyen, kell, hogy $a_i - \overline{a}_k < \overline{a}_{ij} - a_k$ legyen és ahhoz hogy a a_i-hez $\overline{a}_{i,j}$ legyen közelebb kell, hogy $\overline{a}_{ij} - a_k < a_i - \overline{a}_k$ legyen. Ez viszont ellentmond egymásnak. Tehát ebben az esetben is az a_k és az a_i közül az egyik nem a saját csoport átlagához van a legközelebb.



Jelöljön tehát a az a_i, a_j, a_k közül egy olyat amelyik nem a saját részhalmazának középpontjához van a legközelebb, és legyen ennek az elemnek a súlya w. Tartozzon az a az A részhalmazhoz, a 'másik' részhalmaz pedig legyen B, és legyen a két részhalmaz átlaga $\overline{a}$ illetve $\overline{b}$. Ekkor az így választott a elemre a feltételezés szerint teljesül, hogy nem a saját csoport átlagához van a legközelebb:

$$|a - \overline{a}| \geq |a - \overline{b}|$$

A minimalizálandó távolság az A és B halmazt felhasználva a

$$D = \sum_{i=0}^{N} w_i a_i^2 - \sum_{i=0}^{N} w_i \overline{a}_i^2 = \sum_{i=0}^{N} w_i a_i^2 - \overline{a}^2 \sum_{i \in A} w_i - \overline{b}^2 \sum_{i \in B} w_i - \sum_{i \notin A \cup B} w_i \overline{a}_i^2$$

formába írható. Legyen

$$W_A = \sum_{i \in A} w_i, \qquad W_B = \sum_{i \in B} w_i \qquad \text{és} \qquad R = \sum_{i \notin A \cup B} w_i \overline{a}_i^2$$

Ezekkel a jelölésekkel a minimalizálandó mennyiségre a

$$D = \sum_{i=0}^{N} w_i a_i^2 - \overline{a}^2 W_A - \overline{b}^2 W_B - R$$

kifejezést nyerjük.

Helyezzük át az a elemet az A részhalmazból a B részhalmazba!

Ha $\overline{a'}$ illetve $\overline{b'}$ jelöli az így megváltozott A' illetve B' részhalmazok átlagát, akkor a minimalizálandó távolság az áthelyezés után a

$$D' = \sum_{i=0}^{N} w_i a_i^2 - \overline{a'}^2 (W_A - w) - \overline{b'}^2 (W_B + w) - R$$

összegre változik. Belátjuk, hogy D' kisebb mint a D.

A két új átlagra a következő két egyenlőség érvényes:

$$\begin{aligned} \overline{a'} &= \frac{W_A \overline{a} - wa}{W_A - w} \\ \overline{b'} &= \frac{W_B \overline{b} + wa}{W_B + w} \end{aligned}$$

Ezt felhasználva a D' felírásában és egyszerűsítve:

$$\begin{aligned} D' &= \sum_{i=0}^{N} w_i a_i^2 - \left(\frac{W_A \overline{a} - wa}{W_A - w} \right)^2 (W_A - w) - \left(\frac{W_B \overline{b} + wa}{W_B + w} \right)^2 (W_B + w) - R \\ &= \sum_{i=0}^{N} w_i a_i^2 - \frac{(W_A \overline{a} - wa)^2}{W_A - w} - \frac{\left(W_B \overline{b} + wa\right)^2}{W_B + w} - R \end{aligned}$$

A két célfüggvényérték különbsége tehát:

$$D - D' = \frac{(W_A \overline{a} - wa)^2}{W_A - w} - \overline{a}^2 W_A + \frac{(W_B \overline{b} + wa)^2}{W_B + w} - \overline{b}^2 W_B$$

Végezzük el a négyzetre emelést és hozzunk közös nevezőre:

$$\frac{(W_A\overline{a})^2+(wa)^2-2\cdot(W_A\overline{a})(wa)-(W_A\overline{a})^2+wW_A\overline{a}^2}{W_A-w} \quad +$$

$$+ \quad \frac{(W_B\overline{b})^2+(wa)^2+2\cdot(W_B\overline{b})(wa)-(W_B\overline{b})^2-wW_B\overline{b}^2}{W_B+w}$$

Rendezzük és emeljünk ki w-t és a W_A-t illetve a W_B-t.

$$\frac{W_Aw}{W_A-w}\cdot\left(\frac{wa^2}{W_A}-2\overline{a}a+\overline{a}^2\right)+\frac{W_Bw}{W_B+w}\cdot\left(\frac{wa^2}{W_B}+2\overline{b}a-\overline{b}^2\right)$$

Egészítsük ki a két zárójeles tényezőt teljes négyzetté! Ehhez

az első tag második tényezőjéhez $a^2\dfrac{W_A-w}{W_A}+a^2\dfrac{w-W_A}{W_A}=0$ -t

a második tag második tényezőjéhez $-a^2\dfrac{W_B+w}{W_B}+a^2\dfrac{w+W_B}{W_B}=0$ -t

kell hozzá adni. Így a $D-D'$ különbség következő kifejezését nyerjük:

$$\frac{W_A\cdot w}{W_A-w}\cdot(\overline{a}-a)^2+a^2\cdot\frac{W_A\cdot w}{W_A-w}\cdot\frac{w-W_A}{W_A}-\frac{W_Bw}{W_B+w}\cdot(\overline{b}-a)^2+\frac{W_Bw}{W_B+w}\cdot a^2\frac{W_B+w}{W_B}$$

$$=\frac{W_A\cdot w}{W_A-w}\cdot\left(\overline{a}-a\right)^2-w\cdot a^2-\frac{W_B\cdot w}{W_B+w}(\overline{b}-a)^2+w\cdot a^2$$

Ami egyszerűsítve azt adja, hogy:

$$D-D'=w\cdot\left(\frac{W_A}{W_A-w}\cdot(\overline{a}-a)^2-\frac{W_B}{W_B+w}\cdot(\overline{b}-a)^2\right)$$

Ez a különbség viszont nyilvánvalóan nem negatív. Ugyanis mivel a súlyok nem negatívak, a képletben az első tört mint szorzó nagyobb 1-nél, míg a második kisebb 1-nél. Ráadásul a feltételezésünk szerint az a egy olyan elem, amelyre teljesül az

$(a - \overline{a})^2 > (a - \overline{b})^2$. Ezért a D' valóban kisebb mint a D. Tehát az optimális felbontás valóban nem lehet nem-konvex!

Ebből kifolyólag nem szükséges a felbontások során az összes másodfajú Stirling számosságú lehetőséget végignézni, elég csak $\binom{N-1}{K-1}$ lehetőséget kell vizsgálni. ([2]) Mivel ahhoz, hogy az N elemet K halmazba tudjuk sorolni a rendezett N elem közt $K-1$ vágás helyet kell kiválasztani, hogy K halmazt kapjunk. És ahhoz, hogy ne adódjon üres halmaz az $N-1$ lehetséges elválasztóhelyből visszatevés nélkül kell választani. Mivel az $\binom{N-1}{K-1}$ nagyságrenddel kisebb mint az $S(N,K)$, ez a konvex tulajdonság jelentősen csökkenti az optimális darabolás megtalálásához szükséges vizsgálatok számát.

1.3. A csoportmódosítás feltétele

Ebben a részben azt vizsgáljuk, hogy többdimenziós megfigyelések egy csoport felosztására hogyan változik meg a csoportátlagoktól mért össz négyzetes eltérés a csoportfelosztás megváltoztatása esetén [3].

Legyen C_i egy tetszőleges és C_j egy legalább 2 elemű halmaz. Legyen a C_a a C_j egy nem-üres valódi részhalmaza és legyen a C_b egy a C_i-től diszjunkt szintén nem-üres halmaz.

Legyen $C_d = C_j \setminus C_a$ a két halmaz különbsége és $C_s = C_j \cup C_b$ a két halmaz összege.

Legyen C_q a fenti halmazok bármelyike, azaz C_i, C_j, C_a, C_b, C_d vagy C_s.

Ekkor jelölje a C_q halmazra

m_q a halmaz elemeinek a számát

$\overline{x}_q = \frac{1}{m_q} \cdot \sum_{i \in C_q} x_i$ a halmaz középpontját

$e_q = \sum_{i \in C_q} ||x_i - \overline{x}_q||^2$ az elemek a középponttól vett négyzetes össztávolságát.

Állítás

A C_d differencia halmaz középpontja a C_j és a C_a halmaz középpontja és elemszáma alapján:

$$\overline{x}_d = \frac{m_j \cdot \overline{x}_j - m_a \cdot \overline{x}_a}{m_j - m_a}$$

és az elemeinek a középponttól való össz négyzetes eltérése:

$$e_d = e_j - e_a - \tfrac{m_j \cdot m_a}{m_j + m_a} \cdot ||\overline{x}_j - \overline{x}_a||^2.$$

A C_s összeg halmaz középpontja a C_i és a C_b halmaz középpontja és elemszáma alapján:

$$\overline{x}_s = \frac{m_i \cdot \overline{x}_i + m_b \cdot \overline{x}_b}{m_i + m_b}$$

és az elemeinek a középponttól való össz négyzetes eltérése:

$$e_s = e_i + e_b + \tfrac{m_i \cdot m_b}{m_i + m_b} \cdot ||\overline{x}_i - \overline{x}_b||^2.$$

Bizonyítás

A középpont képletei nyilvánvalóak. A négyzetes összeltérés két képlete közül a differencia képlete következő módon látható be:

$$\begin{aligned}
e_d &= \sum_{\ell \in C_d} ||x_\ell||^2 - m_d||\overline{x}_d||^2 \\
&= \Big(\sum_{\ell \in C_j} ||x_\ell||^2 - \sum_{\ell \in C_a} ||x_\ell||^2\Big) - (m_j - m_a)\Big|\Big|\frac{m_j\overline{x}_j - m_a\overline{x}_a}{m_j - m_a}\Big|\Big|^2 \\
&= (e_j + m_j||\overline{x}_j||^2) - (e_a + m_a||\overline{x}_a||^2) - \frac{1}{m_j - m_a}||m_j\overline{x}_j - m_a\overline{x}_a||^2 \\
&= e_j - e_a + \Big(m_j - \frac{m_j^2}{m_j - m_a}\Big)||\overline{x}_j||^2 - \Big(m_a - \frac{m_a^2}{m_j - m_a}\Big)||\overline{x}_a||^2 + \frac{2m_jm_a}{m_j - m_a}\overline{x}_j^T\overline{x}_a \\
&= e_j - e_a - \frac{m_jm_a}{m_j - m_a}||\overline{x}_j||^2 - \frac{m_jm_a}{m_j - m_a}||\overline{x}_a||^2 + \frac{m_jm_a}{m_j - m_a} \cdot 2\overline{x}_j^T\overline{x}_a \\
&= e_j - e_a - \frac{m_jm_a}{m_j - m_a}||\overline{x}_j - \overline{x}_a||^2
\end{aligned}$$

Az összeghalmazra vonatkozó képlet hasonló módon igazolható.

