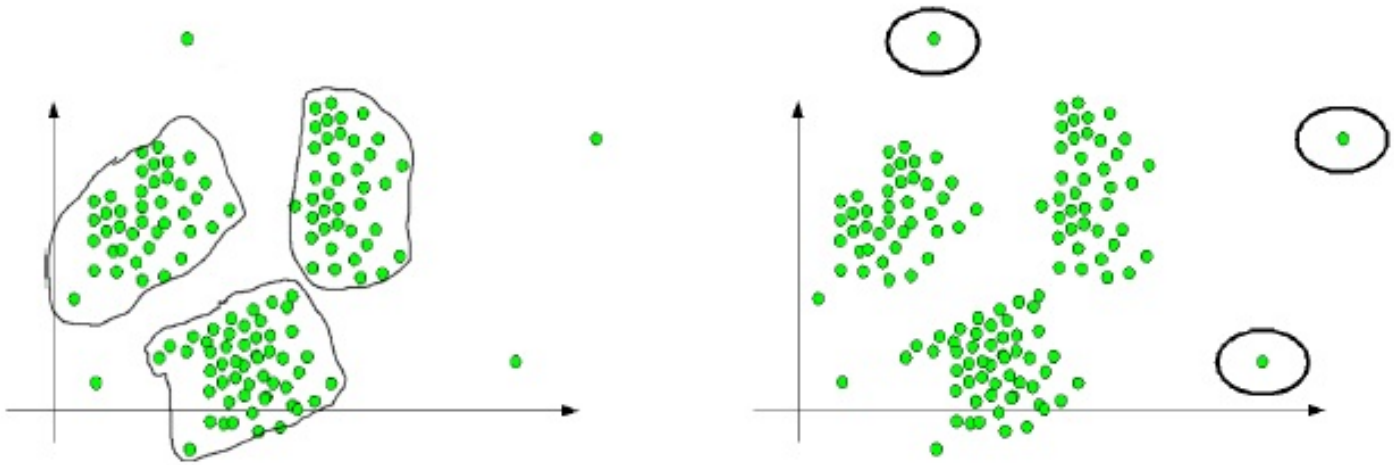
Adatbányányászati módszerek

Illyés Ágota

BME, Budapest

BME, Budapest, 2012.március 1.

Adatbányászati (data mining) algoritmusokat az adatbázisból tőrténő tudásfeltárás (knowledge discovery in databases) során alkalmaznak. A tudáskinyerés adatbázisokból egy olyan folyamat, melynek során érvényes, újszerű, lehetőleg hasznos és végső soron érthető mintákat fedezünk fel az adatokban.



Klaszterezés és különc pontok keresése

Megnevezések tisztázása

- Regresszió vagy előrejelzés (predikció)
  - a változót intervallum skálán mérjük
- Osztályozás vagy klasszifikáció (csoportba sorolás)
  - a változó diszkrét értékkészletű

Adatbányaszatban alkalmazott előrejelző és klasszifikáló módszerek

- Legközelebbi szomszéd módszerek
- Lineáris és logisztikus regresszió
- Mesterséges neurális hálózatok
- Döntési szabályok, sorozatok és fák
- Naiv Bayes klasszifikáció és Bayes hálózatok
- SVM
- Metaalgoritmusok (boosting, bagging, randomization, stb.)

Előrejelző vagy klasszifikáló módszerek tulajdonságai

- *előrejelzés teljesítménye*: milyen értékes információt ad számunkra a modell a nem megfigyelhető magyarázó változóról
- *gyorsaság*: a modell előállításának és használatának időigénye
- *robusztusság*: érzékeny-e a modell hiányzó, vagy outlier(beavatatlan) adatokra
- *skálázhatóság*: használható-e a modell nagyon nagy adathalmazokra is?
- *értelmezhetőség*: kinyerhetünk-e az emberek számára értelmezhető tudást a modell belső szerkezetéből?
- *skála-invariancia*: a klaszterezés lehetetlenség-elméletét adaptálva skála-invariánsnak hívunk egy osztályzó eljárást, ha a módszer kimenete nem változik, ha tetszőleges intervallum típusú magyarázó változó helyett annak $\alpha > 0$-szorosát vesszük

Az eljárások minimum két lépcsőben működnek:

- tanító adatbázison felépítjük a modellt
- Alkalmazzuk a modellt új adatokra, amelyen a magyarázott változó értéke nem ismert, de ismerni szeretnénk

Az osztályozás és a regresszió feladata

Az osztályozás és regresszió során n-esekkel (tuple) fogunk foglalkozni, amelyeket *objektumoknak* vagy *elemeknek* hívunk.
Adott lesz objektumok sorozata (vagy zsákja), amelyet tanító mintáknak, tanító pontoknak, tanító halmazoknak (ugyanaz az objektum többször is szerepelhet most ezekben a halmazokban) nevezünk.
A tanító pontok száma m vagy $|\tau|$ jelöljük és valójában tanításra a tanító pontok egy részét használjuk, a többi pont szerepe a tesztelés.
Az n-es j-edik elemét j-edik attribútumnak hívjuk és egy attribútumra névvel is hivatkozhatunk (pl. kor, magasság, szélesség attribútumok), nem csak sorszámmal. Minden attribútumnak saját értékkészlete van.

