

Strojno učenje

4 (I dio)

Osnovni algoritmi

Tomislav Šmuc

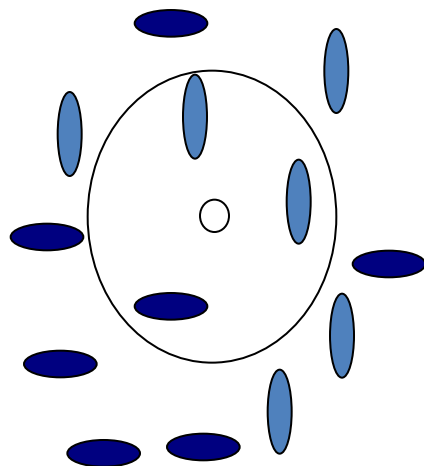
Osnovni algoritmi za učenje pod nadzorom

- Metoda najbližih susjeda – k-nn
- MAP hipoteza i princip Bayes-optimalne klasifikacija
- Naivni Bayesov klasifikator
- Stabla odlučivanja

- “Memorijski” klasifikator – (ROTE-Learner)
 - U memoriji su svi primjeri dostupni u trenutku učenja
($\langle \mathbf{x}_i, y_i \rangle \in T$)
 - Klasifikacija novih instanci – traži isti takav primjer u memoriji
($\mathbf{x}_t = \mathbf{x}_i \mid \mathbf{x}_i \in T$):
 - ukoliko ga nadje, novi primjer dobiva oznaku (y) kao njegova replika iz memorije (primjeri za učenje);
 - ukoliko ne nadje takav primjer – odustaje od klasifikacije !
 - Zapravo ne generalizira – i ne stvara nikakav model ?!

K-nn algoritam (k – najbližih susjeda)

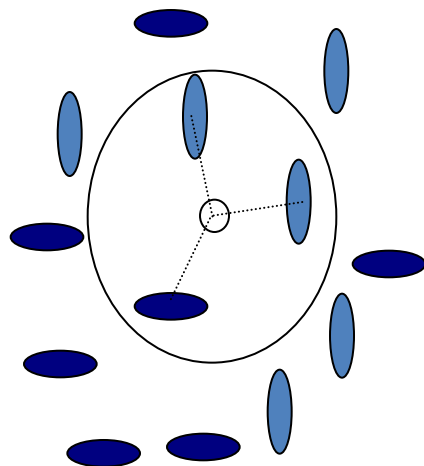
- Malo poopćenje “memorijskog” klasifikatora
- Klasificira koristeći princip analogije:
“Reci mi tko su ti prijatelji” ili “Reci mi tko su ti susjedi”
- Novi primjer dobija vrijednost ciljnog atributa koja je najčešća u njegovom susjedstvu (k-najbližih susjeda)



K-nn algoritam

- Izračuna udaljenost između novog primjera \mathbf{x}_t i svih primjera iz skupa za učenje \mathcal{T}
- Odredi k-najbližih susjeda \mathbf{x}_t iz \mathcal{T}
- Pridjeli \mathbf{x}_t klasu koja je najčešća između k-najbližih susjeda

$$3\text{-nn}(\bigcirc) = \text{two light blue vertical ellipses and one dark blue horizontal ellipse} \Rightarrow c(\bigcirc) = \text{one light blue vertical ellipse}$$



K-nn algoritam

- Udaljenost između primjera (n je broj dimenzija)

$$D(\mathbf{x}, \mathbf{x}_i) = \sqrt{\sum_{j=1}^n (x_j - x_{i,j})^2}$$

- Ako su $i=1,..k$ k -najbližih susjeda (točaka u prostoru):

$$\hat{f}(\mathbf{x}) \leftarrow \arg \max_{c \in C} \left(\sum_{i=1}^k \delta(c, f(\mathbf{x}_i)) \right)$$

gdje je

$$\delta(x, y) = 1 \text{ za } x = y$$

$$\delta(x, y) = 0 \text{ za } x \neq y$$

K-nn algoritam (k – najbližih susjeda)

$$D(\mathbf{x}, \mathbf{x}_i) = \sqrt{\sum_{j=1}^n (x_j - x_{i,j})^2}$$

X_1	X_2	X_3	X_4	X_5	X_6	X_7	y
Godine	Spol	Brak	Obrazovanje	Broj djece	Regija	Primanja (HRK/god)	Klasa G(1) / N(0)
26	m	Da	sš	1	I	88000	0
34	ž	Ne	vss	2	S	65000	1
56	ž	Da	ss	4	J	135000	0
68	m	Ne	vss	1	Z	45000	1
19	m	Ne	ss	0	C	33000	1
.....

Problem udaljenosti između primjera

X_1	X_2	X_3	X_4	X_5	X_6	X_7	y
Godine	Spol	Brak	Obrazovanje	Broj djece	Regija	Primanja (HRK/god)	Klasa G(1) / NG(0)
26	m	Da	sš	1	I	88000	0
34	ž	Ne	vss	2	S	65000	1
19	m	Ne	ss	0	C	33000	1
.....

$$D(\text{Pero}, \text{Matilda}) = \sqrt{(x_1(\text{Pero}) - x_1(\text{Matilda}))^2 + (x_2(\text{Pero}) - x_2(\text{Matilda}))^2 + \dots}$$

$$D(\text{Pero}, \text{Matilda}) = \sqrt{(45-38)^2 + (m-\text{ž})^2 + (\text{Da}-\text{Ne})^2 + (\text{SŠ}-\text{VSS})^2 + (2-1)^2 + \dots}$$

???

K-nn algoritam (k – najbližih susjeda)

K-nn – problem određivanja udaljenosti između primjera

i. Za kontinuirane varijable / atribute

$$D_2(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_{j=1}^n (x_{1,j} - x_{2,j})^2} \quad \text{Euklidska}$$

$$D_1(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^n |x_{1,j} - x_{2,j}| \quad \text{Manhattan}$$

ii. Za kategoričke varijable / atribute

$$D(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^n (1 | x_{1,j} \neq x_{2,j}) \quad \text{Hamming udaljenost (isto što i tzv. edit distance)}$$

K-nn algoritam (k – najbližih susjeda)

K-nn – udaljenost između primjera

– Atributi obično imaju vrlo različite intervale vrijednosti, stoga:

- Normalizacija numeričkih atributa na interval (0,1):

$$a_i' = \frac{a_i - \min(a_i)}{\max(a_i) - \min(a_i)}$$

- Standardizacija numeričkih atributa

$$a_i^s = \frac{a_i - \bar{a}_i}{stdev(a_i)}$$

- Udaljenosti za kategoričke attribute:

$$(x'_j = x_{i,j} \Rightarrow x'_j - x_{i,j} = 0; \quad x'_j \neq x_{i,j} \Rightarrow x'_j - x_{i,j} = 1) ; \text{ ponekad – matrice udaljenosti}$$

– K-nn algoritam možemo isto tako koristiti i za regresijske probleme !

Kako izgleda $f(\mathbf{x})$ u tom slučaju ?

