Strojno učenje
5
Osnovni algoritmi

Tomislav Šmuc

Sadržaj

Osnovni algoritmi za učenje pod nadzorom

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

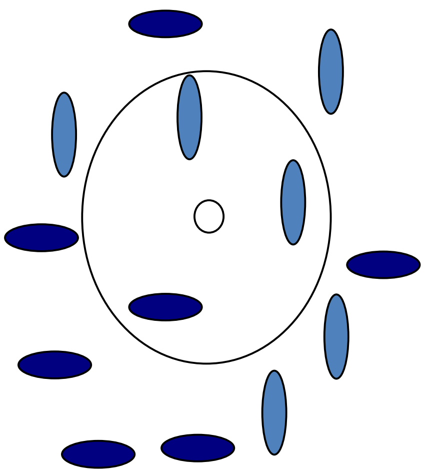
"Memorijski" klasifikator – (ROTE-Learner)

- U memoriji su svi primjeri dostupni u trenutku učenja ($\langle \boldsymbol{x}_i, y_i \rangle \in \boldsymbol{T}$)
- Klasifikacija novih instanci – traži isti takav primjer u memoriji ($\boldsymbol{x}_t = \boldsymbol{x}_i \mid \boldsymbol{x}_i \in \boldsymbol{T}$):
    - 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)

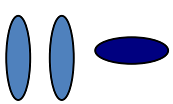
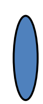
- Malo poopćenje “memorijskog” klasifikatora
- Klasificira koristeći princip analogije:

  “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)

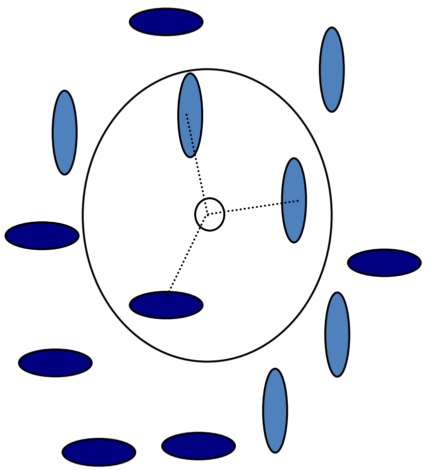


K-nn algoritam

- Izračuna <u>udaljenost</u> između novog primjera $\mathbf{x}_t$ i svih primjera iz skupa za učenje ***T***
- Odredi k-najbližih susjeda $\mathbf{x}_t$ iz ***T***
- Pridjeli $\mathbf{x}_t$ klasu $\mathbf{c}_i$ koja je najčešća za k-najbližih susjeda

3-nn(○) =  => c(○) = 

K-nn algoritam (k – najbližih susjeda)



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):

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

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(Pero, Matilda)= $\sqrt{}$ *(x*$_1$*(Pero)-x*$_1$*(Matilda))*^{2} *+ (x*$_2$*(Pero)-x*$_2$*(Matilda))*^{2} *+ …*

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

???

K-nn algoritam (k – najbližih susjeda)

K-nn – problem određivanja udaljenosti između primjera

Kontinuirane varijable / atribute

$$d_2(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_{j=1}^{n} (x_{1,j} - x_{2,j})^2}$$ Euklidska

$$d_1(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^{n} \left| x_{1,j} - x_{2,j} \right|$$ Manhattan / City block / Taxicab norm

Familije mjera (kontinuirane varijable)

Minkowski $$d_p(\mathbf{x}_1, \mathbf{x}_2) = \sqrt[p]{\sum_{j=1}^{n} (x_{1,j} - x_{2,j})^p}$$

Druge familije

- $\mathbf{L_1}$ (Sorensen, Canberra, Lorentzian…)
- $\mathbf{L_2}$ (Euclidean, Pearson χ^2….)
- **inner-product** (Jaccard, Cosine, Chord…)
- ……..

K-nn algoritam (k – najbližih susjeda)

K-nn – problem određivanja udaljenosti između primjera

Za kategoričke varijable / atribute

X(m-kat.vrijednosti)

=> **X** (**X** – m dimenzionalna Bernoulli var.)