Az osztályozás és a regresszió feladata

Az A attribútumváltozón olyan változót értünk, amely az A értékkészletéből vehet fel értékeket.
Általános módon egy klasszifikáció vagy előrejelző módszer teljesítményét várható hasznosságával mérhetjük.

- Y-magyarázandó attribútumváltozó
- X-magyarázó attribútumváltozó(k)
- f az X értékkeszletről az Y értékkeszletre képez

Célunk a $E[U(Y, f(X))]$ *maximizálása*, ahol $U(y, \hat{y})$ jelöli az előrejelzett $\hat{y}$ *hasznosságát* vagy
$\mathbb{E}[L(Y, f(X))]$ *minimizálása*, ahol L az U inverze, egy *veszteséget* mérő függvény, ezt várható osztályozási hibának nevezik

Első definíció

- Az A attribútumhalmaz felett értelmezett döntési szabály alatt olyan $R : \phi(A) \rightarrow Y = y$ logikai implikációt értünk, amelyek feltételrészében attribútumokra vonatkozó feltételek logikai kapcsolatai állnak, a következményrészben pedig az osztályattribútumra vonatkozó ítélet.

Példa:

HŐMÉRSÉKLET = magas AND SZÉL = nincs $\rightarrow$ IDŐ JÁTÉKRA alkamas

Példa valószínűségi döntésre:

nem = férfi AND gyerek száma = 0 AND autó teljesítmény > 150LE $\rightarrow$ kockázatos = (80%,20%)

- a feltételrészben az AND, OR és negációt használjuk fel tetszőlegesen
- gyakorlatban csak olyan szabályokkal foglalkoznak, amelyben egy alapfeltétel negációja, a feltételek és kapcsolatai szerepelnek
- a szabályok feltételrészében diszjunktív normál formulák állnak, ha az azonos következményrésszel rendelkező szabályokból egy szabályt készítünk, úh. a feltételek vagy kapcsolatát képezzük
- minden formula átírható diszjunktív normál formulává a dupla negáció eliminálásával, a de Morgan és a disztributivitás szabály alkalmazásával

Második definíció

- Az $R : \phi(A) \rightarrow Y = y$ szabályra illeszkedik a t objektum, ha a feltételrész attribútumváltozóiba a t megfelelő értékeit helyettesítjük, akkor igaz értéket kapunk.

  Ha a szabály következménye is igaz, az objektumon $\Rightarrow$ a szabály fennáll vagy igaz az objektumon

Harmadik definíció

- Az $R : \phi(A) \rightarrow Y = y$ lefedi a T objektumhalmazt, ha minden objektum illeszkedik a szabályra. Adott τ tanítóhalmaz esetén az R által fedett tanítópontok halmazát $cover_\tau(R)$-rel jelöljük.
- az R szabály helyesen fedi a T halmazt, ha R fedi T-t és a halmaz összes objektuma az y osztályba tartozik
- a $cover_\tau^+(R)$ az R által helyesen fedett pontok halmaza
- a $cover_\tau^-(R)$ az R által helytelenül fedett pontok halmaza

Negyedik definíció

Az R szabály relatív fedési hibája megegyezik a rosszul osztályozott pontok számának a tanítópontokhoz vett arányával, tehát:

$$E_{r\tau}(R) = \frac{cover_{\tau}^{-}(R)}{cover_{\tau}(R)}$$

Döntési szabályok kifejezőereje

Típusai:

- Ítéletkalkulus-alapú döntési szabályok a feltételrészében predikátumok logikai kapcsolata áll (ítéletkalkulus egy formulája, amelyben nem szerepelnek a $\rightarrow$ és $\leftrightarrow$ műveleti jelek)
  -minden predikátum egy attribútumra vonatkozik
  -ha az attribútum kategória típusú $\Rightarrow A = a$ vagy $a \in \mathcal{A}$ alakú a feltétel, ahol a-konstans $\mathcal{A}$
  -$\mathcal{A}$-az A értékkészletének egy részhalmaza