A fenti képletet tipikusan egy elemű C_a és C_b halmazokra alkalmazzuk.

Ezekre a speciális esetekre, nyilvánvalóan adódnak az alábbi képletek.

Következmény 1

Ha a C_a egy elemű, akkor speciális eseteként adódik, hogy:

$\overline{x}_d = \frac{m_j \cdot \overline{x}_j + x_a}{m_j + 1}$

$e_d = e_j - \frac{m_j}{m_j+1} \cdot ||\overline{x}_j - x_a||^2.$

Ha a C_b egy elemű, akkor

$\overline{x}_s = \frac{m_i \cdot \overline{x}_i + x_b}{m_i + 1}$

$e_s = e_i + \frac{m_i}{m_i+1} \cdot ||\overline{x}_i - x_b||^2.$

Következmény 2

Legyen C_j és C_i diszjunkt és legyen x_ℓ a C_j egy eleme.

Ekkor a két halmazban a középpontoktól vett össz négyzetes eltérés:

$e_j + e_i$

ha az x_ℓ elemet áttesszük az C_i halmazba akkor pedig:

a fenti képleteket a $C_a = x_\ell$ illetve a $C_b = x_\ell$ halmazokra alkalmazva

$e_j - \frac{m_j}{m_j-1} \cdot ||\overline{x}_j - x_\ell||^2 + e_i + \frac{m_i}{m_i+1} \cdot ||\overline{x}_i - x_\ell||^2$

Tehát ha

$e_j + e_i > e_j - \frac{m_j}{m_j-1} \cdot ||\overline{x}_j - x_\ell||^2 + e_i + \frac{m_i}{m_i+1} \cdot ||\overline{x}_i - x_\ell||^2$

azaz ha

$\frac{m_j}{m_j-1} \cdot ||\overline{x}_j - x_\ell||^2 > \frac{m_i}{m_i+1} \cdot ||\overline{x}_i - x_\ell||^2$

akkor az x_ℓ elem C_j-ből C_i-be való áthelyezése *csökkeneti* a középpontoktól vett

össznégyzetes eltérést.

A fenti számolások alapján adódik a következő szabály:

Következmény 3

Az m_j elemű $\overline{x}_j$ közepű C_j egy x elemének az m_i elemű $\overline{x}_i$ közepű C_i-be való áthelyezése pontosan akkor csökkenti az össz-négyzetes eltérést, ha

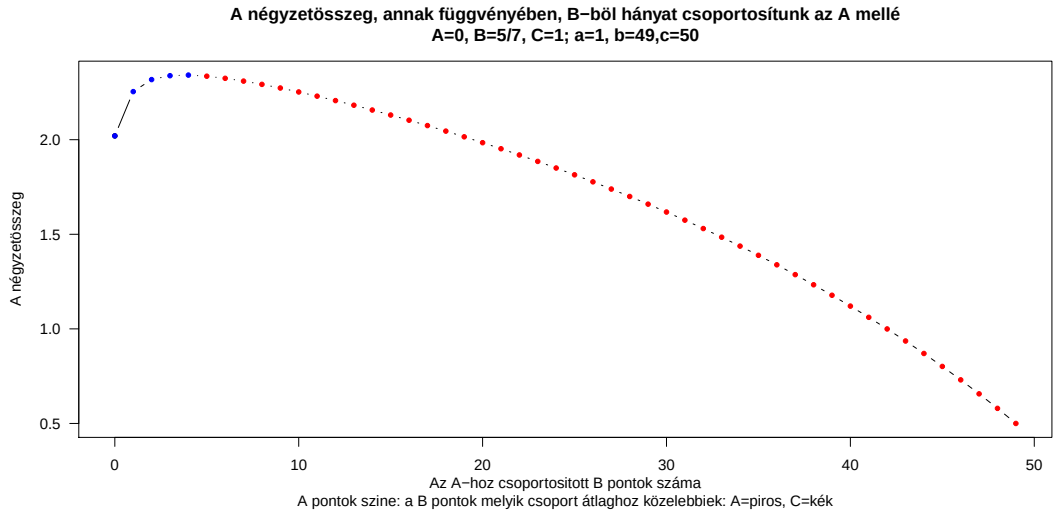
$\frac{m_j}{m_j-1} \cdot ||\overline{x}_j - x||^2 > \frac{m_i}{m_i+1} \cdot ||\overline{x}_i - x||^2$

1.3.1. Példák a csoportmódosítás feltétel alkalmazására

1. Példa

Három egydimenziós multiplicitással vett pont kétfele csoportosítását vizsgáljuk. A három pont $A = 0$, $B = 5/7$ és $C = 1$. A multiplicitásuk, rendre $a = 1$, $b = k + 49 - k$, $c = 50$. A k értéke 0-tól 49-ig fut. Azt mutatja, hogy a B pontok közül hányat csoportosítunk A-val illetve $49-k$ azt, hogy hányat C-vel. A pontok színezése azt mutatja, hogy az adott csoportosítás mellett egy B pont melyik csoporthoz van (a távolságot a klasszikus távolsággal, a különbséggel mérve) közelebb.

A 6.1 függelékbeli program eredménye az alábbi ábra.



Az ábrán látható, hogy az optimális elosztást akkor kapjuk, ha az összes B pontot a C-hez soroljuk. Ugyanakkor, ha egy olyan elosztásból indulunk ki, amelynél az A-hoz legfeljebb 3 pontot csoportosítunk, akkor a csoportosítást az egzakt feltétel szerint javítva nem az optimális csoportosításhoz jutunk.

Tanúság:

1. az optimum: mind a 49 B-t a távolabbi A-hoz kell parositani.

2. az optimum esetén érvényes, hogy mindegyik pont a hozzá legközelebbi centrumhoz tartozik.

3. ha legfeljebb 3-at csatoltunk jó helyre, akkor a "csatoljuk a pontot a legközelebbihez", szekvenciális módszer javít ugyan, de nem az optimum felé visz.

---

2. Példa

Megmutatjuk, hogy a közelebbi centrum csoportjába való áttétel javít az össznégyzetösszegen de az optimumtól mégis távolabb kerülünk.

```
# ---
```

```
# tegyuk fel, hogy az A-hoz csoportositott B-k szama 2
# a hibanegyzetösszeg:
SS(c(A,B,B))+SS(c(rep(B,47),rep(C,50))) # 2.317834

# a ket csoportkozep:
(cent.A<-mean(c(A,B,B)))               # 0.4761905
(cent.C<-mean(c(rep(B,47),rep(C,50)))) # 0.8615611
# egy B pont tavolsaga a cent.A-tol a cent.C-tol
abs(B-cent.A) # 0.2380952
abs(B-cent.C) # 0.1472754
# tehát, ha egy B pont az A-val van együtt, akkor a C-be kell tenni

# áttesszük a C-hez, ezutan az A-hoz csoportositott B-k szama 1
# a hibanegyzetösszeg:
SS(c(A,B))+SS(c(rep(B,48),rep(C,50))) # 2.254269

# rossz iranyba megy, de javult!!!
```

---

3. Példa

Ez egy olyan eset, amikor a közelebbi centrum csoportjába való áttétel javít az össznégyzetösszegen és az optimumhoz is közelebb kerülünk.

```
# ---
# tegyuk fel, hogy az A-hoz csoportositott B-k szama 4
# a hibanegyzetösszeg:
SS(c(A,rep(B,4)))+SS(c(rep(B,45),rep(C,50))) # 2.341568


# a ket csoportkozep:
(cent.A<-mean(c(A,rep(B,4))))               # 0.5714286
(cent.C<-mean(c(rep(B,45),rep(C,50)))) # 0.8646617
# egy B pont tavolsaga a cent.A-tol a cent.C-tol
abs(B-cent.A) # 0.1428571
abs(B-cent.C) # 0.1503759
# az A kozelebbi...
```

```
# tehát, ha egy B pont az C-vel van együtt, akkor a A-ba kell tenni

# áttesszük a A-hoz egyet, ezutan az A-hoz csoportositott B-k szama 5
# a hibanegyzetösszeg:
SS(c(A,B))+SS(c(rep(B,48),rep(C,50))) # 2.254269

# jo iranyba megy es javult!!!
```

---

4. Példa

Mint láttuk, egy β pontot akkor érdemes a `sajat` csoportból a `masik` csoportba áttenni, ha

$$\frac{n_{\texttt{sajat}}}{n_{\texttt{sajat}}-1}(c_{\texttt{sajat}}-\beta)^2 > \frac{n_{\texttt{masik}}}{n_{\texttt{masik}}+1}(c_{\texttt{masik}}-\beta)^2$$

ahol a c a csoport középpontját, n pedig a csoport elemeinek számát jelöli.

Megmutatjuk, hogy ennek a szabálynak az alkalmazása sem vezet mindig a globális optimális megoldás felé.

Vegyük a (1A+4B) (45B+50C) csoportosítást. Vegyük az A-hoz csoportosított 4 B pont közül az egyiket. Legyen ez a kiválasztott pont a β. A β ha azt vesszük, hogy a pontok a hozzájuk közelebbi csoportközépponthoz kell tartozniuk, akkor jó helyen van, mert a (1A+4B) csoport közepéhez közelebbi mint a (45B+50C) csoport közepéhez. Ugyanakkor, ha az egzakt feltételt vesszük, akkor a β rossz helyen van. Át kell tenni a C-hez. Viszont ha áttesszük, akkor ugyan tényleg javul az össz négyzetösszeg, ám az optimális eloszltástól még messzebb kerülünk. Ráadásul úgy, hogy onnét már az optimális lokális feltételt alkalmazva sem jutunk a globálisan optimális pontba. A szükséges számításokat a Függelék 6.2 részében közölt programmal végeztethetjük el.

5. Példa

Az alábbi példa azt mutatja, hogy a legközelebbi középponthoz tartozás nem jellemzi az optimális felbontást.

Az -1, 0, 1, 2.2 két csoportosítását veszzük. Az egyik (-1 0 1) (2.2), a másik (-1 0) (1 2.2). Az 1 mindket esetben a közelebbi centrumhoz tartozó csoportban van:

```
SS<-function(u,v) return(sum((u-mean(u))^2)+sum((v-mean(v))^2))

# a ket kozeppont
c(mean(c(-1,0,1)),mean(c(2.2))) #  0    2.2
c(mean(c(-1,0)),mean(c(1,2.2))) # -0.5  1.6

# a ket negyzetosszeg
SS(c(-1,0,1),c(2.2)) # 2
SS(c(-1,0),c(1,2.2)) # 1.22
```

Megjegyzés

Olyan példa nincs, hogy a közelebbi centrumhoz sorolom és nő, mert ha közelebbi, akkor az egzakt feltétel szerint is cserélni kell az egzakt feltétel melletti csere esetén pedig csökken az SS.