$Dom(\mathbf{X}) = \{a_1, a_2, \dots, a_m\}$

Range (**X**)={0,1}^{m}

Za $X=a_i$

$$\boldsymbol{X} = \boldsymbol{e_i}$$

$$\boldsymbol{e_i} = (0,0,..0,1,0,..,0)^{\boldsymbol{T}}$$

$$(e_{i,j} = 0 \;\; za \;\; j = 1,..i-1; j = i+1, \dots, m)$$

K-nn – problem određivanja udaljenosti između primjera

Za dva primjera opisana s *d* kategoričkih (Bernouli transf.) varijabli

$$\mathbf{x}_i = \begin{pmatrix} \mathbf{e}_{1i_1} \\ \vdots \\ \mathbf{e}_{di_d} \end{pmatrix} \qquad \mathbf{x}_j = \begin{pmatrix} \mathbf{e}_{1j_1} \\ \vdots \\ \mathbf{e}_{dj_d} \end{pmatrix}$$

Sličnost *s* :

$s(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j = \sum_{k=1}^{d} (\mathbf{e}_{ki_k})^T \mathbf{e}_{kj_k}$ => *broj var. kod kojih se vrijednosti slažu*

$$d - s \Rightarrow \ broj\ var.kod\ kojih\ se\ vrijednosti\ ne\ slažu$$

K-nn – problem određivanja udaljenosti između primjera

Udaljenosti/sličnosti - za kategoričke varijable / atribute

- $d_H(x, y) = d - s \quad Hamming\ (udaljenost)$

- $d_J(x, y) = \frac{s}{2d-s} \quad Jaccard\ (sličnost)$

- $d_{cos}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x_i}\| \|\mathbf{x}_j\|} = \frac{s}{d} \quad Cosinus\ (sličnost)$

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

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

K-nn – poboljšanja

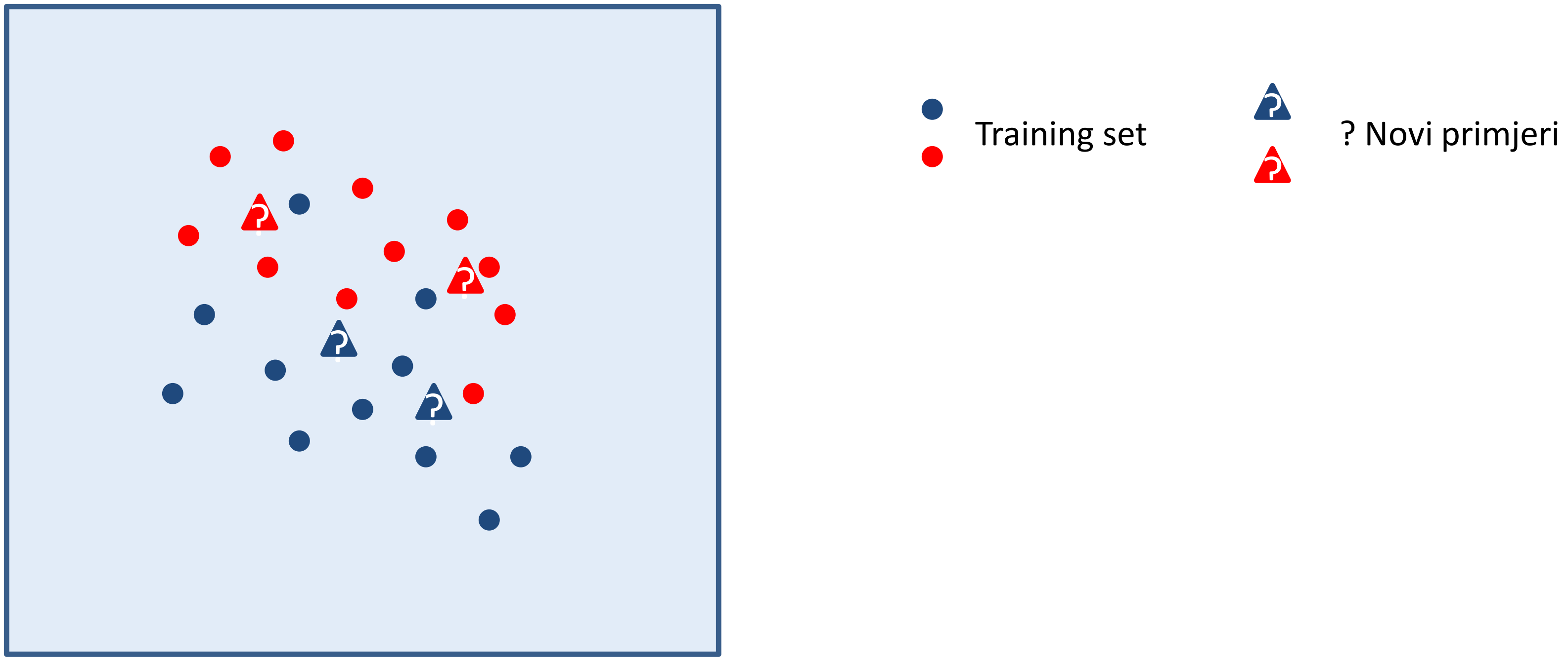
- Težinsko određivanje - uzima u obzir udaljenost k-susjeda prema novom primjeru