Döntési szabályok kifejezőereje

-sorrend vagy intervallum típusú attribútum esetén emellett $A \leq a$ és $a' \leq A \leq a''$ szabályokat is megengedünk
-az algoritmusok többsége csak olyan egyszerű formulákat tud előállítani, amelyekben a predikátumok és kapcsolatai állnak (pl. MAGASSÁG ≤ 170 AND HAJSZÍN $=$ barna AND SZEMSZÍNE $\in$ {kék, zöld}
-a csak ítéletkalkulus alapú szabályokat tartalmazó döntési szabályokat/fákat univariate (egyváltozós) döntési szabályoknak/fáknak hívjuk.

Döntési szabályok kifejezőereje

- Reláció-alapú döntési szabályok
  -ha halmazelméleti szemmel nézzük a predikátumokat, akkor az attribútumokra vonatkozó predikátumot bináris relációnak nevezzük, amelynek egyik tagja egy változó, másik pedig egy konstans
  -a reláció alapú döntési szabályokban a második tag attribútumváltozó is lehet
  -itt pl a hajszín $=$ szemszín vagy szélesség $<$ magasság
  megengedett feltételek
  -a reláció-alapú szabályokat tartalmazó döntési szabályokat/fákat multivariate (többváltozós) döntési szabályoknak/fáknak hívjuk

Döntési szabályok kifejezőereje

- egyes esetekben a relációs szabály helyettesíthető sok egyváltozós szabálypárral
  Példa:
  hajszín = barna AND szemszín = barna, hajszín = kék AND szemszín = kék, hajszín = mályva AND szemszín = mályva

Döntési szabályok kifejezőereje

- Induktív logikai programozás
  Példa:
  építőelemek egy kupaca legyen egy torony
  -a legfelső eleme a csúcs, a maradék elemre pedig a maradék attribútummal hivatkozunk
  -ha a szélesség $<$ magasság, akkor ALAK $=$ álló $\Rightarrow$ szélesség(építőelem) $<$ magasság(építőelem) $\rightarrow$ álló(építőelem)

Döntési szabályok kifejezőereje

-sőt tovább is bonyolíthatjuk a szabályt
Példa: szélesség(torony.csúcs) $<$ magasság(torony.csúcs) AND álló(torony.maradék) $\rightarrow$ álló(torony)
-ez a rekurzív kifejezés, amely szerint egy torony akkor álló, amikor a legfelső elem magassága nagyobb mint szélessége
-a rekurziót le kell zárni: torony $=$ üres $\rightarrow$ álló(torony)
-a rekurzív szabályoknak nagyobb a kifejezőerejük, mint a reláció-alapú döntési szabályhalmazoknak
-a rekurzív szabályokat is tartalmazó szabályhalmazt logikai programnak nevezzük, ezekkel továbbiakban nem foglalkozunk.

Szabályhalmazok és szabálysorozatok

- halmazok esetén a szabályok függetlenek egymástól
- a szabályhalmaz triviális, ha tetszőleges objektum csak egy szabályra illeszkedik
- sorozat esetében egy új objektum osztályattribútumának jóslásánál egyesével sorra vesszük a szabályokat egészen addig, amíg olyat találunk, amelyre illeszkedik az objektum
- ennek a szabálynak a következményrésze adja meg az osztályattribútum értékét

- egy szabályrendszer (halmaz vagy sorozat) teljes, ha tetszőleges objektum illeszthető egy szabályra
- sorozatok esetében a teljességet általában az utolsó, ún. alapértelmezett szabály biztosítja, amely feltételrésze üres $\Rightarrow$ minden objektum illeszkedik rá
- a szabályok közötti sorrend (prioritás) biztosításával kerüljük el azt, hogy ha egy objektumra több , különböző következményrésszel rendelkező szabály illeszkedik
- a prioritás nem minden esetben kedvező!
- szabályhalmaz esetében minden szabály tudásunk egy töredékét rögzíti
- sorozatok esetén egy szabályt nem emelhetünk ki a környezetéből

Szabályhalmazok és szabálysorozatok

- a szabályok sorozata átírható szabályok halmazába úgy, hogy egyesével vesszük a szabályokat az elsőtől és a feltételrészhez hozzáfőzzük az előtte álló szabályok feltételrész negáltjainak kapcsolatát

Döntési táblázatok

- minden oszlopa egy attribútumnak felel meg, az utolsó oszlop viszont az osztályattribútumnak
- az A attribútumhoz tartozó oszlopban az A értékére vonatkozó feltétel szerepelhet, leggyakrabban A=a alakban (ítéletkalkulus-alapú döntési szabály)
- a táblázat egy sora egy döntési szabályt rögzít
- ha az attribútumok a sorban szereplő feltételeket kielégítik, akkor az osztályattribútum értéke megegyezik a sor utolsó elemének értékével