1.4. A szuboptimalizációs lemma

Jelölje $A_1 \vdots A_2$ az A optimális particiója két diszjunkt részhalmazra, és a $P(A_1)$ az A_1 egy optimális felbontása g_1 részre és $P(A_2)$ pedig A_2-nek g_2 részre, akkor a $P(A_1) \vdots P(A_2)$ az $A_1 \vdots A_2$ optimális felbontása $g_1 + g_2$ részre.

Ez többek közt azt is jelenti hogy, ha egy halmazt például négy részre akarunk osztani úgy, hogy előbb két részre majd a két részt újabb két részre bontjuk, akkor

ennek az eljárásnak megfelelő optimális négy részre bontáskor az első két részre való optimális bontás után elég csak a két rész optimális két részre bontásával foglalkozni.

1.5. A mohó bontás eltévedési valószinűsége

Ebben a pontban öt program részletet mutatunk be. Ezek eredményei abból a szempontból fontosak, hogy lássuk, mekkora az esélye annak, hogy egy szekvenciális mohó algoritmus hibás eredményt ad. Az egyes programrészletek a következők:

1. a khinégyzet tesztstatisztika és a determináns kapcsolata
2. annak bemutatása, hogy van tábla amin a mohó téved
3. az összes 200 megfigyeléses, 2×3-as kontingencia tábla amin a mohó téved
4. a mohó tévedésének a valószínűsége a megfigyelésszám függvényében
5. a mohó tévedés valószínűsége a kontingencia tábla mérete függvényében

1.5.1. A függetlenség teszt és a determináns kapcsolata

Az alábbi rövid programrészlet azt mutatja, hogy

$$\text{ha } M = \begin{pmatrix} \text{a} & \text{b} \\ \text{c} & \text{d} \end{pmatrix} \text{ akkor } \chi^2(M) = det(M)^2 \frac{n}{(a+b)(c+d)(a+c)(b+d)}$$

ahol $n = a + b + c + d$ és

$$\chi^2(M) = \Big(\frac{a - \frac{(a+b)(a+c)}{n}}{\frac{(a+b)(a+c)}{n}}\Big)^2 + \Big(\frac{b - \frac{(a+b)(b+d)}{n}}{\frac{(a+b)(b+d)}{n}}\Big)^2 + \Big(\frac{c - \frac{(a+c)(c+d)}{n}}{\frac{(a+c)(c+d)}{n}}\Big)^2 + \Big(\frac{d - \frac{(b+d)(d+c)}{n}}{\frac{(b+d)(d+c)}{n}}\Big)^2$$

az M gyakoriságtábla függetlenségének ellenőrzésére szolgáló szokványos χ^2 statisztika. Azaz egy 2×2 méretű gyakoriságtábla esetén a khinégyzet statisztika arányos a tábla determinánsával. Ez a képlet felhasználható 2×2-es gyakoriságtáblák esetén a khinégyzet statisztika könnyebb kiszámítására.

```
# kontingencia tablak
# ----
# 2x2-es matrix eseten a chi es a det viszonya
# a b
# c d
a <- 1; b <- 2; c <- 3; d <- 4;
M<-matrix(c(a,b,c,d),2,2,byrow=TRUE)
D <- det(M)^2*sum(M)/prod(c(rowSums(M),colSums(M)))
C <- sum(((M-outer(rowSums(M),colSums(M))/sum(M))/
       sqrt(outer(rowSums(M),colSums(M))/sum(M)))^2)
suppressWarnings(c(det=D,chi=C,chisq.test(M,corr=FALSE)$sta))
```

1.5.2. Van olyan 2x3-as gyakoriság tábla ami a mohót megtéveszti

A 6.3 függelék programrészlete véletlen-generátorral nyert, 2×3 méretű mátrixok, alapértelmezésben 200 hosszú sorozatából választja ki azokat, amelyek esetén a mohó amalgációs eljárás téves eredményt ad. A program segitségével olyan gyakoriság mátrixként értelmezhető mátrixokat találhatunk, amelyekre az 1. és a 2. oszlop szignifikánsan nem különbözik és ha az első két oszlop összevonása mellett nyert 2×2 méretű mátrixot vesszük, akkor abban a két oszlop szintén nem különbözik szignifikánsan. Ugyanakkor, ha azt a 2×2 méretű mátrixot vesszük, amely a 2×3 méretű mátrixból a második két oszlop összevonásával adódik, akkor ez egy olyan 2×2-es mátrix, amelynek a két oszlopa szignifikánsan különbözik. Tehát míg a talált 2×3-as mátrixok esetén a mohó amalgációs módszer alapján arra a következtetésre jutunk, hogy az oszlopokat meghatározó változó nem szignifikáns tényező, addig az valójában szignifikáns: csak ennek megmutatásához nem az első két, hanem a második két oszlopot kell összevonni.

A program az adott beállítás mellett egy 200 hosszú generátumban 4 olyan mátrixot talált, amelyik esetén a mohó módszer a leírt módon téved. Közülük a 3. a

$\begin{pmatrix} 12 & 4 & 5 \\ 7 & 7 & 11 \end{pmatrix}$ mátrix esetén külön leellenőrzi a kereső program helyességét.

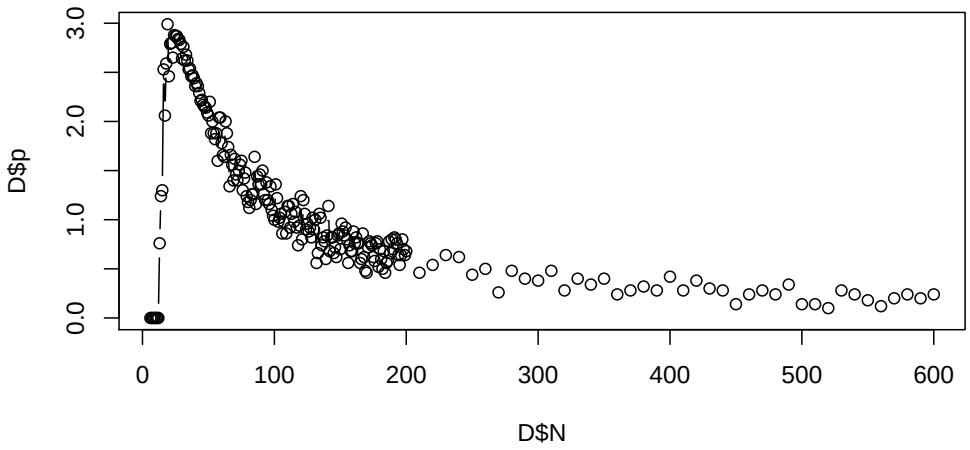
1.5.3. A mohó tévedési esélye adott megfigyelésszám esetén

Azt vizsgáljuk az összes olyan 2×3 méretű gyakoriság tábla körében, amely 200 véletlenszerűen generált megfigyelésre vonatkozik, hogy mennyi annak az esélye, hogy egy olyan táblát nyerünk, amelyre a mohó eljárás téves következtetést ad.

A 6.4 programrészlet az összes olyan 2×3 méretű gyakoriság táblát végigvizsgálva, amely 200 megfigyelés alapján készült, 6 olyat talált, amelyik esetén a mohó módszer téves következtetésre jut. Tehát nem ritka, hogy a mohó algoritmus téved.

1.5.4. A mohó tévedési esélye a megfigyelésszám függvényében

A Függelék 6.5 pontbeli programrészlete annak valószínűségét vizsgálja, hogy a mohó módszer mekkora valószínűséggel téved, a megfigyelésszám függvényében. 2×3 méretű kontingencia táblákat vizsgálunk. Azt a megfigyelésszámot, amelyből a táblázat származik 6-tól 600-ig változtatjuk. A 6 és 50 megfigyelésszám közt megszámoljuk az összes olyan mátixot amelyre a mohó módszer téves eredményt ad. Az 51 és 199 közti minden megfigyelésszámra, 200 és 600 közt a megfigyelésszámot 10-enként növelve, szimulációval adunk becslést az olyan mátrixok számára, amelyek esetén a mohó algoritmus hibás eredményt ad.
A számlálás és szimuláció 6.5 program szerinti eredményét az utolsó parancsa által generált, alább közölt ábra mutatja.



Látható, hogy az eltévedés valószínűsége a táblázat által tartalmazott megfigyelésszám függvényében erőteljesen csökken.

1.5.5. A mohó tévedés valószínűsége a táblázat mérete függvényében

Most a 6.6 programrészlet segitségével azt elemezzük, hogy hogyan függ az eltévedés gyakorisága a kontingencia tábla méretétől, feltételezve hogy összesen annyi megfigyelés van, hogy a $k \times 3$ méretű gyakoriság táblában a cellánkénti átlagos megfigyelés gyakoriság a tábla k sorszámától független állandó.

A tábla sorainak számaként a $k = 2, 3, 4, 5, 10, 12, 20$ értékeket vettük. A cellánkénti átlagos megfigyelés számot pedig 12 és 600 közt változtatunk a $K = 12, 24, 36, 48, 50, 100, 150, 200, 300, 400, 500, 600$ értékeket véve.

A mondott k és K értékekre, a $k{\cdot}3{\cdot}K$ megfigyelés alapján készült $k \times 3$ méretű gyakoriság táblák közül egyenletes eloszlással ezret-ezret vettünk. Ezeket megvizsgáltuk, hogy a mohó eljárás az adott tábla esetén eltévedne-e. A program eredményeként nyert alábbi táblázat azt mutatja, hogy az adott sorszámú és az adott átlagos cella-

gyakoriságú táblák esetén 1000-ből hány esetében tévedett volna a mohó összevonási eljárás.

```
#     | 12 24 36 48 50 100 150 200 300 400 500 600
# ------------------------------------------------
# 2   | 52 37 43 29 32  10  24  18  14   9   9   6
# 3   | 53 37 39 24 31  21  12  12  11  12   9   7
# 4   | 48 42 33 31 34  17  17  15  14   9  10   5
# 5   | 57 33 34 31 31  23  19  16  11  12  11   3
# 10  | 50 32 30 29 36  24  20  14   9  14  11  11
# 12  | 45 49 25 30 20  27   9  16  12   7  10  10
# 20  | 44 43 17 27 22  24  10  13  12  10   7   6
```

A közölt táblázaton az látható, hogy az eltévedés esélye a gyakoriság tábla méretétől kevésbé függ mint attól, hogy cellánként átlagosan hány megfigyelés esett. Ugyanakkor az is látható, hogy ez utóbbi szerint az eltévedés valószínűsége erősen csökkenő.