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

gdje je

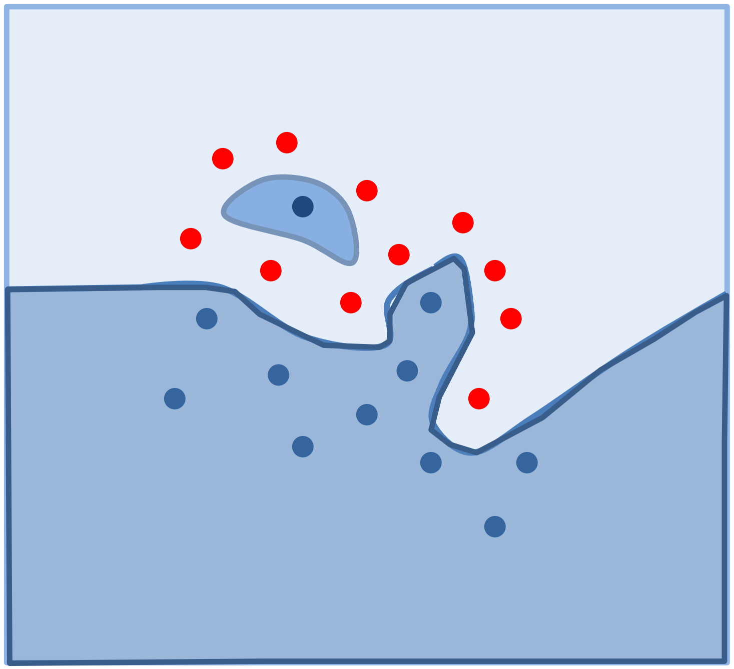
$$w_i \equiv \frac{1}{d(\mathbf{x}, \mathbf{x_i})}$$

K-nn – overfitting

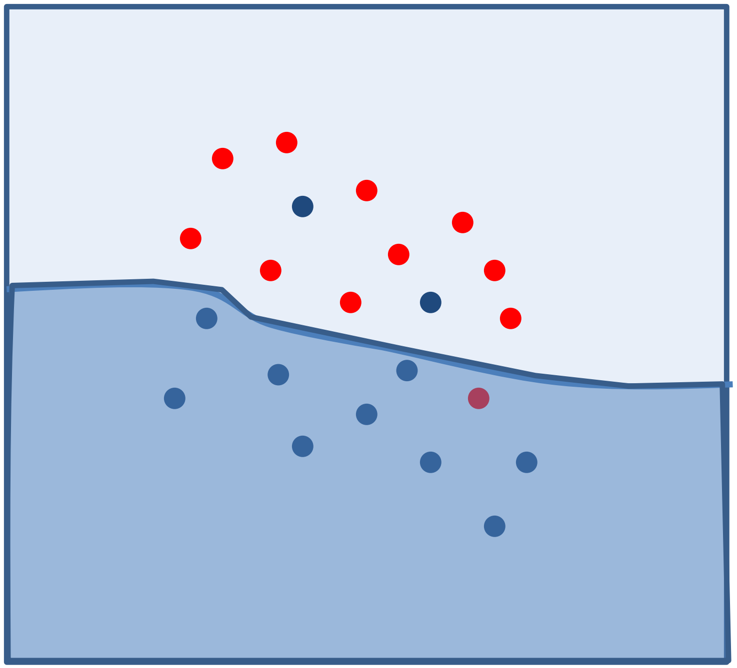


K-nn algoritam (k – najbližih susjeda)