Döntési táblázat

időjárás	hőmérséklet	páratartalom	szél	játékidő
napos	meleg	magas	nincs	nem
napos	meleg	magas	van	nem
borús	meleg	magas	nincs	nem
esős	enyhe	magas	nincs	igen
esős	hideg	magas	nincs	igen
esős	hideg	magas	nincs	igen
esős	hideg	magas	nincs	igen

Döntési táblázat

- egy döntési táblázat tulajdonképpen egy speciális döntési szabályhalmaz, amelyre igaz, hogy a feltételrészben pontosan ugyanazok az attribútumok szerepelnek
- kérdések tisztázása:

1. az attribútumok melyik részhalmazát érdemes kiválasztani?
   - ideális eset, ha minden részhalmazt ki tudnánk értékelni és kiválasztani azt, amelyik a legkisebb hibát(rosszul osztályozott tanítópontok száma) adja
   - a gyakorlatban az attribútumok száma nagy, ezért az összes részhalmaz kipróbálása sok idő
2. hogyan kezeljük a folytonos attribútumkat?
   - az előző példában a hőmérsékletet diszkretizáltuk
   - ideális az lenne, ha a folytonos attribútumokat az algoritmus automatikusan tudná diszkretizálni

Az 1R algoritmus

-kiválaszt egy attribútumot és az osztályozásban kizárólag ezt használja
-annyi szabályt állít elő, ahány értéket felvesz a kiválasztott attribútum a tanítóhalmazban
-az $A{=}a \rightarrow Y{=}c$ szabály következményrésszében szereplő c osztály a legtöbbször eloforduló osztály az A attribútumában a értéket felvevő tanítóminták közül
-nyilvánvaló, hogy 1R egyértelmű szabályhalmazt állít elő

-minden attribútumértékhez meg tudjuk határozni a rosszul osztályozott tanítópontok számát
-osztályozó attribútumnak választjuk a legkevesebb rosszul osztályozott tanítópontot adó attribútumot
-hiányzó attribútumokat úgy kezelünk, mintha lenne az attribútumnak egy különleges, a többitől eltérő értéke
-sorrend és intervallum típusú attribútumnál A$\leq$ a, a'$\leq$ A $\leq$ a" és a"'$\leq$ A típusú szabályokat célszerű előállítani
-ehhez csoportosítjuk az egymást követő értékeket , úh homogén csoportok legyenek az osztályérték szempontjából (vagyis diszkretizáljuk)