K-nn - poboljšanja

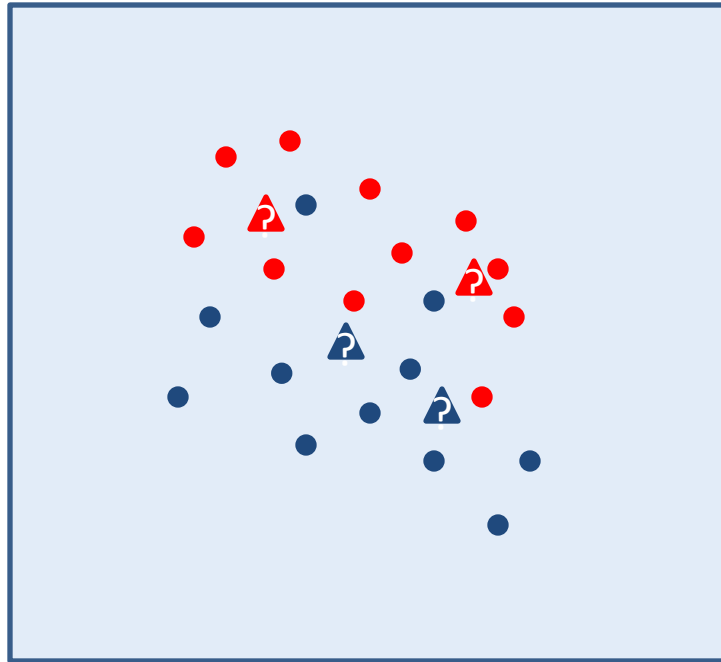
- Težinsko određivanje - uzima u obzir udaljenost k-susjeda prema novom primjeru

$$\hat{f}(\mathbf{x}) \leftarrow \arg \max_{c \in C} \left(\sum_{i=1}^k w_i \delta(c, f(\mathbf{x}_i)) \right)$$

gdje je

$$w_i \equiv \frac{1}{d(\mathbf{x}, \mathbf{x}_i)}$$

K-nn – overfitting

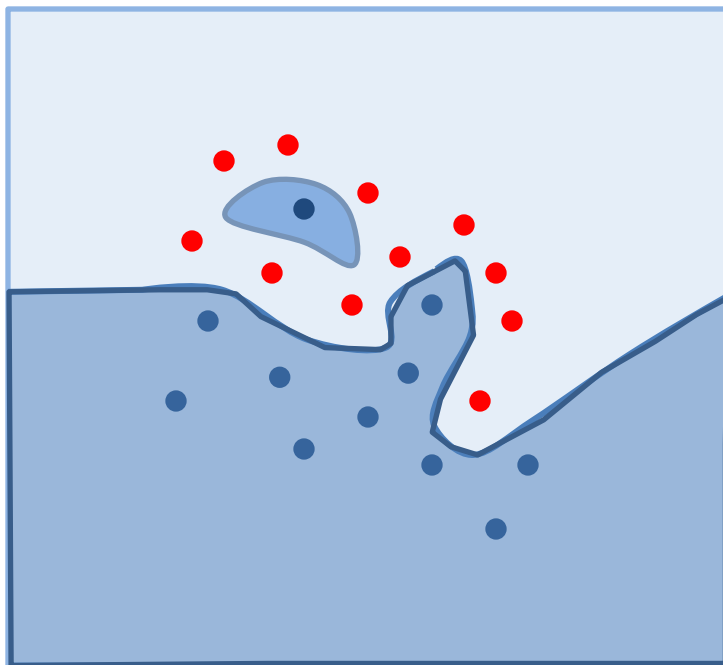


● Training set
●

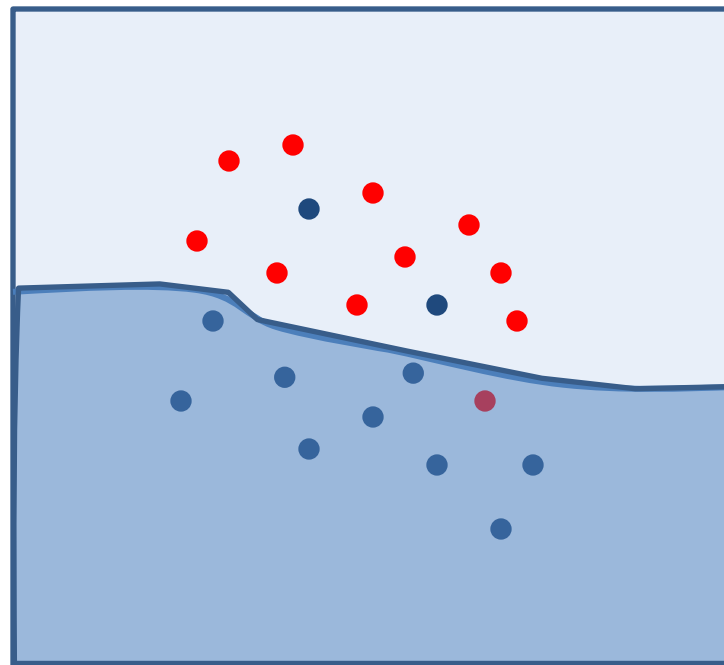
△ ? Novi primjeri
△

K-nn algoritam (k – najbližih susjeda)

1-nn (približne granične linije)

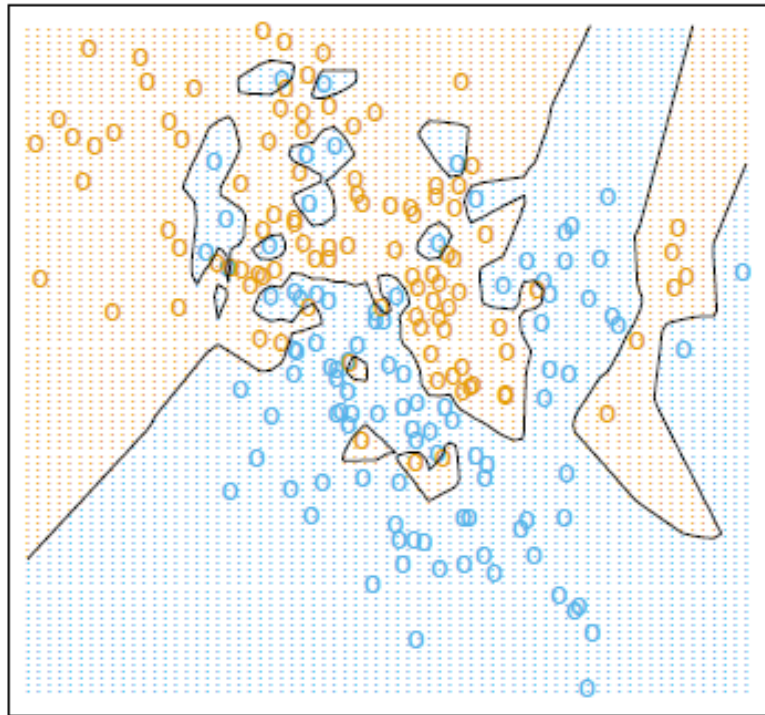


5-nn (približne granične linije)