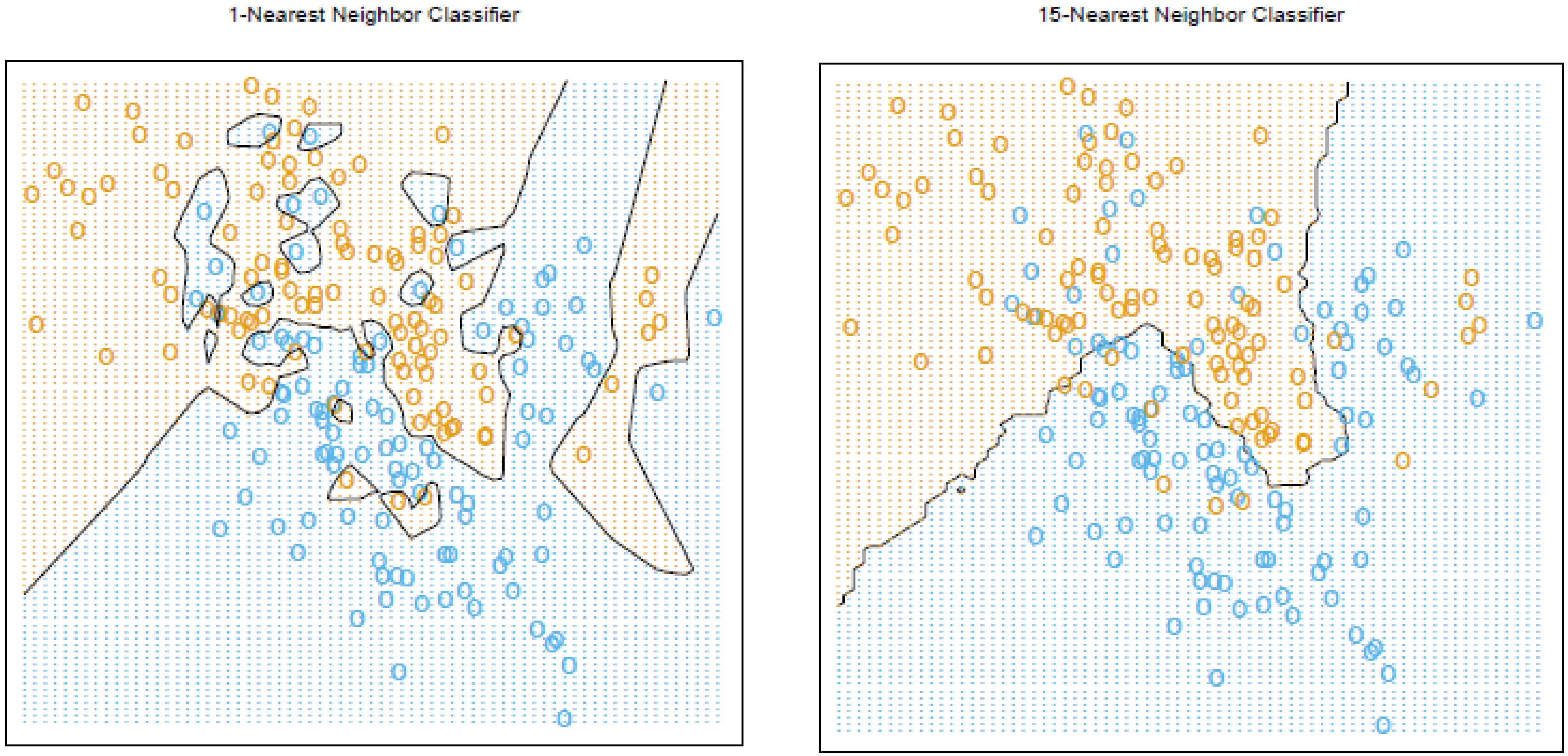
1-nn (približne granične linije)



5-nn (približne granične linije)



K-nn algoritam (k – najbližih susjeda)



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

Karakteristike k-nn klasifikatora

Prednosti

- Jednostavan za implementaciju
- Gotovo optimalan za ($N \rightarrow \infty$)

  $Er_{OBayes} < Er_{1\text{-}nn} < 2 * Er_{OBayes}$

- 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:
  - “glađe” 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

Q: K-nn algoritam možemo jednostavno preinačiti za regresijske probleme.
Kako izgleda $\hat{f}(\mathbf{x})$ u tom slučaju ?