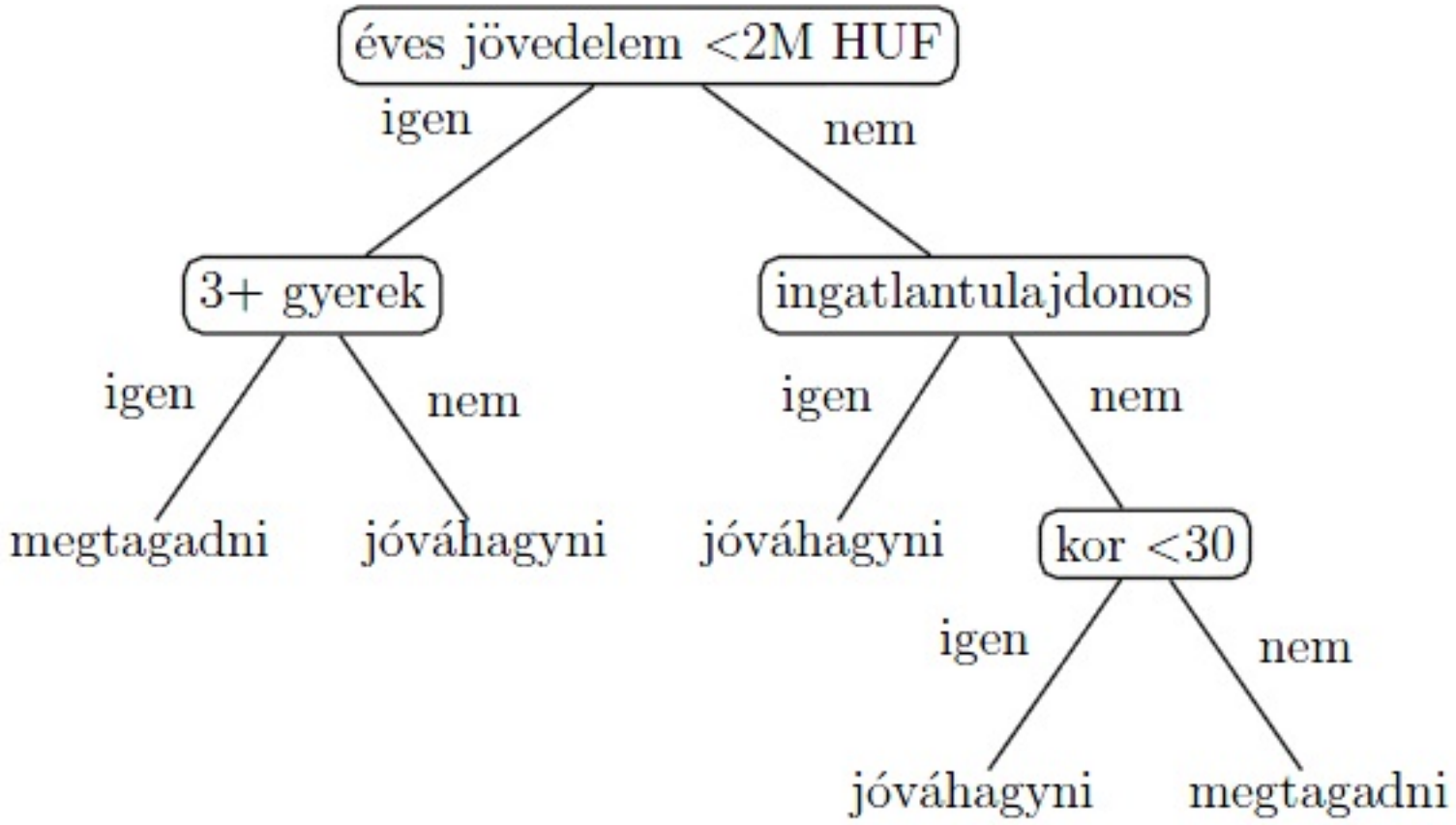
-az 1R módszer nem túl bonyolult és egyes esetekben nagyon is pontos
-van 0R osztályzó attribútum is, amely nem használ fel egyetlen attribútumot sem
-ebben az esetben az osztályozó egy feltétel nélküli szabály, amely ítéletrészében a leggyakoribb osztály áll

Döntési fák

- alapötlet: bonyolult összefüggések egyszerű döntések sorozatára vezet vissza.
- a fa gyökeréeből kiindulva haladunk lefele a csomópontokon keresztül és a csomópontokban feltett kérdésekre adott válaszoknak megfelelően addig lépkedünk, amíg egy levélbe nem érünk.
- a döntést a levél cimkéje határozza meg.
- a döntési fák nagy előnye, hogy automatikusan felismerik a lényegtelen változókat. Ha egy változóról nem nyerhető információ az adott változóról, akkor azt nem is tesztelik.
- azért előnyös ez a tulajdonság, mert így a fák teljesítménye zaj jelenlétében sem romlik, a problémamegértésünket is nagyban segíti, ha megtudjuk, hogy mely változók fontosak, és melyek nem.

- a legfontosabb változókat a fa a gyökér közelében teszteli. Másik előny, hogy a döntési fák nagyméretű adathalmazokra is hatékonyan felépíthetők.
- a döntési fák egyik fontos tulajdonsága, hogy egy csomópontnak mennyi gyereke lehet.
- egy olyan fa, amely pontjainak kettőnél több gyermeke is lehet, mindig ábrázolható bináris fával.
- a legtöbb algoritmus ezért csak bináris fát tud előállítani.

Döntési fa hitelbírálatra (Bodon Ferenc)