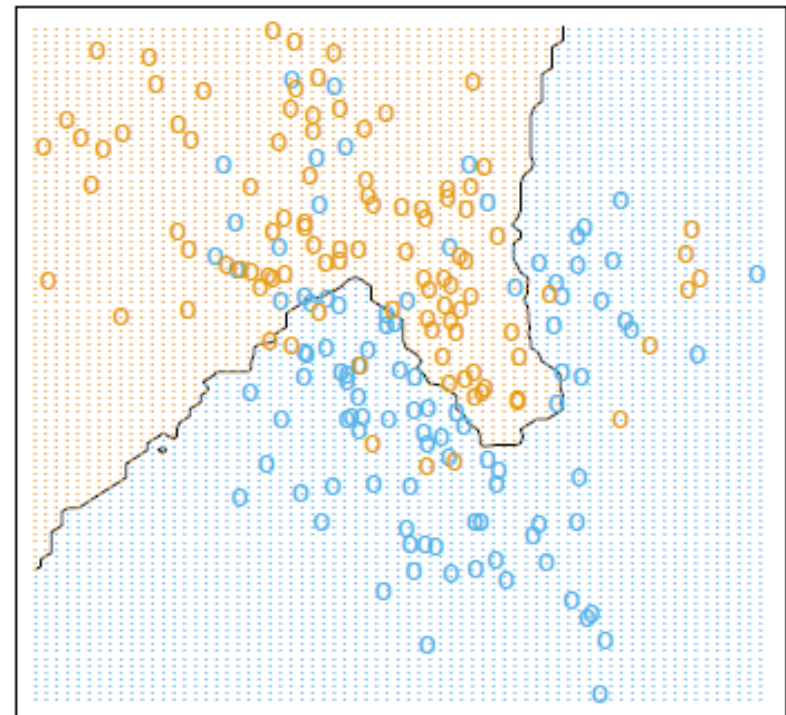


K-nn algoritam (k – najbližih susjeda)

1-Nearest Neighbor Classifier



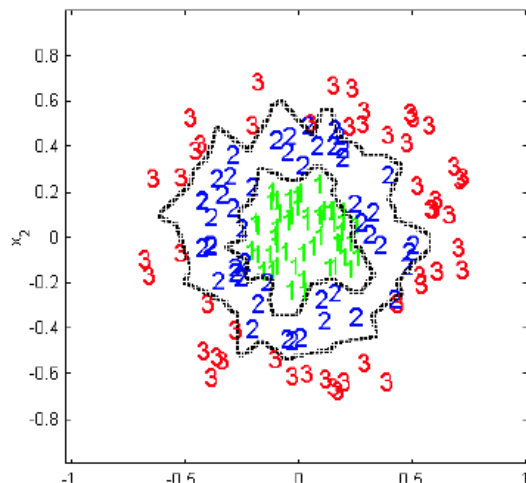
15-Nearest Neighbor Classifier



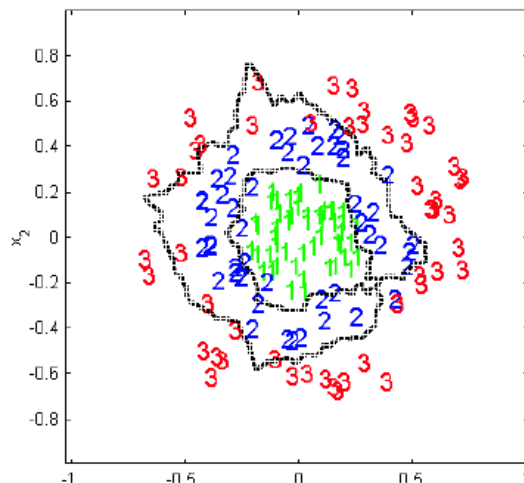
Elements of Statistical Learning (2nd Ed.) ©Hastie, Tibshirani & Friedman 2009 Chap 2

K-nn algoritam (k – najbližih susjeda)

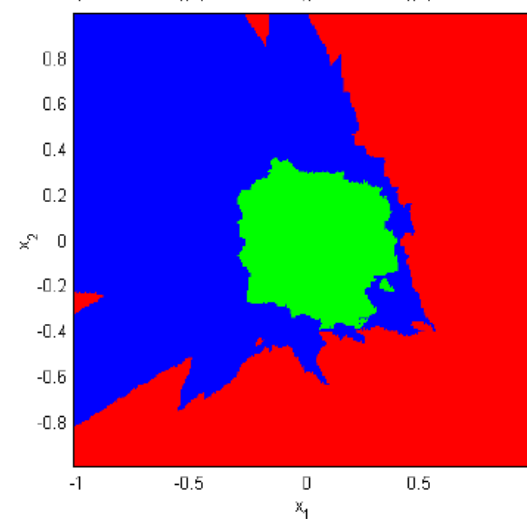
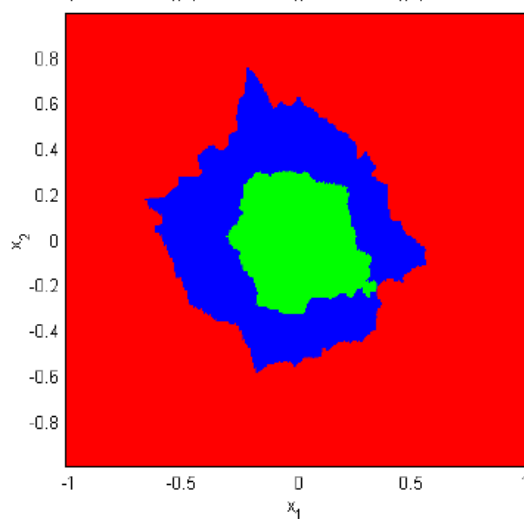
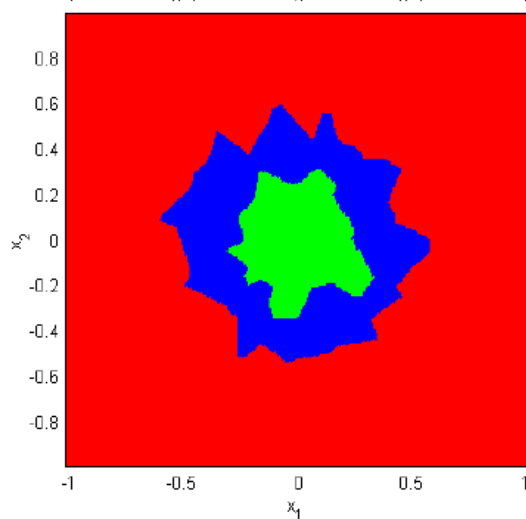
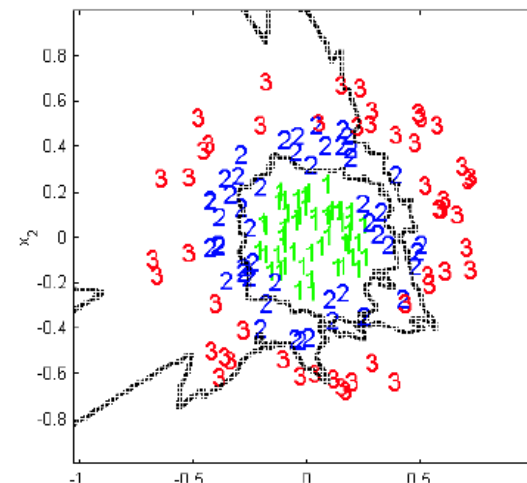
1-NN



5-NN



20-NN



Karakteristike k-nn klasifikatora

Prednosti

- Jednostavan za implementaciju
- Gotovo optimalan za ($N \rightarrow \infty$)
$$Er_{\text{O Bayes}} < Er_{1\text{-nn}} < 2 * Er_{\text{O Bayes}}$$
- Koristi lokalnu informaciju – vrlo adaptivan
- Pogodan za paralelnu implementaciju

Mane

- Memorija
- Trajanje u procesu klasifikacije
- tzv. “curse of dimensionality” problem

Karakteristike k-nn klasifikatora

1-nn vs k-nn

- korištenje većih vrijednosti k:
 - “glade” granice između klasa (manja vjerojatnost overfitting-a)
 - pruža dodatnu “probabilističku” informaciju
- opasnosti prevelikog k:
 - može potpuno poništiti prednosti lokalne estimacije (“prisiljen” koristiti sve udaljenije primjere)
 - povećava se zahtjev na računalne resurse kod klasifikacije