2. Az automatikus bontás algoritmusa

Ebben a részben azzal foglalkozom, hogyan találhatók meg egy változó lehetséges értékeinek azon csoportokra bontása, amely mellett a csoportok egymástól a lehető legjobban különböznek és az egyes csoportok viszont a lehető leghomogénebbek.

2.1. Az optimális felbontás egy algoritmusa

Az első lépésben a CHAID minden X_i magyarázó változó kategória közül az összes lehetséges módon kiválaszt kettőt. A kiválasztott magyarázó változó kategóriák és célváltozó kategóriáira kapott kontingencia táblára Pearson féle khi-négyzet teszt segítségével szignifikanciaszintet számol. Majd kiválasztásra kerül az a kontingencia-tábla mely a legmagasabb p értékkel rendelkezik - vagyis a leginkább függetlennek tekinthető - és a kiválasztott kategóriapárok összevonásra kerülnek. Ez a lépés addig folytatódik, míg van olyan p, ami magasabb, mint a megadott összevonási kritérium.

A második lépésben, mikor már nincs olyan p amire a kapott változók függetlennek tekinthetőek lennének, az algoritmus kiválasztjat a legkisebb p értéket, majd annak a kategóriái szerint felbontja az adatbázist és az algoritmus kezdődik elölről, de a már felbontott részadatbázisokra.

Az eljárás addig folytatódik, míg valamely leállási kritérium az alábbiak közül nem teljesül:

- A magyarázó változónak csak egy lehetséges kategóriája van.
- A célváltozónak csak egy lehetséges kategóriája van.
- A felosztandó részadatbázis esetszáma nem éri el az előre meghatározott minimumot.

- Az esetlegesen felosztásra kerülő új részadatbázis esetszáma nem éri el az előre meghatározott minimumot.
- A felosztások száma eléri az előre definiált maximumot
- A döntési fa mélysége elérte a maximumát.

3. A CHAID program módszere

Az automatikus kölcsönhatás-érzékelő programnak nevezett AID bevezetése az Amerikai Statisztikai Egyesület Journaljában (Morgan és Sonquist, 1963) volt. "Az automatikus interakciós detektor nevű számítógépes programot a Nemzeti Tudományos Alapítvány pénzeszközeivel fejlesztették ki, és egy monográfiában írták le (Sonquist és Morgan, 1964). AID-t később javították és leírták egy másik monográfiában (Sonquist, Baker és Morgan, 1974-ben), majd ismét javították, és átnevezték a keresést. " [1]

A CHAID az AID kiterjesztése a Chi-négyzet statisztikával (a Bonferroni korrekcióhoz a CHAID algoritmusban végzett többszörös összehasonlításokhoz). A CHAID a Ph.D. A dél-afrikai Gordon V. Kass [4] értekezésének témája 1980-ban. Kass doktori értekezlet után soha nem tapasztalták, hogy eredeti gondolkodásmódja - a népesség / adatállomány rekurzív megosztása egymást kölcsönösen kizáró és kimerítő szegmensekbe - jelentős mértékben hozzájárulna a statisztikai világra, majd a kapcsolódó kvantitatív területekre. Soha nem vizsgálta meg eredeti disszertációját, és soha nem kapott pennynyi díjat a CHAID-ről. Míg sok szakember, tudós és kockázatitőkebefektető, akik befektettek a CHAID szoftver kereskedelmi forgalomba hozatalába, nagyban részesültek pénzügyileg.

Más programok, főként Sonquist és Morgan 1973-ban, bizonyos kvantitatív célok kiigazításával jöttek létre: MNA, MCA és THAID. Az AID többváltozós változata MAID, az 1980-as évek végén alakult ki. Az érdekelt olvasók folytathatják ezeket a programokat, amelyek végül másokhoz vezetnek, mivel a listám nem teljes.

Az előző két szakaszban bemutatott módszer gyakorlati alkalmazására rendelkezésre áll az a CHAID [5] program, amelyet az R-project [6] kiegészítéseként a Müncheni Egyetemen készítettek. Ebben a részben ennek a kiegészítő csomagnak a

chaid() eljárását mutatom be.

Az eljárás neve egy rövídítés: 'CHi-squared Automatical Interaction Detection'. Hasonló módszer más statisztikai program rendszerek keretében is elérhető. Példa erre az SPSS, amely ugyanezen a néven tartalmaz egy lényegében az itt bemutatottal azonos eljárást.

Az alkalmazott program, mint magyarázóváltozókat csak diszkrét változókat fogad el. A diszkrét változók egyaránt lehetnek minősítő (nominal) és rendező (ordial) változók. Az eljárás akkor is alkalmazható ha folytonos magyarázó változók vannak. De akkor a változókat előzőleg diszkretizálni kell.

Kategóriák összevonása (Merging): A célváltozó szempontjából szignifikánsan nem különböző kategóriákat összevonjuk. A végül a csomópont adatait csoportokra bontjuk az kapott összes esetlegesen összevonással keletkezett kategóriának megfelelően.

1. Ha az összevonás után az aktuális csomopontnak csak egy kategóriája marad, akkor p értékét 1-nek vesszük.

2. Ha az összevonás után az aktuális csomopontnak két kategóriája marad, akkor a 8. ponttal folytatjuk.

3. Egyébként, meg kell keresni a megengedett kategória párjait az összevont kategóriáknak (rendezett kategóriák esetén csak szomszédos kategóriák lehetnek megengedettek.)

4. A chaid program és a partykit infrastruktura

A dolgozat fő tárgyát képező CHAID eljárás az R-project [6] keretein belül készített 'chaid' programcsomag segitségével futtatható. Ez a csomag úgy van megírva, hogy a modell reprezentációja alapvetően támaszkodik néhány a 'partykit' csomagban definiált objektumra és methodusra. Ezért előbb bemutatjuk ez utóbbi két, a témánk szempontjából fontos elemét. Majd rátérünk a 'chaid' csomag bemutatására is.

4.1. A 'constparty' és 'party' osztályú objektumok

Az R-project keretein belül T. Hothorn és A. Zeileis munkájának eredményeként rendelkezésre áll a partykit [7] objektum infrastruktura. Ez lehetővé teszi egy tetszőleges hierarchikus csoportra bontási technika mellett a csoportok hierarchiájának adminisztrálását. A csoportok tulajdonságainak a szöveges és grafikus megjelenítését. Eljárásai vannak a csoportok felbontására és összevonására és egyéb, a csoportokkal kapcsolatos műveletekre.

4.2. A 'chaid' csomag eljárásainak paraméterezése

```
chaid(formula, data, subset, weights, na.action = na.omit,
   control = chaid_control())
```

- **formula:** (vote3 .) A célváltozó és az azt magyarázó változók megadása.
- **data:** (data = USvoteS) A változókat tartalmazó adatbázis megadása.
- **subset:** egy opcionális vektor, amely az illesztési folyamatban felhasználandó megfigyelések egy részhalmazát határozza meg.

- **weights:** egy opcionális súlymérő vektor, amelyet az illesztési folyamatban kell használni. NULL vagy numerikus vektornak kell lennie.
- **na.action:** olyan funkció, amely azt jelzi, hogy mi történjen, ha az adatok NA-kat tartalmaznak. Az alapértelmezés na.omit.
- **control:** Az algoritmus hiperparaméterei a chaid_control által visszaadott paraméterek

```
chaid_control (alpha2 = .05, alpha3 = -1, alpha4 = .05, stump =FALSE,
   minsplit = 20, minbucket = 7, minprob = 0.01, maxheight = -1)
```

- **alpha2:** az összeolvasztás szignifikancia szintje (2. lépés).
- **alpha3:** az ismételt szétválasztás szignifikancia szintje (3. lépés)
- **alpha4:** a szétválasztás szignifikancia szintje (5. lépés).
- **minsplit:** minimális megfigyelésszám, amely mellett a csoport felosztható
- **minbucket:** végcsoportok minimális megfigyelésszáma
- **minprob:** a végcsoportok minimális hányada az összmegfigyelésszám szerint
- **stump:** csak egy szintű bontást végez
- **maxheight:** a fa maximális magassága

Tehát az alapértelmezés szerint a 20 megfigyelésnél kisebb csoportok nem bonthatóak (`minsplit = 20`). Minden végcsoportnak legalább 7 elemének kell lennie (`minbucket = 7`) vagy nem lehet egyik sem kisebb, mint az össz megfigyelés 1%-a (`minprob = 0.01`). A két érték közül csak az egyiknek kell teljesülnie. Megengedett

a több szintű bontás (`stump =FALSE`) és nincs megkötés a fa maximális magasságára (`maxheight = -1`). Nem megengedett az egyesített kategóriák későbbi bontása (`alpha3 = -1`). Egyaránt 5% a szignifikancia szintje annak, hogy több kategóriát összeolvasszunk (`alpha2 = .05`) és annak, hogy két katagóriát különbözőnek tekintsünk (`alpha4 = .05`).