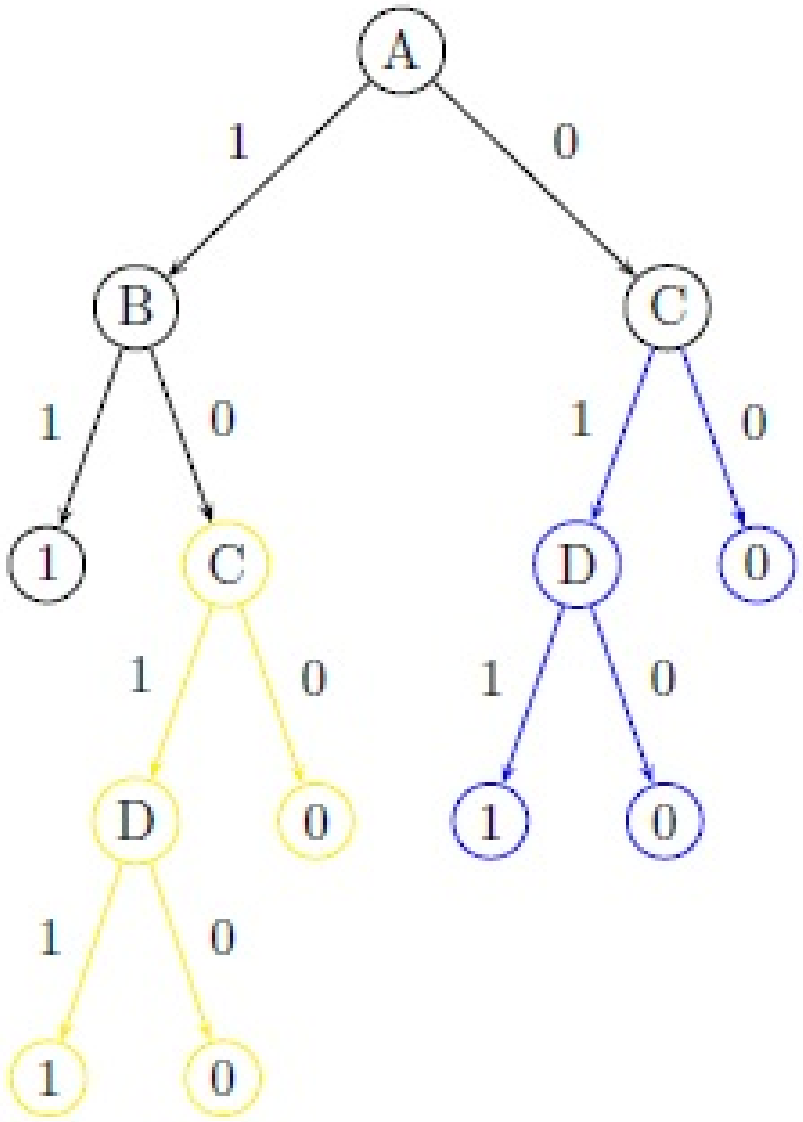
Döntési fák és döntési szabályok

- a döntési fák tulajdonsága, hogy a gyökérből egy levélbe vezető út mentén a feltételeket összeolvasva könnyen értelmezhető szabályokat kapunk a döntés meghozatalára, illetve egy laikus számára is érthető módon azt is meg tudjuk magyarázni, hogy a fa miért pont az adott döntést hozta.
- a döntési fákból nyert döntési szabályhalmazok egyértelműek. Ez triviális, hiszen tetszőleges objektumot a fa egyértelműen besorol valamelyik levélbe, a levélhez tartozó szabályra az objektum illeszkedik, a többi nem.

Vannak olyan döntési feladatok, amikor a fák túl bonyolult szabályokat állítanak elő, pl.:

- négy bináris magyarázó attribútum: A, B, C, D
- az osztályattribútum is bináris és Y-nal jelöljük
- a döntési szabálysorozat 3 szabályból áll:
  - $A = 1\ AND\ B = 1 \rightarrow Y = 1$
  - $C = 1\ AND\ D = 1 \rightarrow Y = 1$
  - $Y = 0$

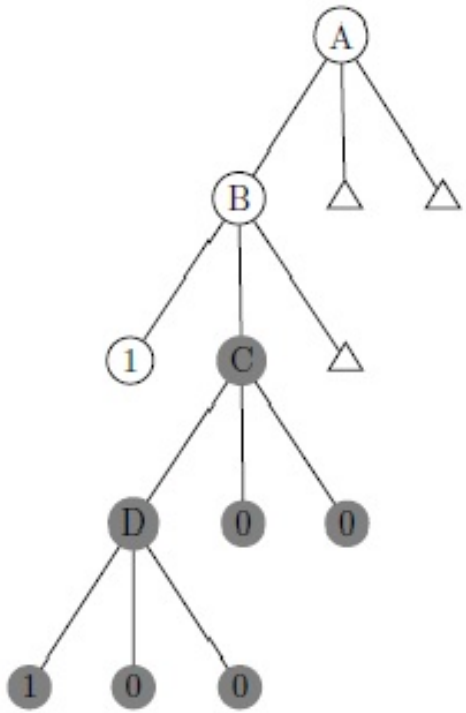
Ekkor a szabálysorozat teljes, hisz az utolsó, feltétel nélküli szabályra minden objektum illeszkedik.
A fenti példában a fa az osztályozás bonyolultabb leírását adja, mint a szabálysorozat.