K-nn – problemi

1. Koji k je optimalan ? => Zahtijeva eksperimentiranje sa različitim k
2. Velik broj atributa – puno nevažnih – u određivanju udaljenosti svi imaju istu težinu !?
 - Rješenja: Težinski faktori za svaki atribut/varijablu
 - Svaki atribut dobija vlastitu težinu – kako je odrediti ?
3. Velik broj primjera u skupu za učenje
 - Za svaki novi primjer koji želimo klasificirati moramo odrediti udaljenost prema svim primjerima u memoriji i odabrati najbliže susjede !
 - Rješenja:
 - Indeksiranje primjera (**kd-trees** metoda)
 - Selektivno spremanje primjera za učenje

MAP (Maximum a posteriori) klasifikacija

Bayes-optimalna klasifikacija

(NB) - Naivni Bayes klasifikator

- U strojnom učenju u principu želimo naći najuspješniju hipotezu/model koja opisuje podatke koji su nam dostupni za učenje (T)

Probabilistički pristup

Najbolja hipoteza \approx **najvjerojatnija hipoteza**

- Mnogi algoritmi strojnog učenja baziraju se upravo na traženju najvjerojatnije hipoteze/modela (generativni algoritmi)

Vjerojatnost hipoteze ovisi od njene prethodne vjerojatnosti, vjerojatnosti dobivanja upravo onakvih podataka kakve smo skupili - uz danu hipotezu, te o samim podacima (generativni algoritmi!)

MAP hipoteza

Pojam h_{MAP} - **MAP – maximum a posteriori** hipoteze

h_{MAP} = Maksimalno vjerojatna hipoteza(e) uz dostupne podatke

$$h_{MAP} = \arg \max_{h \in H} \frac{P[T|h] \cdot P[h]}{P[T]} = P[T|h] \cdot P[h]$$

No, ustvari naš konačni cilj je dobiti najbolji rezultat ! Što znači (u slučaju klasifikacije):

- ❑ koja je **najvjerojatnija klasifikacija novog primjera** uz podatke (primjere u skupu za učenje) koje imamo ?
- ❑ Da li je odgovor uz ovakav cilj - **predikcija dobivena od h_{MAP}** ?

$$c(x_i) = \arg \max_{c_j \in C} P[c_j | h_{MAP}, x_i]$$

Iako je h_{MAP} najvjerojatnija hipoteza iz H uz dane podatke T , najvjerojatniju klasifikaciju novog primjera dobit ćemo kombiniranjem predikcija svih hipoteza iz H , i to s težinama koje odgovaraju (posteriori) vjerojatnosti hipoteza

$$c(x_i) = \arg \max_{c_j \in C} \sum_{h_k \in H} P[c_j | h_k] \cdot P[h_k | T]$$

Ovo je princip tzv. Bayesove optimalne klasifikacije

Za bilo koji sustav koji klasificira nove instance po ovom principu – možemo reći da je optimalan u Bayes-ovom smislu.

Osnovni problem s takvim klasifikatorima:

- resursi (sve hipoteze sudjeluju u klasifikaciji !)

Za potrebe klasifikacije:

recimo da nemamo hipoteze/modele – nego samo podatke (skup primjera za učenje):

$$P[c_j|x] = \frac{P[x|c_j] \cdot P[c_j]}{\sum_{k=1}^{|C|} P[x|c_k] \cdot P[c_k]} = \frac{P[x|c_j] \cdot P[c_j]}{P[x]}$$

$c_j \in C$ – klasa

x - vrijednost atributa

(pretpostavimo da se radi o kategoričkoj varijabli)

Učenje:

- Odrediti $P(\mathbf{x} | c_j)$, $P(c_j)$ iz podataka T

Klasifikacija:

- Korištenjem Bayesovog pravila odrediti:

$$c_{MAP} = \arg \max_{c_j \in C} P[\mathbf{x}^{novi} | c_j] \cdot P[c_j]$$

$c_j \in C$ – klasa

\mathbf{x} - vektor vrijednosti atributa \mathbf{x}
 $= \{x_1, x_2, \dots, x_n\}$

(pretpostavka da se radi o
kategoričkim varijablama)

$$c_{MAP} = \arg \max_{c_j \in C} P(c_j | x_1, x_2, \dots, x_n)$$

uz Bayesov teorem

$$\begin{aligned} c_{MAP} &= \arg \max_{c_j \in C} \frac{P(x_1, x_2, \dots, x_n | c_j) \cdot P(c_j)}{P(x_1, x_2, \dots, x_n)} \\ &= \arg \max_{c_j \in C} P(x_1, x_2, \dots, x_n | c_j) \cdot P(c_j) \end{aligned}$$

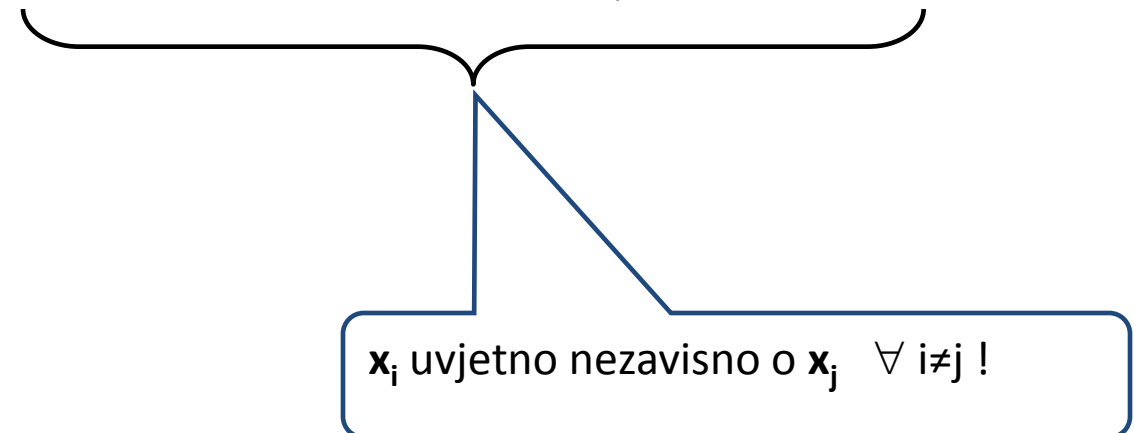
Odrediti

$P(c_j)$ - je lako

$P(x_1, x_2, \dots, x_n | c_j)$ - gotovo nemoguće !

**Potrebno je imati vrlo,
vrlo velik skup primjera !**

Osnovna pretpostavka

$$P(x_1, x_2, \dots, x_n | c_j) \approx \prod_{i=1}^n P(x_i | c_j)$$


x_i uvjetno nezavisno o $x_j \quad \forall i \neq j !$

$$\forall(i, j, k) P(c = c_j | x_1 = x_{1,j}, x_2 = x_{2,k}) = P(c = c_j | x_1 = x_{1,j})$$

c je uvjetno nezavisno od x_2 uz zadan x_1 ako je distribucija vjerojatnosti nad c neovisna o vrijednosti x_2 , uz zadane vrijednosti x_1 .

Što nam to donosi ?