5. Az automatikus csoportosítás egy példája

Az előzőekben ismertetett eljárásokat számos helyen alkalmazzák [8, 9, 10]. Az eljárás bemutatásához `PracTools` [11] csomag `mibrfss` adatbázisát fogjuk használni.
Az `mibrfss` adatsor egy Michigan állam beli felmérés eredményeit tartalmazza. Az adatelvétel fő célja annak a vizsgálata volt, hogy kiderítse, bizonyos körülmények az arthritis kialakulásában milyen szerepet játszanak. Az arthritis sajnálatos gyakoriságú, súlyos autoimmun betegség, amely főleg az izületeket károsítja.
A `mibrfss` adathalmazt a

```
library( PracTools)
data(mibrfss)
```

parancsok végrehajtása után, `mibrfss` néven érhetjük el. Ahogyan a

```
dim(mibrfss) # 2845   21
names(mibrfss)
```

parancsok mutatják, ez az adathalmaz 21 dimenziós és 2845 megfigyelésből áll.
A 21 változó neve és értelmezése:

- SMOKE100: 100-nál több cigit szívott el élete során (Igen, Nem)
- BMICAT3: BMI kategória(1=vékony, 2=megfelelő, 3=túlsúlyos)
- AGECAT: Korcsoport (1 = 18-24 éves; 2 = 25-34 éves; 3 = 35-44 éves; 4 = 45-54 éves; 5 = 55-64 év; 6 = 65+)
- GENHLTH: Általános egészségi állapot (1 = Kitünő; 2 = Nagyon jó; 3 = Jó; 4 = Rossz; 5 = Nagyon rossz)
- PHYSACT: Fizikai aktivitás: Az utolsó hónapban sportolt vagy végzett-e valamilyen fizikia mozgást, mint például golf, séta vagy kertészkedés? (1 = Igen; 2 = Nem)
- HIGHBP: Magasvérnyomás volt-e valaha? (1 = Igen; 2 = Nem)
- ASTHMA: Volt-e valaha asztmája? (1 = Igen; 2 = Nem)

- HISPANIC: A hispán népcsoporrthoz tartozik-e? (1 = Igen; 2 = Nem; 7 = Nincs válasz)
- GENDER: A vizsgált neme: (1 = Férfi; 2 = Nő)
- CELLPHON: Van-e WIFI telefonja? (1 = Igen; 2 = Nem)
- INETHOME: Rendelkezik-e otthoni internettel? (1 = Igen; 2 = Nem)
- WEBUSE: Milyen sűrűn használja az internetet? (1 = Naponta; 2 = Heti 5-6-szor; 3 = Heti 2-4-szor; 4 = Heti 1-szer; 5 = Nem minden héten; 6 = Rikábban mint havonta)
- RACECAT: Afro- vagy euro-amerikai-e? (1 = Euro; 2 = Afro; 3 = Más)
- EDCAT: Tanultsági szintje: (1 = Kevesebb mint érettségi; 2 = Érettségi; 3 = Szakmunkás vagy szakképesítés; 4 = Egyetem)
- INCOMC3: Jövedelem (1 = Kevesebb mint 15e dollár; 2 = 15-25e dollár közt; 3 = 25-35e dollár közt; 4 = 35-50edollár közt; 5 = 50e dollár fölött)
- DIABETE2: Volt-e valaha diabétesze? (1 = Igen ; 2 = Nem)
- CHOLCHK: Volt-e valaha koleszterin problémája? (1 = Igen; 2 = Nem)
- BMI: BMI index (folytonos)
- BINGE2: Iszik-e alkoholt? (1 = Igen; 2 = Nem)
- ARTHRIT: Volt-e valaha rheumatoid arthritis-e? (1 = Igen ; 2 = Nem ; 3 = Nem tud róla)

A következő programrészlet egy igen egyszerű modellen mutatja be a `CHAID` program működését. Az `ARTHRIT` változó eloszlását tekintjük célváltozónak. Magyarázó változóként a vizsgált nemét (`GENDER`) és a jövedelmét (`INCOMC3`) használjuk fel. A felhasználáshoz mindhárom változót `factor` osztályú változóvá kell alakítani, célszerűen úgy, hogy a lehetséges faktor értékek emlékeztessenek az adott faktor szint tényleges értelmezésére. A következőkben a `chaid` program hívásakor nem változtatunk a program paramétereinek alapértelmezés szerinti értékén. Ez azt jelentette, hogy az alkalmazott paraméterértékek $\alpha_2 = 5\%$, $\alpha_3 = -1$, $\alpha_4 = 5\%$, `minsplit = 20`, `minbucket = 7`, `minprob = 0.01`, `stump = FALSE`, `maxheight = -1` lesznek. Mint

látni fogjuk, speciálisan egyszerű modellünk esetén ezek közül a paraméterek közül érdemben csak az ??? értékek befolyásolják az optimálisnak vett modellt.

```
library(CHAID)
library( PracTools)
data(mibrfss)
D <- mibrfss # az adathalmaz munka valtozata
# a magyarazo valtozok tarolasi formaja kotelezoen 'factor'
D$GENDER <- factor(D$GENDER,labels=c('Male','Female'))
D$INCOMC3 <- factor(D$INCOMC3,labels=c('15','25','35','50','50+'))
D$ARTHRIT <- factor(D$ARTHRIT,labels=c('I', 'N', '?'))
M<-chaid(ARTHRIT~GENDER+INCOMC3,data=D)
table(D$ARTHRIT)/dim(D)[1]
print(M)
plot(M)
```

A `table` parancs azt mutatja, hogy a teljes minta szerint az arthritis előfordulásának az esélye kerekítve `37.3%`, annak az esélye, hogy nincs az illetőnek arthritise `18.2%`. Mig az össz vizsgált népesség `44.4%`-a nem tud róla, hogy van-e ilyen betegsége.
A `print(M)` a modell illesztés nyomtatott eredményét adja, a következők szerint:

```
Model formula:
ARTHRIT ~ GENDER + INCOMC3

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35
|   |   [4] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [5] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [6] INCOMC3 in 50, 50+: "?" (n = 1163, err = 49.7%)

Number of inner nodes:    2
Number of terminal nodes: 4
```

Ebből a kiírásból a következő olvasható ki:

- a rendszer az első lépésben a két változó közül a `INCOMC3` jövedelem változó

szerint látta a leghatékonyabbnak a csoportokra bontást

- az első lépésben keletkezett három csoport közül az első a [2]-vel jelölt

a leggyengébben, 25e dollár alatt keresők, a második a [3]-al jelölt, a 25-35e dollár közt keresők, a harmadik a [6]-tal jelölt, azok akik mind 35e dollár felett keresnek.

- a három csoportba rendre 487, 509+686=1195, 1163 megfigyelés esik.
- a program a második lépésben a [3]-s jelű csoportot látta felbonthatónak
- a [3]-s csoportot a vizsgált neme (`GENDER`) szerint bontotta
- két alcsoportba 509 illetve 686 megfigyelés esik.

Az eljárás 4 csoportra bontotta a megfigyeléseket. A 4 csoport a kiövetkező:

[2]-es jelű, amely 487 elemű, ebben többségében vannak

az arthritisben szenvedők, ezek aránya 100-47.8=52.2%,

jellemzésük, az hogy a jövedelmük alacsony, 25e dollár alatti.

[4]-es jelű, amely 509 elemű, ebben legtöbben (100-52.7=47.3%) azok vannak

akik nem tudnak róla, hogy van-e arthritisük, jellemzésük az, hogy

a jövedelmük közepes, 25-35e dollár közti, és hogy férfiak

[5]-ös jelű, amely 686 elemű,

ebben legtöbben (100-56.3=43.7%) azok vannak akiknek van arthritiszük,

jellemzésük az, hogy a jövedelmük közepes, 25-35e dollár közti, és hogy nők

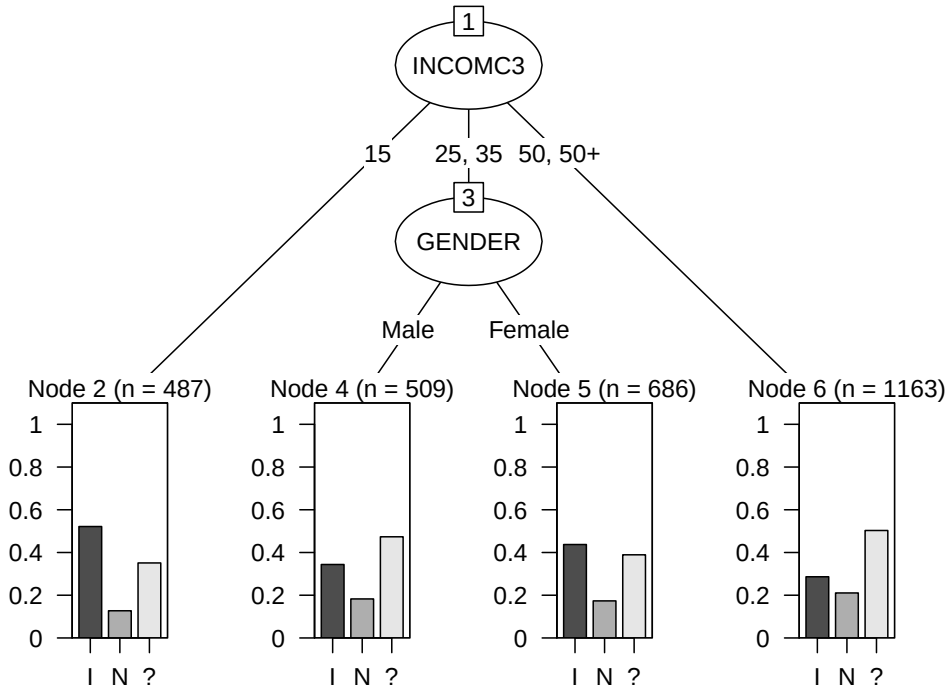
[6]-os jelű, amely 1163 elemű, ebben többségében vannak azok,

akik nem tudnak róla, hogy van-e arthritisük (100-49.7=50.3%),

jellemzésük az, hogy a jövedelmük magas, 35e dollár fölötti

Tehát az adatok és az alkalmazott modell szerint az arthritis a közepes jövedelmű nőkre és a gyenge jövedelműekre tipikus.

A futtatás eredményének grafikus képét mellékeljük



A legmagasabb oszlopok azt mutatják, hogy mi a legjellemzőbb az adott csoportra.

5.1. A paramétereinek hatása a csoportosításra

A következőkben a modell és a paraméterek változtatása mellett megmutatjuk, hogyan változhat a `CHAID` módszerrel nyert csoportosítás.

A kiindulási állapotunk az alapértelmezett `chaid_control()` függvényre a következő:

```
Model formula:
ARTHRIT ~ GENDER + INCOMC3

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35
```

```
|   |   [4] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [5] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [6] INCOMC3 in 50, 50+: "?" (n = 1163, err = 49.7%)

Number of inner nodes:    2
Number of terminal nodes: 4
```

Az előbbiekben bemutatott `chaid_control()` függvény minden egyes input argumentumára külön-külön meghívásra került egy-egy for ciklus ami az éppen aktuálisan kiválasztott argumentum összes lehetséges értékére lefuttatta a `chaid()` eljárást. A továbbiakban bemutatásra kerül, hogy a kiválasztott változók mely értékeik esetén kapunk a fenti alapértelmezetttől eltérő fát.

Az `alpha2` mint szignifikanciaszint csak 0 és 1 közötti értéket vehet fel. A 0-ra kapott döntési fa a következő:

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15, 25, 35
|   |   [3] GENDER in Male: "?" (n = 678, err = 54.7%)
|   |   [4] GENDER in Female: "I" (n = 1004, err = 52.7%)
|   [5] INCOMC3 in 50, 50+: "?" (n = 1163, err = 49.7%)

Number of inner nodes:    2
Number of terminal nodes: 3
```

Az `alpha2` $= 0.12$ szignifikanciaszintre kapott döntési fa a következő:

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
```

```
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35
|   |   [4] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [5] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [6] INCOMC3 in 50: "?" (n = 522, err = 53.1%)
|   [7] INCOMC3 in 50+: "?" (n = 641, err = 47.0%)

Number of inner nodes:    2
Number of terminal nodes: 5
```

Az elsőfajú hiba növelésével az egyenlőtől távol leszünk.