A
1
0
B
C
1
0
1
0
1
C
D
0
1
0
1
0
D
0
1
0
1
0
1
0

- a sárga és kék részfák izomorfak
- a részfa által adott osztályozást egyszerűen tudjuk kezelni a döntési szabálysorozattal, de a részfák ismételt felrajzolása nem elkerülhető döntési fák esetében.
- ez egy alapprobléma, neve ismétlődő részfa probléma (replicated subtree problem)

Döntési fa előállítása

- a fát a tanító adatbázisból rekurzívan állítjuk elő
- kiindulunk a teljes adatbázisból és egy olyan kérdést keresünk, aminek segítségével a teljes tanulóhalmaz jól szétvágható
- egy szétvágás jó, ha a magyarázandó változó eloszlása a keletkezett részekben kevésbé szórt, kevésbé bizonytalan, mint a szétvágás előtt
- egyes algoritmusban a keletkező részek kb egyformák
- a részekre rekurzívan alkalmazzuk a fenti eljárást
- egy csomópont leszármazottjaiban nem vizsgáljuk többé azt az attributumot, ami alapján szétosztjuk a mintát



Ismétlődő részfaprobléma

A rekurziót megszakítjuk, ha:

- nincs több attribútum, ami alapján az elemeket továbboszthatnánk
- a csomóponthoz tartozó osztály ekkor az lesz, amelyikhez a legtöbb tanítópont tartozik
- az adott mélység elért egy megadott korlátot
- nincs olyan vágás, amely javítani tudna az aktuális osztályon

Minden levélhez hozzá kell rendelnünk a magyarázandó változó egy értékét, a döntést
Ez általában az ún. többségi szavazás elve alapján történik, az lesz a döntés, amely kategóriában a legtöbb tanító minta tartozik

Három fő algoritmust említhetünk meg a döntési fák előállítására:

- Interative Dichotomizer 3 (ID 3) család, jelenlegi változat $C5.0$"
- Classification and Regression Trees (ART^5)
- Chi-squared Automatic Interaction Detection(CHAID)

- ID3 egyik legrégibb és legismertebb algoritmus
- J. Ross Quinlan fejlesztette ki az algoritmust, ami döntési fákat hoz létre ("tanul meg") a számára megadott "tanuló" példák alapján
- ezeket a fákat a gyökértől a levelek felé haladva építi fel
- a valós életben jó néhány ilyen problémával találkozhatunk, ezek valamilyen osztályozási funkciót látnak el (pl. betegeket sorolnak kategóriákba a tüneteik alapján)
- alapötlet: kiválasztunk egy attribútumot, amelynek az értékére kíváncsiak vagyunk $\rightarrow$ ez lesz a célfüggvény
- ezek után feltesszük a következő kérdést: melyik az a további attribútum, amely a legjobban "meghatározza" a célfüggvény kimeneti értékét a példák alapján

- ez lesz a fa gyökere és ezen attribútumon lehetséges értékei lesznek az ágak
- a következő szinten ugyanez a kérdés, stb.
- a tesztattribútum kiválasztása az entrópia csökkenését alkalmazza
- ha Y egy l lehetséges értéket $p_i (i = 1, ..., l)$ valószínűséggel felvevő valószínűségi változó, akkor Y Shanner-féle entrópiáján a $H(Y) = H(p_1, \ldots, p_k) = -\sum_{j=1}^{l} p_j log_2 p_j$
- az entrópia az információ-elmélet központi fogalma

Feltételek a csomópontokban

- az ID3 algoritmus kiválasztja a minimális feltételes entrópiával rendelkező attribútumot és annyi gyerekcsomópont jön létre, amennyi értéket felvesz az attribútum
- leállási feltétel: egy ágat nem vágunk tovább, ha nincs több vizsgálható, azaz a fa maximális mélysége = az attribútumok számával
- az ID3 algoritmus nem feltétlenül bináris fát állít elő
- ha bináris fa előállítása a cél, akkor a magyarázó X attribútum típusától függően kétféle feltételt szokás létrehozni:

- intervallum típusú attribútumoknál a c két szomszédos tanítóérték átlaga
  -kategória típusú esetében $X \subseteq K$, ahol K az X értékkészletének egy részhalmaza
- az első esetben X felvett értékeivel lineáris arányos feltételes entrópiát kell számítani, a másodikban pedig a felvett értékek számával exponenciális számút (ugyanis egy n elemű halmaznak 2^n darab részhalmaza van)
- ha egy gyökérből levélig vezető úton egy attribútumot többször is vizsgálunk (különböző konstansokkal), akkor ebben az esetben kapunk jó bináris döntési fát (a fa mélysége az attribútumok számánál jóval nagyobb is lehet)