Umjesto svih permutacija $P(x_1, x_2, \dots, x_n | c_j)$

$$N_{BC}(P) \approx |C| * |x_1| * |x_2| \dots * |x_n|$$

Trebaju nam samo $P(x_1 | c_j), P(x_2 | c_j), \dots, P(x_n | c_j)$

$$N_{NBC}(P) \approx |C| * [|x_1| + |x_2| + |x_n|]$$

NB – algoritam (kategoričke varijable)

Učenje:

- Za svaku vrijednost klasu c_j odredi :

$$\hat{\phi}_k = P(c = c_k) = \frac{\#T\{c = c_k\}}{|T|}$$

- Za svaku vrijednost $x_{i,j}$ za svaki atribut x_i :
odredi:

$$\hat{\kappa}_{ijk} = P(x_i = x_{i,j} | c = c_k) = \frac{\#T\{x_i = x_{i,j} \wedge c = c_k\}}{\#T\{c = c_k\}}$$

NB - algoritam

Klasifikacija:

- Za novi primjer \mathbf{x}^{novi}

$$c(\mathbf{x}^{novi}) \leftarrow \arg \max_{c_j} \hat{\phi}_j \prod \hat{\kappa}_{ijk}$$

Primjer

Spol	Brak	Klasa G(1) / N(0)
m	Da	1
m	Ne	1
ž	Da	1
ž	Da	1
ž	Ne	0
m	Ne	0
ž	Da	0

učenje

$$P(c=1)=4/7; P(c=0)=3/7 \quad \leftarrow \quad \hat{\phi}_k = P(c = c_k) = \frac{\#T\{c = c_k\}}{|T|}$$

$$\hat{K}_{ijk} = P(x_i = x_{i,j} | c = c_k) = \frac{\#T\{x_i = x_{i,j} \wedge c = c_k\}}{\#T\{c = c_k\}}$$



$$P(\text{Spol}=m \mid \text{Klasa}=1) = 1/2$$

$$P(\text{Spol}=m \mid \text{Klasa}=0) = 1/3$$

$$P(\text{Brak}=Da \mid \text{Klasa}=1) = 3/4$$

$$P(\text{Brak}=Da \mid \text{Klasa}=0) = 1/3$$

klasifikacija

Novi primjer: $\mathbf{x}^n = \{\text{Spol}=m; \text{Brak}=Da\}$, $c(\mathbf{x}^n) = ?$

$$\left. \begin{aligned} P(c(\mathbf{x}^n)=1) &= 4/7 * 1/2 * 3/4 = 12/56 = 3/14 \\ P(c(\mathbf{x}^n)=0) &= 3/7 * 1/3 * 1/3 = 3/63 = 1/21 \end{aligned} \right\} \quad c(\mathbf{x}^n) = 1$$

problemi

Kada je stvarna vjerojatnost

$$P(x_i = x_{i,j} \mid c = c_k) \ll 1$$

a imamo relativno mali broj primjera za učenje $|T|$

=> Vrlo je vjerojatno da ćemo dobiti procjenu

$$P(x_i = x_{i,j} \mid c = c_k) \approx \kappa_{ijk} = 0 \quad \Rightarrow \text{ovaj će faktor dominirati pri}$$

izračunu $c(\mathbf{x}^{novi})$

Rješenje:

m-estimate (drugi naziv - Laplacian smoothing)

$$\hat{\kappa}_{ijk} = P(x_i = x_{i,j} \mid c = c_k) = \frac{\#T\{x_i = x_{i,j} \wedge c = c_k\} + m \cdot p}{\#T\{c = c_k\} + m}$$

p - naša procjena
vjerojatnosti; (\sim
 $p=1/\text{br.kategorija varijable}$)

m – veličina “virtualnog”
uzorka
(en. equivalent sample size).

Ako su atributi/varijable kontinuirane ?

Rješenje:

- ❑ diskretizacija – više različitih pristupa
- ❑ pretpostavka da se x_j ponaša prema normalnoj distribuciji $N(\mu, \sigma)$
 - ❑ učenje: za svaku varijablu j i klasu c_k odredimo zasebno $N(\mu, \sigma)_{j,k}$
 - ❑ klasifikacija: odredimo $P(x_j^{novi} | c=c_k)$ prema:

$$P(x_j^{novi} | c = c_k) = \frac{1}{\sigma_{jk} \sqrt{2\pi}} \exp\left(\frac{-(x_j^{novi} - \mu_{jk})^2}{2\sigma_{jk}^2}\right)$$

NB – algoritam u praksi

- ❑ jednostavan: nema učenja hipoteza/modela !
- ❑ pretpostavka o uvjetnoj nezavisnosti - iako u većini situacija nije opravdana => rezultati općenito vrlo dobri !
- ❑ robustan na šum u podacima

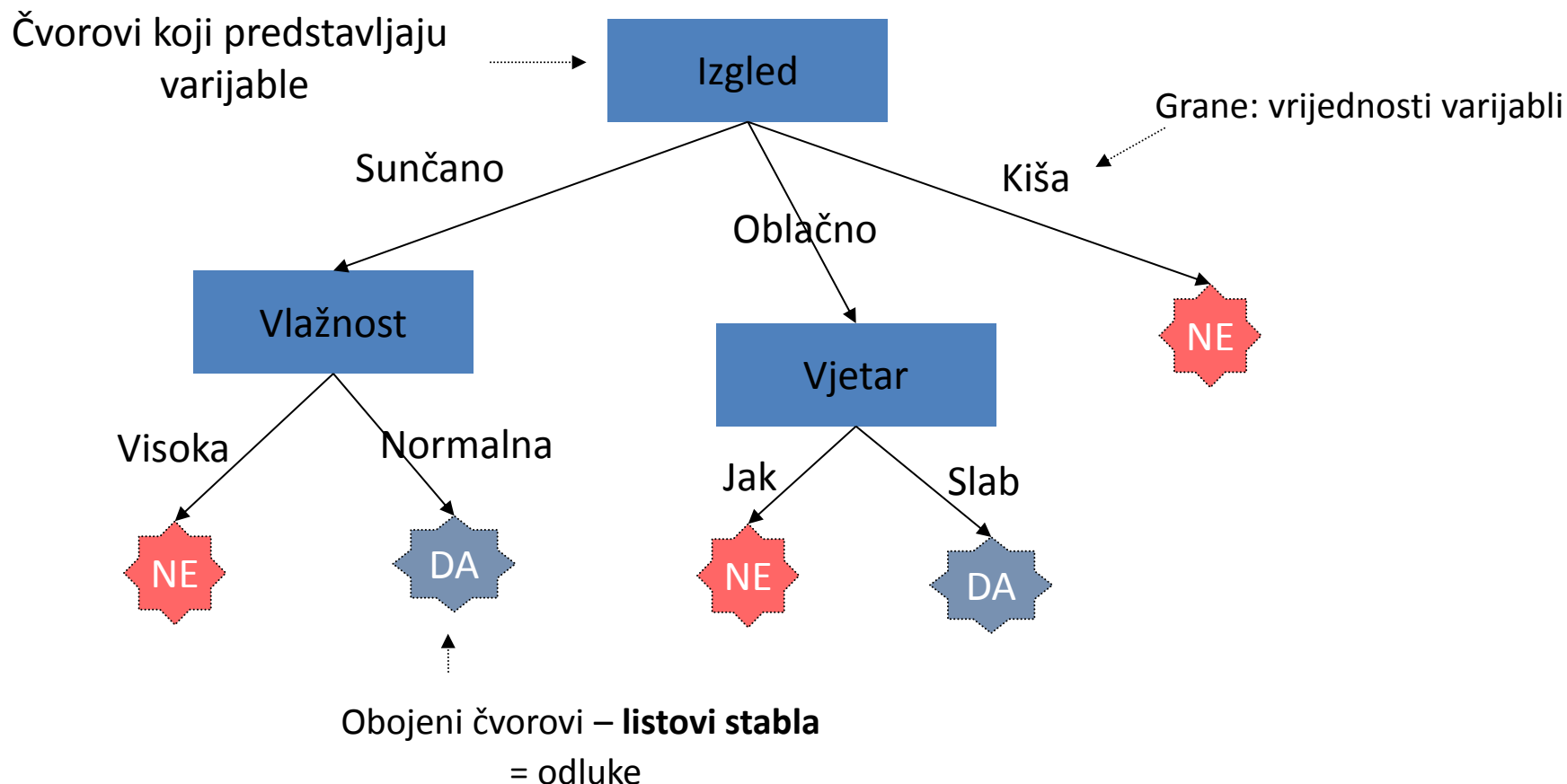
Stabla odlučivanja

Primjer: Igrati ili ne igrati !