Klasifikacija prema Bayesu

MAP (Maximum a posteriori) klasifikacija

Optimalni Bayes klasifikator

Naivni Bayes klasifikator

Probabilistički pristup

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

Probabilistički pristup

Najbolja hipoteza ≈ 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.

MAP hipoteza

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

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, <u>ustvari naš konačni cilj je dobiti najbolji rezultat</u> ! 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} ?

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

Bayes-optimalna klasifikacija

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 !)

Bayesov klasifikator

Za potrebe klasifikacije:

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

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

$c_j \in C$ – klasa

x - vrijednost atributa

(pretpostavimo da se radi o kategoričkoj varijabli)

Učenje:

- Odrediti $P(\mathbf{x} \mid c_j)$, $P(c_j)$ 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* $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$

(pretpostavka da se radi o kategoričkim varijablama)

Bayesov klasifikator

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

uz Bayesov teorem

$$c_{MAP} = \arg\max_{c_j \in C} \frac{P(x_1, x_2, \ldots, x_n \mid c_j) \cdot P(c_j)}{P(x_1, x_2, \ldots, x_n)}$$

$$= \arg\max_{c_j \in C} P(x_1, x_2, \ldots, x_n \mid c_j) \cdot P(c_j)$$

Odrediti

$P(c_j)$ - je lako

$P(x_1, x_2, \ldots, x_n \mid c_j)$ – u praksi nemoguće !

Potrebno je imati vrlo, vrlo velik skup primjera !

Naivni Bayesov klasifikator

Osnovna pretpostavka

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

$\mathbf{x_i}$ uvjetno nezavisno o $\mathbf{x_j}$ $\forall$ i≠j !

Naivni Bayesov klasifikator

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

c je uvjetno nezavisno od x_2 uz zadan x_1 ako je distribucija <u>vjerojatnosti nad *c* neovisna o vrijednosti x_2</u>, uz zadane vrijednosti x_1.

<u>Što nam to donosi ?</u>

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

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

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

$N_{NBC}(P) \approx |C| * [\ |x1| + |x2| + |xn|\ ]$

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

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

⇓

P(Spol=m | Klasa= 1) = 1/2

P(Spol=m | Klasa= 0) = 1/3

P(Brak=Da | Klasa= 1) = ¾

P (Brak=Da | Klasa= 0) = 1/3

klasifikacija

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

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

problemi

Kada je stvarna vjerojatnost

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

a imamo relativno mali broj primjera za učenje |T|

=> Vrlo je vjerojatno da ćemo dobiti procjenu

$P(x_i = x_{i,j} \mid c = c_k) \approx \kappa_{ijk} = 0$ => ovaj će faktor dominirati pri izračunu $c(\boldsymbol{x}^{novi})$

Rješenje:

m-estimate (drugi naziv - Laplacian smoothing)

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

p - naša procjena vjerojatnosti; (~ p=1/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 $P(x_j^{novi} \mid c = c_k)$ prema:

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

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 !