Látható, hogy az alapértelemezetthez képest a jövedelem 50 és 50+ értékei nem kerültek összevonásra, mivel nem érték el az összevonás szignifikanciaszintjét. Ha megnézzük a két értékre kapott chi-négyzet valószínűsége 0.1136 ami még a korábbi 0-11-es értékeknél még nagyobb, de a 0.12-esnél már kisebb így a két sor nem kerül összevonásra.

Természetesen hasonló a magyarázata a 0.27-re kapott döntési fának is, mivel itt a 25 és 35-ös értékek nem kerültek összevonásra, mivel azok chi-négyzet teszt alapján kapott valószínűségük 0.2604 és az így kapott döntési fa a következő:

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25
|   |   [4] GENDER in Male: "?" (n = 291, err = 51.9%)
|   |   [5] GENDER in Female: "I" (n = 404, err = 54.5%)
|   [6] INCOMC3 in 35: "?" (n = 500, err = 57.2%)
|   [7] INCOMC3 in 50: "?" (n = 522, err = 53.1%)
|   [8] INCOMC3 in 50+: "?" (n = 641, err = 47.0%)

Number of inner nodes:    2
Number of terminal nodes: 6
```

Ebből kifolyólag, ha növelem az `alpha2` értéket vagyis annak a hipotézisnek a vizsgálatához használt szignifikanciaszintet, mely szerint a H_0 hipotézis, hogy az adott magyarázóváltozó két értéke hasonló illetve az ellenhipotézis, hogy különböző, akkor nem kerülnek összevonásra a magyarázóváltozók, vagyis minél inkább tartok alpha2-vel a nullához, annál jobban tekintem azonosnak a változókat.

Elsőfajú hibát, akkor követünk el, ha elvetjük a H_0 hipotézist pedig az igaz, vagyis az igazából hasonló változókat különbözőnek tekintjük. Látható, hogy minél inkább nő a szignifikanciaszint, annál inkább csökken az elsőfajú hiba valószínűsége.

Másodfajú hibát követünk el, akkor ha elfogadjuk a H_0 hipotézist pedig az hamis. Látható, hogy itt a szignifikanciaszint csökkentésével lehet csökkenteni a másodfajú hiba valószínűségét.

A hibás döntés valószínűsége az elsőfajú és a másodfajú hiba valószínűségének összege. Ez a valószínűség csökkenthető, ha mind az elsőfajú mind a másodfajú hiba valószínűségét csökkentem, viszont mivel az elsőfajú hiba valószínűségének a szignifikanciaszint emelésével történő csökkentése magával hozza a másodfajú hiba valószínűségének növekedését illetve fordítva, így mind a két érték csökkentése csak a mintaelemszámok növelésével lehetséges.

Az `alpha3` az egyesített kategóriák újbóli szétválasztásának szignifikanciaszintje sajnos már az `alpha2`-től is függő változó, így elkerülve azt, hogy egyenlőség esetén a ciklus végtelen cíklusba menjen át. A tesztelés csak az alpha2 $>$ alpha3 esetén futott tovább, ha alpha2 $<$ alpha3 esetén hibaüzenettel leállt az eljárás, ezért az alpha2=alpha3+0.01 összefüggést választottam, viszont eltérést csak az előzőekhez hasonlókat tapasztaltam, ami az alpha2-ből adódó összefüggés volt, ezért az alpha2 értéket a 0.12-ben és később a 0.27-ben maximalizálva újból lefuttatva az alábbi lehetséges kimeneteket kaptam az alpha2=0.12 és minden 0<alpha3<alpha2 esetén:
`Model formula:`

```
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35
|   |   [4] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [5] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [6] INCOMC3 in 50: "?" (n = 522, err = 53.1%)
|   [7] INCOMC3 in 50+: "?" (n = 641, err = 47.0%)

Number of inner nodes:    2
Number of terminal nodes: 5
```

Mint látható az 50 és 50+ értékek nem kerültek összevonásra.

Az alpha2=0.27 esetén minden 0<alpha3<alpha2-re

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25
|   |   [4] GENDER in 1: "?" (n = 291, err = 51.9%)
|   |   [5] GENDER in 2: "I" (n = 404, err = 54.5%)
|   [6] INCOMC3 in 35: "?" (n = 500, err = 57.2%)
|   [7] INCOMC3 in 50: "?" (n = 522, err = 53.1%)
|   [8] INCOMC3 in 50+: "?" (n = 641, err = 47.0%)

Number of inner nodes:    2
Number of terminal nodes: 6
```

Az `alpha4` a CHAID eljárás szétválasztásának szignifikancia szintje egyenlő 0 esetén?:

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root: "?" (n = 2845, err = 55.6%)

Number of inner nodes:    0
Number of terminal nodes: 1
```

Ekkor mint látható nem történt meg az adatbázis részadatbázisokra történő felbontása, mivel minden lehetőség meghaladta a 0-nak megadott szétválasztási szignifikanciaszintet.

Az `alpha4` = 0.01

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35
|   |   [4] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [5] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [6] INCOMC3 in 50, 50+: "?" (n = 1163, err = 49.7%)

Number of inner nodes:    2
Number of terminal nodes: 4
```

Az `alpha4` = 0.12

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER
```

```
Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35
|   |   [4] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [5] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [6] INCOMC3 in 50, 50+
|   |   [7] INCOMC3 in 15, 25, 35, 50: "?" (n = 522, err = 53.1%)
|   |   [8] INCOMC3 in 50+: "?" (n = 641, err = 47.0%)

Number of inner nodes:    3
Number of terminal nodes: 5
```

Az alpha4 = 0.2

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35
|   |   [4] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [5] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [6] INCOMC3 in 50, 50+
|   |   [7] INCOMC3 in 15, 25, 35, 50: "?" (n = 522, err = 53.1%)
|   |   [8] INCOMC3 in 50+
|   |   |   [9] GENDER in Male: "?" (n = 276, err = 44.6%)
|   |   |   [10] GENDER in Female: "?" (n = 365, err = 48.8%)

Number of inner nodes:    4
Number of terminal nodes: 6
```

Az alpha4 = 0.22

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15
|   |   [3] GENDER in Male: "I" (n = 169, err = 53.3%)
|   |   [4] GENDER in Female: "I" (n = 318, err = 45.0%)
|   [5] INCOMC3 in 25, 35
|   |   [6] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [7] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [8] INCOMC3 in 50, 50+
|   |   [9] INCOMC3 in 15, 25, 35, 50: "?" (n = 522, err = 53.1%)
|   |   [10] INCOMC3 in 50+
|   |   |   [11] GENDER in Male: "?" (n = 276, err = 44.6%)
|   |   |   [12] GENDER in Female: "?" (n = 365, err = 48.8%)

Number of inner nodes:    5
Number of terminal nodes: 7
```

Az alpha4 = 0.47

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15
|   |   [3] GENDER in Male: "I" (n = 169, err = 53.3%)
|   |   [4] GENDER in Female: "I" (n = 318, err = 45.0%)
|   [5] INCOMC3 in 25, 35
|   |   [6] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [7] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [8] INCOMC3 in 50, 50+
```

```
|   |   [9] INCOMC3 in 15, 25, 35, 50
|   |   |   [10] GENDER in Male: "?" (n = 228, err = 56.1%)
|   |   |   [11] GENDER in Female: "?" (n = 294, err = 50.7%)
|   |   [12] INCOMC3 in 50+
|   |   |   [13] GENDER in Male: "?" (n = 276, err = 44.6%)
|   |   |   [14] GENDER in Female: "?" (n = 365, err = 48.8%)

Number of inner nodes:    6
Number of terminal nodes: 8
```

Az alpha4 $= 0.48$

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15
|   |   [3] GENDER in Male: "I" (n = 169, err = 53.3%)
|   |   [4] GENDER in Female: "I" (n = 318, err = 45.0%)
|   [5] INCOMC3 in 25, 35
|   |   [6] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [7] GENDER in Female
|   |   |   [8] INCOMC3 in 15,25,50,50+: "I" (n=404,err= 54.5%)
|   |   |   [9] INCOMC3 in 35: "I" (n = 282, err = 58.9%)
|   [10] INCOMC3 in 50, 50+
|   |   [11] INCOMC3 in 15, 25, 35, 50
|   |   |   [12] GENDER in Male: "?" (n = 228, err = 56.1%)
|   |   |   [13] GENDER in Female: "?" (n = 294, err = 50.7%)
|   |   [14] INCOMC3 in 50+
|   |   |   [15] GENDER in Male: "?" (n = 276, err = 44.6%)
|   |   |   [16] GENDER in Female: "?" (n = 365, err = 48.8%)

Number of inner nodes:    7
```

```
Number of terminal nodes: 9
```

Az alpha4 = 0.49

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15
|   |   [3] GENDER in Male: "I" (n = 169, err = 53.3%)
|   |   [4] GENDER in Female: "I" (n = 318, err = 45.0%)
|   [5] INCOMC3 in 25, 35
|   |   [6] GENDER in Male
|   |   |   [7] INCOMC3 in 15,25,50,50+: "?" (n = 291, err = 51.9%)
|   |   |   [8] INCOMC3 in 35: "?" (n = 218, err = 53.7%)
|   |   [9] GENDER in Female
|   |   |   [10] INCOMC3 in 15,25,50,50+: "I" (n = 404, err = 54.5%)
|   |   |   [11] INCOMC3 in 35: "I" (n = 282, err = 58.9%)
|   [12] INCOMC3 in 50, 50+
|   |   [13] INCOMC3 in 15, 25, 35, 50
|   |   |   [14] GENDER in Male: "?" (n = 228, err = 56.1%)
|   |   |   [15] GENDER in Female: "?" (n = 294, err = 50.7%)
|   |   [16] INCOMC3 in 50+
|   |   |   [17] GENDER in Male: "?" (n = 276, err = 44.6%)
|   |   |   [18] GENDER in Female: "?" (n = 365, err = 48.8%)

Number of inner nodes:     8
Number of terminal nodes: 10
```

Az alpha4 = 0.49

```
Error: node stack overflow
```

Az `minsplit` = 0 minimális megfigyelésszám, ami még felbontható. Az `minsplit` = 1196 értékre kaptuk a következő fát, ahol látható, hogy az össz esetszámunk 2845, ami meghaladja megadott 1196-os értéket, így az adatbázis felbontható, viszont továbbbontás már nem lehetséges, mivel a minimálisan megadott 1196-os értéket, egyik részaadatbázis sem éri már el.