- Želimo naučiti prepoznavati:
“kada je dobar dan za igranje tenisa” – igrati ili ne ?
- Mjerenja:
 - Varijable i njihove vrijednosti + oznake

Izgled	Temperatura	Vlažnost	Vjetar	Igrati DA/NE
<i>Sunčano</i>	<i>Hladno</i>	<i>Visoka</i>	<i>Slab</i>	<i>Da</i>
<i>Kišno</i>	<i>Vruće</i>	<i>Srednja</i>	<i>Osrednji</i>	<i>Ne</i>
<i>Oblačno</i>	<i>Toplo</i>	<i>Visoka</i>	<i>Jak</i>	<i>Ne</i>

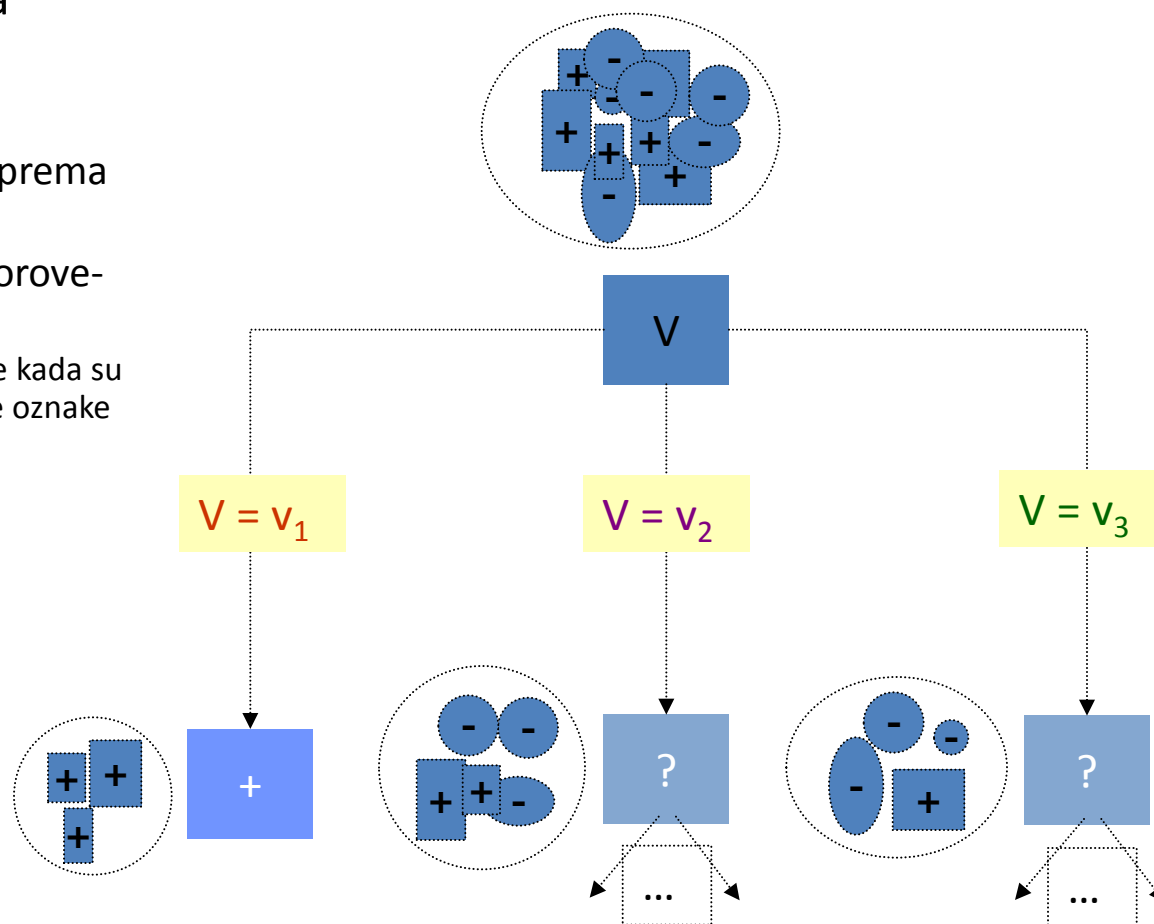
Stabla odlučivanja (decision trees)



(Izgled = Sunčano \wedge Vlažnost = Normalna) \rightarrow Igrati tenis = DA

Stvaranje stabla odlučivanja

- Uz dani skup primjera za učenje – T :
 - Izaberi varijablu V
 - Podijeli (split) primjere prema vrijednosti varijable V
 - Stvori (pod)stabla za čvorove-djecu rekurzivno;
 - Zaustavi proces podjele kada su svi primjeri u čvoru iste oznake



Algoritam stabla odlučivanja (ID3) (top-down induction of decision trees)

- **ID3 (*Primjeri*, *Ciljna varijabla*, *Varijable*)**
 - Kreiraj osnovni čvor stabla
 - Ako su svi *Primjeri* iste klase (+) => stablo je osnovni čvor sa oznakom (+)
 - Ako su svi *Primjeri* iste klase (-) => stablo je osnovni čvor sa oznakom (-)
 - Ako su sve *Varijable* iskorištene => vrati osnovni čvor sa oznakom najbrojnije klase u skupu *Primjeri*
 - Inače
 - Dok *Varijable* > 0
 - $V_i \leftarrow$ **odredi varijablu koja najbolje dijeli/klasificira** skup *Primjeri*
 - Za svaku vrijednost v_j od V_i
 - Dodaj novu granu ispod čvora, koja korespondira sa testom $V_i=v_j$
 - Neka su $\text{Primjeri}_-(V_i=v_j)$ onaj podskup *Primjeri* – koji zadovoljava $V_i=v_j$
 - Ako $\text{Primjeri}_-(V_i=v_j)=0$
 - Tada dodaj oznaku $c =$ *najčešće klase u skupu Primjeri*
 - Inače na novu granu dodaj novo pod-stablo
ID3($\text{Primjeri}_-(V_i=v_j)$, *Ciljna varijabla*, *Varijable* – $\{V_i\}$)