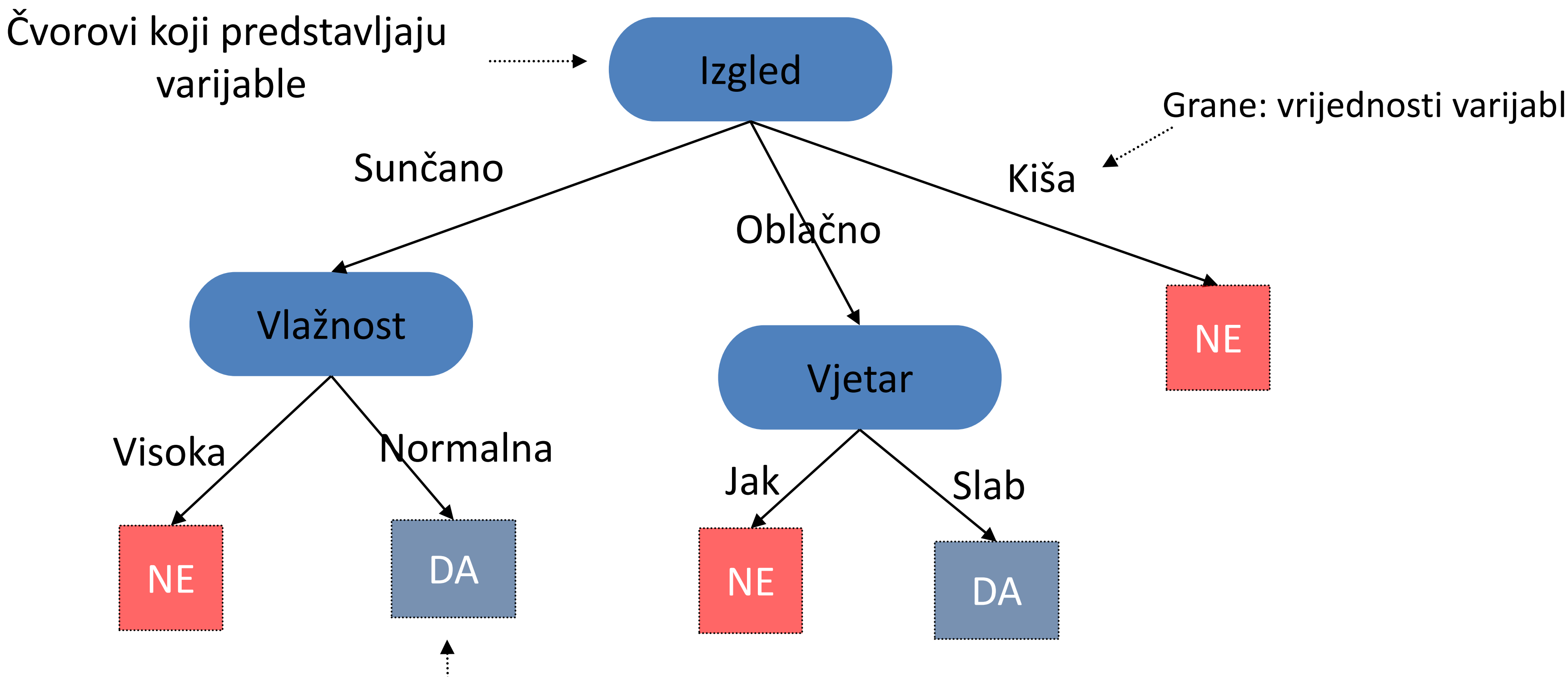
- Želimo naučiti prepoznavati:

  "kada je dobar dan za igranje tenisa" – igrati ili ne ?

- Mjerenja:
  - Varijable i njihove vrijednosti + oznake

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

Stabla odlučivanja (decision trees)