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35: "?" (n = 1195, err = 57.5%)
|   [4] INCOMC3 in 50, 50+: "?" (n = 1163, err = 49.7%)

Number of inner nodes:    1
Number of terminal nodes: 3
```

Ha nagyobb értéket adunk meg mint az adatbázis elemeinek a száma, akkor nem történik meg egyszer sem a felbontás, mint ahogy látható az `minsplit` = 2846 értékre.

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root: "?" (n = 2845, err = 55.6%)

Number of inner nodes:    0
Number of terminal nodes: 1
```

A `minbucket` a végcsoportok minimális megfigyelésszáma. Különböző értékekre történő futtatásakor feltűnik, hogy nem kapunk az alapértelmezettől eltérő fát, még akkor sem, ha az összes esetszámnál nagyobb értéket adunk meg kritériumnak.

Ennek oka, hogy `minbucket` és a `minprob` kritériumokat egyszerre ellenőri az algoritmus, így bármelyik értékre teljesül a minimum, akkor a másik figyelmen kívül marad.

A `minprob` a végcsoportok minimális hányada az összmegfigyelés szám szerint. Ha csak az egyik kritériumot szeretnénk ha figyelembe venné az algoritmus, akkor a minprob értéket kell 1-re állítani és megadni a minbucket értéket, vagy a minbucket értéknek kell minimális 0 értéket megadni, hogy a minprob értéket vegye csak figyelembe.

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 1, 2, 3
|   |   [3] GENDER in 1: 3 (n = 678, err = 54.7%)
|   |   [4] GENDER in 2: 1 (n = 1004, err = 52.7%)
|   [5] INCOMC3 in 4, 5: 3 (n = 1163, err = 49.7%)

Number of inner nodes:    2
Number of terminal nodes: 3
```

A `stump` = FALSE esetén az algoritmus az első szétvágás után leáll.

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35: "?" (n = 1195, err = 57.5%)
|   [4] INCOMC3 in 50, 50+: "?" (n = 1163, err = 49.7%)

Number of inner nodes:    1
```

```
Number of terminal nodes: 3
```

Az `maxheight` = -1 a fa mélysége. A -1 érték nem határoz meg maximális fa mélységet, míg a pozitív számok az adatbázis részadatbázisokra történő bontásának maximális száma.

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root: "?" (n = 2845, err = 55.6%)

Number of inner nodes:    0
Number of terminal nodes: 1
```

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER

Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35: "?" (n = 1195, err = 57.5%)
|   [4] INCOMC3 in 50, 50+: "?" (n = 1163, err = 49.7%)

Number of inner nodes:    1
Number of terminal nodes: 3
```

`maxheight` = 2 Mivel a kiindulási döntési fa mélysége is 2 volt, így 2 vagy annál nagyobb érték nem befolyásolja a

```
Model formula:
ARTHRIT ~ INCOMC3 + GENDER
```

```
Fitted party:
[1] root
|   [2] INCOMC3 in 15: "I" (n = 487, err = 47.8%)
|   [3] INCOMC3 in 25, 35
|   |   [4] GENDER in Male: "?" (n = 509, err = 52.7%)
|   |   [5] GENDER in Female: "I" (n = 686, err = 56.3%)
|   [6] INCOMC3 in 50, 50+: "?" (n = 1163, err = 49.7%)

Number of inner nodes:    2
Number of terminal nodes: 4
```

6. Függelék – program részletek

Ez a függelék 6 hosszabb programrészletet tartalmaz. Az első kettő az egzakt csoportosítás feltételét elemzi, a második négy eredményei alapján a mohó algoritmus hatékonysága értékelhető.

6.1. Egzakt és minimum feltétel szerinti csoportosítás

A program eredményei alapján az egzakt és a legkisebb távolságon alapuló csoportosítás hatékonysága vizsgálható. A program a 1.3.1 pont 1. példájának eredményeit számolja ki.

```
rm(list=ls())
SS<-function(u) return(sum((u-mean(u))^2))

A<-0; B<-5/7; C<-1
a<-1; b<-49 ; c<-50

ss<-vector("numeric",b-1)
for (k in 1:(b-1))
  ss[k] <- SS(c(A,rep(B,k)))+SS(c(rep(B,b-k),c(rep(C,c))))
ss0<-SS(A)+SS(c(rep(B,b),rep(C,c)))
ss49<-SS(c(A,rep(B,b)))+SS(rep(C,c))
ss<-c(ss0,ss,ss49)

(K0<-c(A,mean(c(rep(B,b),rep(C,c))))) # mind a C-hez kozepek
(D0<-c(abs(K0[1]-B),abs(K0[2]-B))) # egy B pontra a ket tavolsag
T0<-if(D0[1]<D0[2]) 0 else 1

(K49<-c(mean(c(A,rep(B,b))),C)) # mind az A-hoz kozepek
(D49<-c(abs(K49[1]-B),abs(K49[2]-B))) # egy B pontra a ket tavolsag
T49<-if(D49[1]<D49[2]) 0 else 1

T<-vector("numeric",b-1)
Kmat<-matrix(NA,b-1,2)
```

```
Dmat<-matrix(NA,b-1,2)
for (k in 1:(b-1)) # k az A-hoz csoportositott pontok szama
  { Kmat[k,]<-c(mean(c(A,rep(B,k))),mean(c(rep(B,b-k),rep(C,c))))
    Dmat[k,]<-c(abs(Kmat[k,1]-B),abs(Kmat[k,2]-B))
    T[k]<-if(Dmat[k,1]<Dmat[k,2]) 0 else 1 }

Kmat<-rbind(K0,Kmat,K49)
Dmat<-rbind(D0,Dmat,D49)
T<-c(T0,T,T49)

cbind(0:49,ss,Kmat,Dmat,T)

plot(0:b,ss,t="b",las=1,pch=20,col=2*T+2,
    xlab="Az A-hoz csoportositott B pontok száma",
    ylab="A négyzetösszeg + szin: a B pontok melyik csoport
          átlaghoz közelebbiek: A=piros, C=kék ",
    main="A négyzetösszeg, annak függvényében, B-ből hányat
          csoportosítunk az A mellé\n
          A=0, B=5/7, C=1; a=1, b=49,c=50")
```

6.2. Az egzakt feltétel nem ad globális optimumot

Ez a programrészlet egy példát mutat arra, hogy az optimális szabály alkalmazása sem vezet mindig a globális optimum felé (lásd: 1.3.1 pontbeli 4.példa).

```
# ---
# egy pelda arra, amikor a tav szerint nem kell attenni,
# de az egzakt feltetel szerint igen,
# ugyanakkor tavolodunk az "idealis" felosztastól
# ami mint lattuk(1A+49B) (0B+50C)

A<-0; B<-5/7; C<-1
a<-1; b<-49 ; c<-50
SS<-function(u,v) return(sum((u-mean(u))^2)+sum((v-mean(v))^2))

# a vizsgalt csoportositas: (1A+4B) (45B+50C)
beta <- B # a vizsgalt pont
```

```
n_sajat <- 1+4 # az A-hoz csoportositottak ossz-szama
c_sajat <- mean(c(A,rep(B,4))) # az A csoport kozeppontja

n_masik <- 45+50 # a C-hez csoportositottak ossz-szama
c_masik <-  mean(c(rep(B,45),rep(C,50))) # a C csoport kozeppontja

# a kivalasztott B pontot ertekeljuk
c(abs(c_sajat-beta),abs(c_masik-beta)) # 0.1428571 0.1503759
# a beta a sajat csoportatlaghoz közelebb mint a masik csoportatlaghoz
# tehat a tavolsag szerint a beta-t nem erdemes attenni...

# a beta egzakt kalkulusa
dA <- n_sajat/(n_sajat-1)*(c_sajat-beta)^2 # a sajat csoportban
dC <- n_masik/(n_masik+1)*(c_masik-beta)^2 # a masik csoportban
# 0.02551020 0.02237737 tehat az egzakt szerint erdemes a C-hez tenni
c(dA,dC)
SS(c(A,rep(B,4)),c(rep(B,45),rep(C,50))) # 2.341568
SS(c(A,rep(B,3)),c(rep(B,46),rep(C,50))) # 2.338435
# tehát tenyleg javul, de a rossz iranyba megyunk ...
```

6.3. A mohó összevonás 200-ból 4-szer téved

Az 1.5.2 részben leírt program 200 véletlenszerűen választott mátrixon vizsgálja, hogy a mohó algoritmus téved-e. Az adódik, hogy 4 mátrix esetén a mohó összevonás tévesen vonja össze a mátrix oszlopiat.

```
# ----
# peldat keresunk: a szukcesszív ossszevonas teved
# egy 2x3-as tabla Poisson elemekkel
# az oszlopok A, B, C
# az [A .eq. B] es [(A+B) .eq. C] de [A .ne. (B+C)]
# a harom feltetel:
# (a) A=B
# (b) (A+B)=C
# (c) A .ne. (B+C)
```

```
kiserlet<-function(n=200)
 { sW<-function(a) suppressWarnings(a)
   eps<-.05
   k<-0
   l<-matrix(NA,0,6)
   p<-matrix(NA,0,3)
   for(i in 1:n)
     {
      M <- matrix(rpois(6,10),2,3)
      pa <- sW(chisq.test(c=FALSE,M[,1:2]                        )$p.v)
      pb <- sW(chisq.test(c=FALSE,cbind(rowSums(M[,1:2]),M[,3]))$p.v)
      pc <- sW(chisq.test(c=FALSE,cbind(M[,1],rowSums(M[,2:3])))$p.v)

   if(all(c(pa>eps,pb>eps,pc<eps)))
       {
         k <- k+1
         p <- rbind(p,c(pa,pb,pc))
         l <- rbind(l,c(M))
       }
     }
return(list(k=k,vg=p,list=l))

}

set.seed(123);kiserlet()

M <- matrix(c(12,7,4,7,5,11),2,3)
M
#   12    4    5
#    7    7   11


# az 1. es 2. oszlop fuggetlen -- tehat osszevonhato
#   12    4
#    7    7
# p-value = 15.63%
```

```
chisq.test(corr=FALSE,M[,1:2])

# az 1+2. es 3. oszlop fuggetlen -- tehat osszevonhato
#  16    5
#  14   11
# p-value = 15.21%
chisq.test(corr=FALSE,cbind(rowSums(M[,1:2]),M[,3]))

# vagyis a 3 oszlop két lepesben osszevonhato !!!

# ugyanakkor az 1. es 2+3. oszlop nem fuggetlen -- tehat szetvaghato!
# 12    9
#  7   18
# p-value = 4.56%
chisq.test(corr=FALSE,cbind(M[,1],rowSums(M[,2:3])))
```