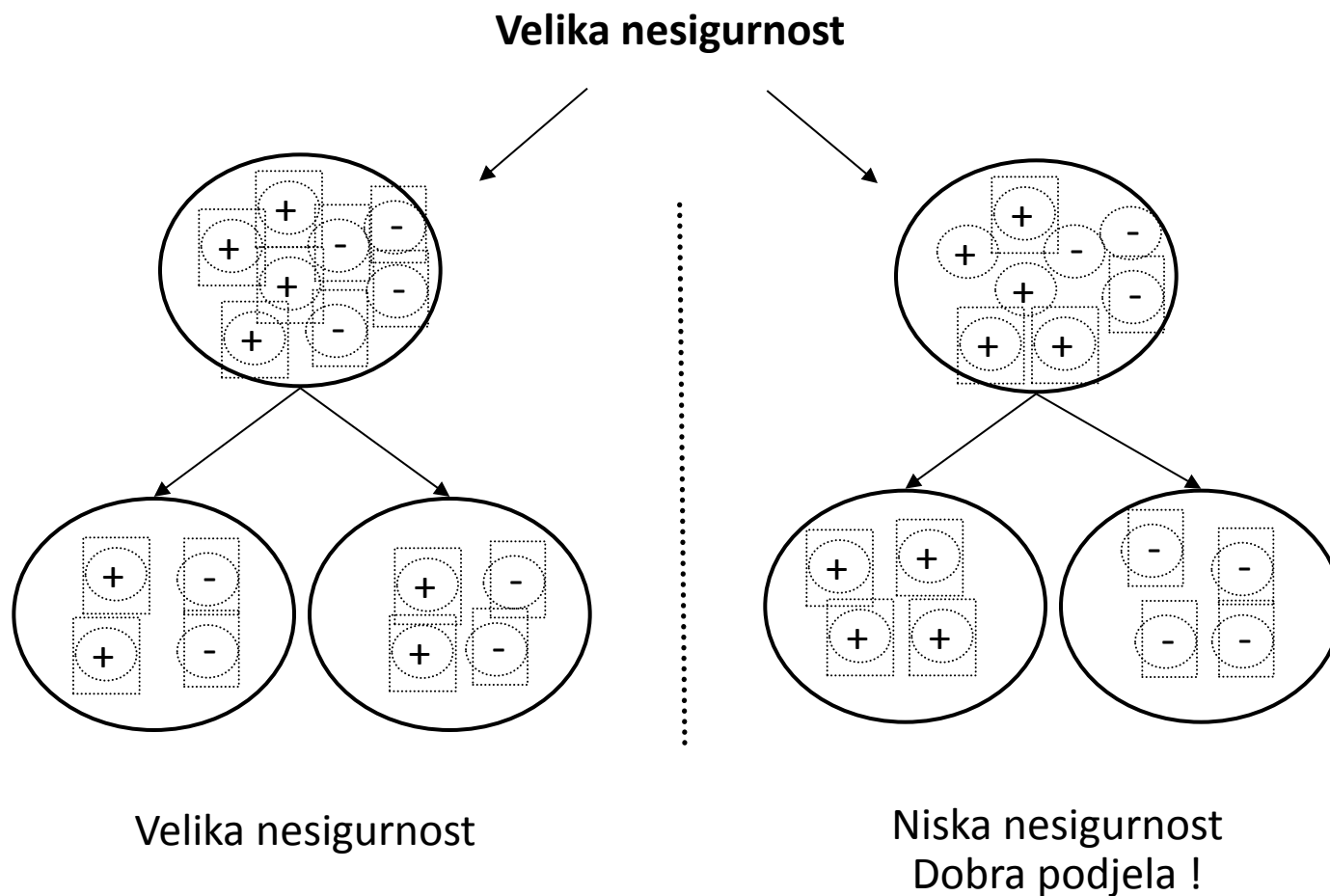
Kraj

ID3 (Quinlan)

- ID3 – algoritam za generiranje Stabla Odlučivanja
- Koristi (en. *information gain*) *porast informacije* u procesu izbora varijable u čvoru za podjelu primjera (split)
- Kasnije verzije C4.5 (=J48 – WEKA), C5

Porast informacije

(stvaranje homogenijih grana)



Entropija

- Entropija u skupu T primjera za binarni (1/0) klasifikacijski problem :

$$E(T) = -p_1 \log_2(p_1) - p_0 \log_2(p_0)$$

- p_1 udio primjera klase 1 u skupu T
- p_0 udio primjera klase 0 u skupu T

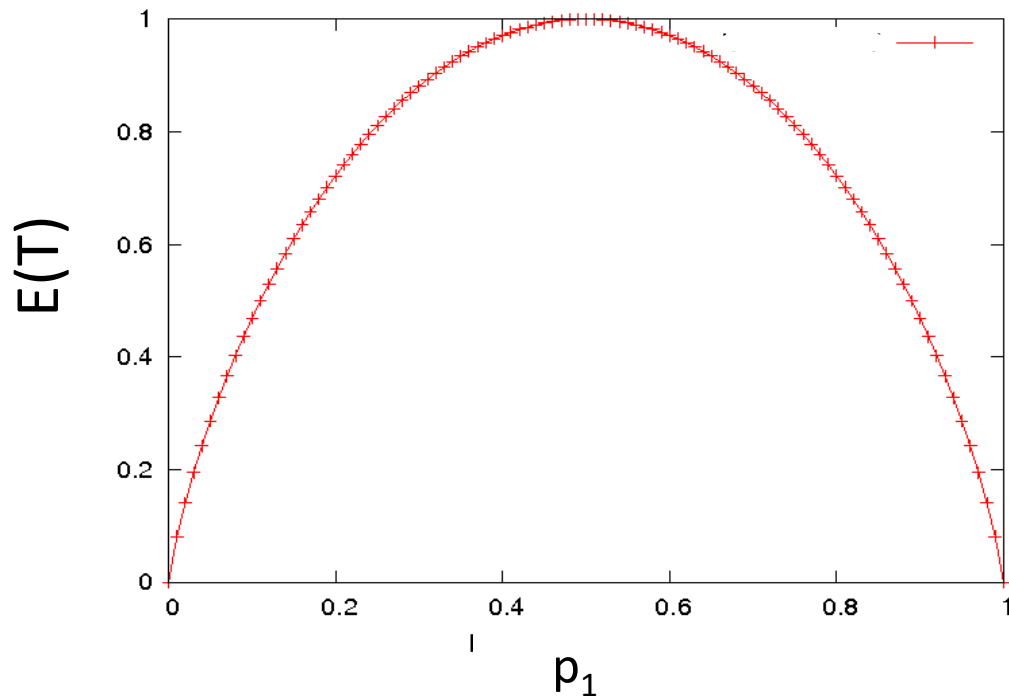
Za $p_1/p_0 = 1$, $E(T)=0$, za $p_1/p_0 = 0.5$ $E(T)=1$.

- Za probleme s više klasa :
$$E(T) = -\sum_i p_i \log_2(p_i)$$

Teorija informacija:

$E(X)$ - očekivani broj bitova potreban da se kodira slučajno odabrana vrijednost iz skupa X

Entropija



$$E(T) = -p_1 \log_2(p_1) - p_0 \log_2(p_0)$$

Porast informacije (en. information gain)

$$InfoGain(V) = Porin(V) = E(p_1, p_0) - \sum_{i=1}^{N_v} \frac{n_i}{N} \cdot E(p_{1,i}, p_{0,i})$$

- ❑ p_1, p_0 : broj (1,0) primjera u čvoru
 - ❑ $E(p_1, p_0)$: entropija uz dane p and n
 - ❑ N_v : broj mogućih vrijednosti (v_i) varijable V
 - ❑ n_i : broj primjera u grani i induciranoj split-om $V=v_i$
 - ❑ N : broj primjera u čvoru na koje radimo split
- ID3 bira varijablu s najvećim porastom informacije za podjelu !