Döntési fák nyesése

-célja, hogy a felépített fá kicsit egyszerűsítsük
-feltételezzük, hogy a fa megtanult olyan esetiségeket is, amelyek csak a tanítóhalmazra jellemző
-a nyesést egy különös teszthalmazon szokás elvégezni
-előnyesés: egy intelligens STOP feltétel
-utónyesés: nagy fát növesztünk, majd elkezdjük azt zsugorítani
-a két legismertebb utónyesési eljárás:

- a részfa helyettesítés(subtree replacement): egy belső pontból induló, minden útjában levélig érő fát egyetlen levéllel helyettesítjük
- a részgráf felhúzása(subtree raising)

Döntési fák ábrázolása

-a döntési fák előállítása után két fontos kérdés szokott megfogalmazódni:

- melyek azok a szabályok, amelyek sok tanítópontra érvényesek? (mennyire jelentős az adott levél?)
- a levelek mennyire jól osztályoznak? (mennyire jó, mennyire igaz a levélhez tartozó szabály?)

-elterjedt módszer, hogy minden levelet egy körcikkely reprezentál
-a körcikkely nagysága arányos a levélhez tartozó tanítópontokkal, a színe pedig a levélhez tartozó szabály jóságát adja meg pl. minél sötétebb a szín, annál rosszabb az osztályozás aránya.
-hanyag döntési fák: amelyekben az azonos szinten elhelyezkedő pontokban ugyanazt az attribútumot vizsgáljuk

Bayesi hálózatok

Elvek, amire épülnek

- a maximum likelihood
- a Bayes-tétel

A Bayes-tétel szerint meghatározható a klasszifikációs szabály: Jelöljük Y_i-vel azt, amikor a klasszifikálandó eset az i-edik osztályba tartozik ($Y = y_i$) Az elemek megfigyelhető tulajdonságait az X vektor írja le. Az egyszerűség kedvéért a tévedés költsége legyen minden esetben azonos. Ekkor egy ismeretlen, X tulajdonságú példányt abba az osztályba (i) érdemes (optimális) sorolni, amelyikre $P(Y_i|X)$ maximális. A Bayes-szabály alapján:

$$P(Y_i|X) = \frac{P(X,Y_i)}{P(X)} = \frac{P(X|Y_i)P(Y_i)}{P(X)}$$

$$P(Y_i|X) = \frac{P(X,Y_i)}{P(X)} = \frac{P(X|Y_i)P(Y_i)}{P(X)}$$

- Y_i, amikor a klasszifikálandó eset az i-edik osztályba tartozik
- X vektor adja az elemek megfigyelhető tulajdonságait
- a tévedés költsége legyen minden esetben azonos (egyszerűség)
- egy X tulajdonságú példányt abba az osztályba érdemes (optimális) sorolni, amelyire $P(Y_i|X)$ maximális
- $P(X)$ minden i-re konstans $\rightarrow$ elegendő $P(X|Y_i)P(Y_i)$-t maximalizálni
- $P(Y_i)$-t meg tudjuk határozni
- *csak* a $P(X|Y_i)$-t kell meghatározni

Naív Bayes hálók

- a $l(2^k - 1)$ darab megbecsülendő paraméter száma $l * k$-ra csökken
- *Legyen X,Y és Z három valószínűségi változó. Az X feltételesen független Y-tól adott Z esetén, ha* $\mathbb{P}(X = x_i | Y = y_j, Z = z_k) = \mathbb{P}(X = x_i | Z = z_k)$ minden lehetséges x_i, y_j, z_k hármasra
- a naiv Bayes-hálóban egy osztályon belül az attribútumok feltételesen függetlenek egymástól
- ekkor $\mathbb{P}(X|Y)$ valószínűség kifejezhető a $\mathbb{P}(X_j|Y)$ valószínűségek szorzataként: $\mathbb{P}(X_1, X_2|Y_i) = \mathbb{P}(X_1|X_2, Y_i)\mathbb{P}(X_2|Y_i) = \mathbb{P}(X_1|Y_i)\mathbb{P}(X_2|Y_i)$
- magyarázó változó esetén: $\mathbb{P}((X_1, X_2, \ldots, X_k) = (x_1, x_2, \ldots, x_k)|Y_i) = \prod_{j=1}^{k} \mathbb{P}(X_1|Y_i)\mathbb{P}(X_2|Y_i)$

Szakirodalom

[1] Bodon Ferenc. Adatbányászati algoritmusok. *BME*, Feb. 2010

[2]http://www.cs.bme.hu/nagyadat/konyvek.html

Köszönöm a figyelmet!