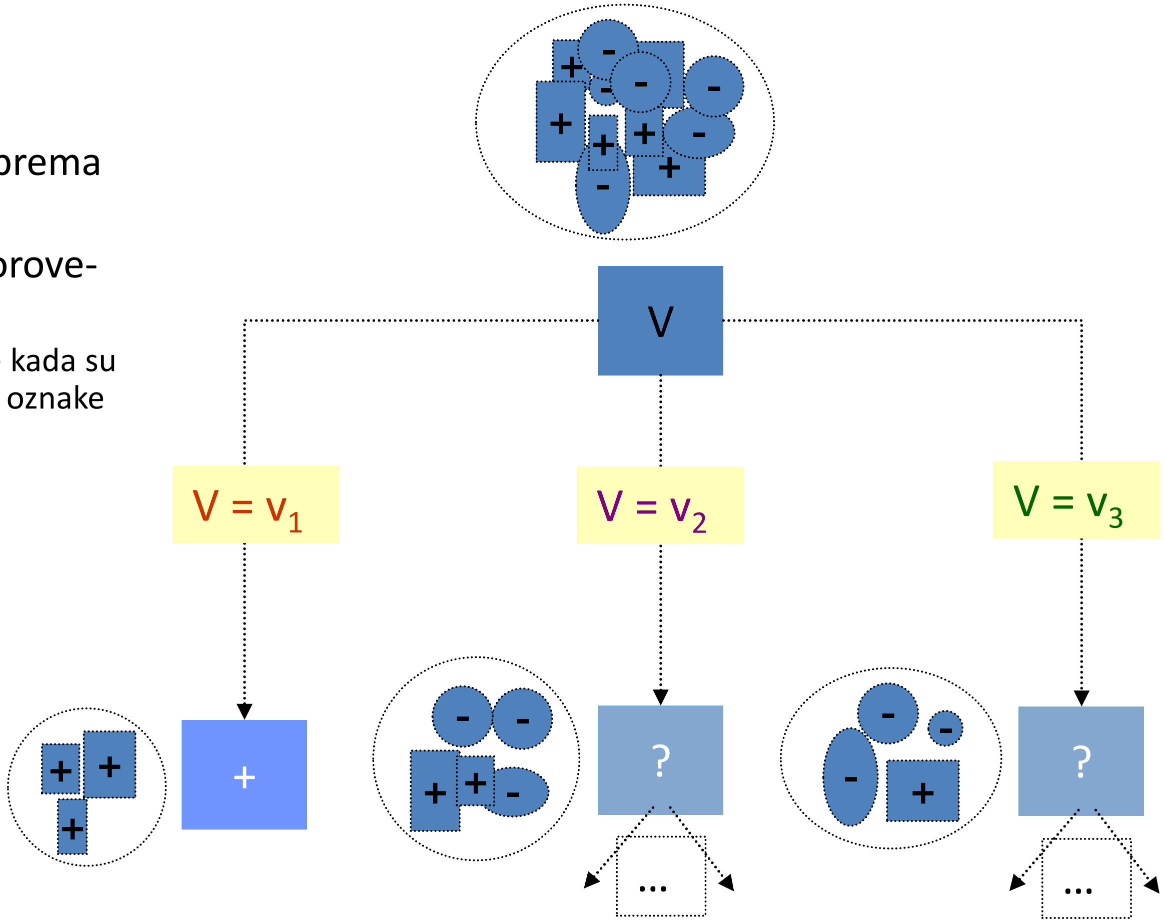


Obojeni čvorovi – **listovi stabla**
=> odluke

(Izgled = Sunčano $\wedge$ Vlažnost = Normalna) → 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



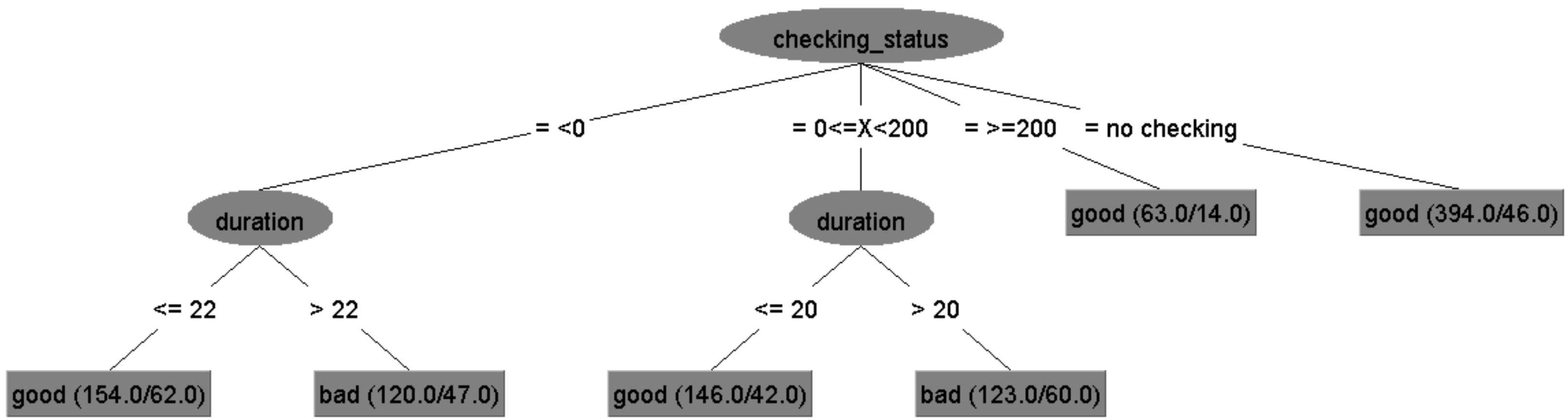
Algoritam stabla odlučivanja (ID3)
(top-down induction of decision trees)