Pimjer odabira atributa za split

Skup primjera u čvoru

x1	x2	x3	y
velik	težak	Plosnat	1
mali	težak	Plosnat	1
mali	težak	zaobljen	0
velik	lagan	Plosnat	0

$$InfoGain(V) = Porin(V) = E(p_1, p_0) - \sum_{i=1}^{N_v} \frac{n_i}{N} \cdot E(p_{1,i}, p_{0,i})$$

$$Porin(X1) = 1 - (0.5 \cdot 1 + 0.5 \cdot 1) = 0$$

$$Porin(X2) = 1 - (0.75 \cdot 0.918 + 0.25 \cdot 0) = \underline{0.311}$$

$$Porin(X3) = 1 - (0.75 \cdot 0.918 + 0.25 \cdot 0) = \underline{0.311}$$

Varijable s numeričkim vrijednostima

- Domena V je numerička:

$$\text{Domena}_V = \{10 - 250\}$$

- Kako je podijeliti ?

- Mogućnost: **diskretizacija**:

$$\text{Domena}_V = \{10 - 40, 40 - 120, 120 - 250\}$$

Problemi:

- Koja diskretizacija je najbolja?
- Kako razlikovati primjere u istom intervalu
 - Ako V predstavlja ocjene-bodove, nema razlike između đaka unutar intervala 40-120 !

Varijable s numeričkim vrijednostima

- Rješenje: dinamička podjela

Poredaj primjere prema vrijednosti V – sortiranje

a. Za svaki $x_i \in Domene(V)$

- Probaj podijeliti u 2 dijela: $(x \in d_1) \leq x_i$ i $(x \in d_2) > x_i$
- Izmeri *InfoGain/Porin* podjele

b. Samo između vrijednosti kod kojih dolazi do promjene klase

X_i	:	40	45	50	71	78	85	88	90	100
class:		0	0	1	1	1	0	0	0	1

Overfitting u stablima odlučivanja

- Naučiti stablo koje perfektно klasificira primjere iz skupa za učenje u principu vodi na lošije rezultate na novim primjerima !

2 razloga:

- Šum u podacima – koji onda nastojimo “naučiti”
- Algoritam prilikom generiranja stabla radi split-ove na malo podataka – što znači da su te odluke (statistički) nepouzidane

Prevenција overfittinga u stablima odlučivanja

2 osnovna pristupa

❑ Ograničavanje rasta (en. Prepruning):

- Zaustavljanje rasta stabla u toku konstrukcije kada je broj podataka premali za dobivanje pouzdane odluke

❑ Rezanje po završetku stabla (en. Postpruning):


- Konstruiramo kompletno stablo, a nakon toga režemo pod-stabla koja ne poboljšavaju točnost ukupnog stabla – pritom označujemo listove koji rezultiraju rezanjem sa oznakom klase koja prevladava
- Metode koje koristimo za to koja pod-stabla ćemo “odrezati”:
 - Unakrsna validacija
 - Izdvojimo unaprijed dio podataka za validaciju
 - Statistički testovi

Problemi

- ❑ Porast informacije preferira attribute sa mnogo vrijednosti

- ❑ Moguće rješenje

$$GainRatio(T, V) = Porin_norm(T, V) = \frac{Porin(T, V)}{SplitI(T, V)}$$

$$SplitI(T, V) = - \sum_{i=1}^c \frac{|T_i|}{|T|} \cdot \log_2 \frac{|T_i|}{|T|}$$


- ❑ Gdje su T_i podskupovi T koji nastaju splitom na V (primjeri s vrijednosti v_i)

Problemi

- ❑ Nedostajuće vrijednosti u primjerima ($x_i = \{13, 54, M, ?, Da, 345, \dots\}$)

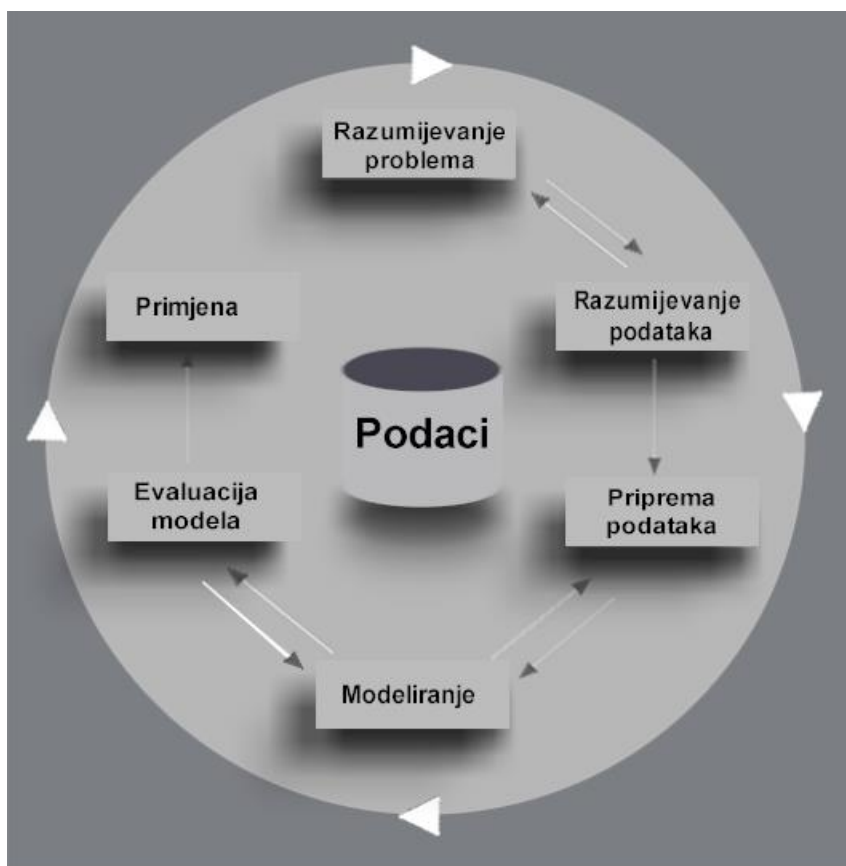
Mogući pristupi

- ❑ Ako čvor n “testira” varijablu V , za primjer koji ima nedostajuću vrijednost za V :
 - pridijeli najčešću vrijednost za primjere u koji su u čvoru n
 - pridijeli najčešću vrijednost za V koju imaju primjeri iste klase koji su u čvoru n
 - Dodijeli vjerojatnost p_i svakoj mogućoj vrijednosti v_i varijable V , te proslijedi udio p_i svakoj grani i koja ide iz čvora
- ❑ Kada se klasificiraju novi primjeri – koristi se ista shema kao i kod učenja !

Različite varijante algoritma

- ❑ Stabla odlučivanja za regresijske probleme - ciljne varijable s realnim vrijednostima (en. regression trees)
- ❑ Kombiniranje i testiranje više atributa u čvoru
- ❑ Različiti uvjeti za split

- K-nn, Nbayes i Stabla Odlučivanja
 - vrlo često se koriste u dubinskoj analizi podataka
- DM – kao standardni proces (CRISP-DM)



	Važnost	Trajanje
Razumijevanje problema i podataka	80 %	20%
Priprema podataka, modeliranje i evaluacija	20 %	80%

Literatura:

Machine learning,

T. Mitchel (djelovi ch. 3,6,8)

The Elements of Statistical Learning

Hastie, Tibshirani, Friedman (ch. 2)