6.4. A mohó tévedési valószínűsége adott megfigyelésszámra

Az 1.5.3 részben leírt alábbi program első része megvizsgálja, hogy a 2x3 méretű, 200 megfigyelésre vonatkozó kontingencia táblák esetén mennyi annak az esélye, hogy egy olyan táblát nyerünk, amelyre a mohó eljárás téves következtetést ad.

```
# N megfigyeles eseten
#  mennyi az eltevedes eselye,
#  a 2x3-as kontingencia tablaban

# A tenyleges vg az itt szamolt kb 3 szorosa,
#  mert azok a tablak amelyek (1,2)(1+2,3) uton hibazodnak
#  permutalva megjelennek mit (1,3)(1+3,2) es (2,3)(2+3,1) hibasak
#  de nem pontosan haromszoros, mert lehet a 3 oszlop kozt azonos...

eps<-.05

# ---
# 5 pont ami az N-t 6 poz osszeadandora bontja
bont<-function(b,N=20)
  {
```

```
    k<-6-which(c(rev(b),0)!=c((N-1):(N-5),1))[1]
    if(!k) b<-1:5 else b[k:5]<-b[k]+1:(6-k)
    return(b)
  }

# proba
N<-12
b<-(N-5):(N-1) # kezdo bonto pontok == az utolso
(b<-bont(b,N=N));(f<-diff(c(0,b,N)));sum(f)

szamol<-function(N=20)
  {
   p.val<-function(X)
     return(pchisq(det(X)^2*sum(X) /
            prod(c(rowSums(X),colSums(X))),1,,FALSE))
   b<-(N-5):(N-1)
   k<-0
   n<-choose(N-1,5)
   for(j in 1:n)
    {
      b<-bont(b,N)
      M<-matrix(diff(c(0,b,N)),2,3)
      pa <-  p.val(M[,1:2]                      )
      pb <-  p.val(cbind(rowSums(M[,1:2]),M[,3]))
      pc <-  p.val(cbind(M[,1],rowSums(M[,2:3])))
      if(all(c(pa>eps,pb>eps,pc<eps))) k <- k+1
    }
   return(list("N="=N,"freq="=c(n=n,k=k),"p="=round(k/n*100,2)))
  }

szamol() # p= 2.73

szamol(12)
# $N= 12
# $freq=  n   k 462   0
# $p= 0%

szamol(13) # p= 0
```

```
# $N= 11
# $freq=  n   k 792   6
# $p= 0.76%

# ---
# melyik az a 6?

b<-8:12

L<-matrix(NA,0,6)
B<-matrix(NA,0,5)
for(k in 1:choose(12,5))
{
  b <- bont(b,13)
  M <-matrix(diff(c(0,b,13)),2,3)
  pa <-  p.val(M[,1:2]                        )
  pb <-  p.val(cbind(rowSums(M[,1:2]),M[,3]))
  pc <-  p.val(cbind(M[,1],rowSums(M[,2:3])))
  if(all(c(pa>eps,pb>eps,pc<eps))) {B<-rbind(B,b);L<-rbind(L,c(M))}
}
B
L

# a 6 eset valojaban 3+3 mert a sorok felcserelhetoek

# 5 1 1  # 1 2 3  # 5 1 1  # 1 1 4  # 5 1 1  # 1 3 2
# 1 1 4  # 5 1 1  # 1 3 2  # 5 1 1  # 1 2 3  # 5 1 1
```

6.5. A tévedés valószínűsége a megfigyelésszám függvényében

A 1.5.4 pontban leírt alábbi programrészlet azt vizsgálja, hogy mennyi a mohó tévedési esélye a megfigyelésszám függvényében.

```
# N megfigyeles eseten
#  mennyi az eltevedes eselye,
#  a 2x3-as kontingencia tablaban
```

```
# kis N ertekekre a lehetseges esemenyek atvizsgalasaval
# nagy N ertekekre szimulalt minta alapjan becsles

# ====================================================================

setwd("E:/work/19e/=oeu/CHAID")
dir()

# ---
# bont az

szamol<-function(N=20)
  {
   bont<-function(b,N) # a kovetkezo 5 lehetseges bonto az N-1 -bol
     { k<-6-which(c(rev(b),0)!=c((N-1):(N-5),1))[1]
       if(!k) b<-1:5 else b[k:5]<-b[k]+1:(6-k)
   return(b)  }
   p.val<-function(X) # egy 2x2 ertekelese a chi p.ertekevel
     return(pchisq(det(X)^2*sum(X) /
            prod(c(rowSums(X),colSums(X))),1,,FALSE))
   eps<-.05
   # -
   b<-(N-5):(N-1) # kezdes
   k<-0 # eltevedesek szama
   n<-choose(N-1,5) # osszes esetek szama
   for(j in 1:n)
    {
      b<-bont(b,N)
      M<-matrix(diff(c(0,b,N)),2,3)
      pa <-  p.val(M[,1:2]                        )
      pb <-  p.val(cbind(rowSums(M[,1:2]),M[,3]))
      pc <-  p.val(cbind(M[,1],rowSums(M[,2:3])))
      if(all(c(pa>eps,pb>eps,pc<eps))) k <- k+1
    }
   return(list("N="=N,"freq="=c(n=n,k=k),"p="=round(k/n*100,2)))
  }

szamol()   # N=20,  k=286
```

```
szamol(19) # k=256
szamol(13) # k=6
szamol(12) # k=0

# ----
# futtatas

write(c("N","freq.n","freq.k","p"),
      "gv.dat",sep=";",ncolumns =4,append = FALSE)
for(N in 6:50)
  write(unlist(szamol(N)), "gv.dat",sep=";",append = TRUE)

# ----
# szimulacio nagy N-re

stray_prob<-function(N,r=1000)
  {
    p.val<-function(X)
      return(pchisq(det(X)^2*sum(X)/
       prod(c(rowSums(X),colSums(X))),1,,FALSE))
    eps<-.05

    k<-0
    for(ri in 1:r)
     {
      M <- matrix(diff(c(0,sort(sample(N-1,5)),N)),2,3)
      pa <-  p.val(M[,1:2]                      )
      pb <-  p.val(cbind(rowSums(M[,1:2]),M[,3]))
      pc <-  p.val(cbind(M[,1],rowSums(M[,2:3])))
  if(all(c(pa>eps,pb>eps,pc<eps))) k <- k+1
     }
return(c(N=N,r=r,k=k))
 }

stray_prob(100)

# ----
# futtatas
```

```
for(N in 51:199)
  write(c(w<-stray_prob(N,5000),round(w[3]/w[2]*100,2)),
        "gv.dat",sep=";",append = TRUE)

for(N in seq(200,600,by=10))
  write(c(w<-stray_prob(N,5000),round(w[3]/w[2]*100,2)),
        "gv.dat",sep=";",append = TRUE)

# ----
# eredmeny

setwd("E:/work/19m/=oeu/CHAID")
D<-read.table("gv.dat",header = TRUE, sep = ";")
plot(D$N,D$p,t="b")
```

6.6. A tévedés valószínűsége a táblázat mérete függvényében

Az alábbi program a mohó algoritmus tévedés valószínűségét a táblázat mérete függvényében vizsgálja. (1.5.5)

```
# N megfigyeles eseten
#  mennyi az eltevedes eselye,
#  a k x 3 -as kontingencia tablaban

# szimulacios vizsgalat N=k*3*K megfigyeles szamra
#  K= 12,24,36,48,50,100,150,200,300,400,500,600 (atlagos mfi szam)
#  k=2,3,4,5 (sorokszama)

# szimulacios kiserletek szama minden K értékre: S=1000

# ====================================================================

setwd("E:/work/19e/=oeu/CHAID")
dir()

p.val<-function(two) # egy kx2 ertekelese a chi p.ertekevel
```

```
     return(suppressWarnings(chisq.test(two,corr=FALSE)$sta))

S<-1000
K_values<-c(12, 24, 36, 48, 50, 100, 150, 200, 300, 400, 500, 600)
k_values<-c(2,3,4,5,10,12,20)
F<-matrix(NA,length(k_values),K_values)
colnames(F)<-K_values
rownames(F)<-k_values

eps<-.05
for(K in K_values)
  for(k in k_values)
   {{
   N<-k*3*K
   n<-0
   for(s in 1:S)
     {
       X<-matrix(diff(c(0,sort(sample(N-1,k*3-1)),N+1)),2,3)
       pa <-  p.val(X[,1:2]                      )
       pb <-  p.val(cbind(rowSums(X[,1:2]),X[,3]))
       pc <-  p.val(cbind(X[,1],rowSums(X[,2:3])))
       if(all(c(pa>eps,pb>eps,pc<eps))) n <- n + 1
     }
   F[which(as.numeric(rownames(F))==k),
   which(as.numeric(colnames(F))==K)]<-n
   }}

F
```

Hivatkozások

[1] W. D. Fisher. **On Grouping for Maximum Homogeneity**. *Journal of the American Statistical Association*, **53**:789–798, 1958. 5

[2] D. Aloise; A. Deshpande; P.Hansen; P.Popat. **NP-hardness of Euclidean sum-of-squares clustering**. *Machine Learning*, **Vol 75**:245–248, 2009. 12

[3] H. H. Bock. **Automatische Klassifikation**. In *Lecture Notes in Op. Res. and Math. Systems, Statistische Methoden II.*, **Vol 39**, pages 36–80. Springer, 1970. 12

[4] G. V. Kass. **An Exploratory Technique for Investigating Large Quantities of Categorical Data**. *Applied Statistics*, pages 119–127, 1980. 27

[5] The FoRt Student Project Team. *CHAID: CHi-squared Automated Interaction Detection*, 2015. R package version 0.1-2. 27

[6] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2018. 27, 29

[7] Torsten Hothorn and Achim Zeileis. **partykit: A Modular Toolkit for Recursive Partytioning in R**. *Journal of Machine Learning Research*, **16**:3905–3909, 2015. 29

[8] Y. Susanti et al. **Analysis of Chi-square Automatic Interaction Detection and Classification and Regression Tree for Classification of Corn Production**. *J. Physics Conf*, **Ser. 909**:1–8, 2017. 32

[9] Y. Wang; XSh. Ni; B. Stone. **An Automatic Interaction Detection Hybrid Model for Bankcard Response Classification**. *International Conf. on Systems and Informatics*, **ICSAI2018**:pp 1–9, 2019. 32

[10] HK. Turesson; S. Ribeiro; D. Pereira; JP. Papa; VHC. de Albuquerque. **Machine Learning Algorithms for Automatic Classification of Marmoset Vocalizations**. *PLOS*, **pone.0163041**, 2016. 32

[11] Richard Valliant, Jill A. Dever, and Frauke Kreuter. *PracTools: Tools for Designing and Weighting Survey Samples*, 2018. R package version 1.1. 32