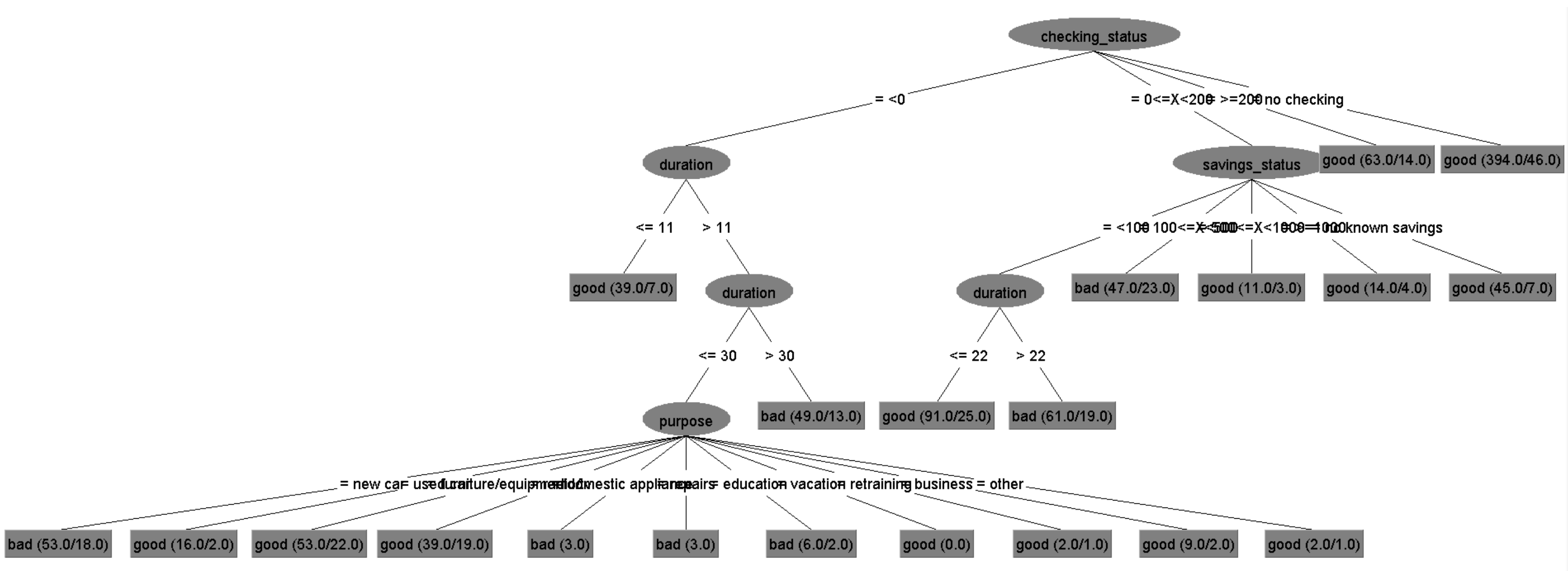
- ID3 (*Primjeri, Ciljna varijabla, Varijable)*
  - Kreiraj osnovni čvor stabla
  - Ako su svi *Primjeri* iste klase (+) => stablo je osnovni čvor sa oznakom (+)
  - Ako su svi *Primjer*i iste klase (-) => stablo je osnovni čvor sa oznakom (-)
  - Ako su sve *Varijable* iskorištene => vrati osnovni čvor 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 *Primjeri_*($V_i=v_j$) onaj podskup *Primjeri* – koji zadovoljava $V_i=v_j$
      - Ako *Primjeri_*($V_i=v_j$)=0
        - Tada dodaj oznaku *c = najčešće klase u skupu Primjeri*
        - Inače na novu granu dodaj novo pod-stablo

          ID3(*Primjeri_*($V_i=v_j$) *, Ciljna varijabla , Varijable –* $\{V_i\}$ *)*

    Kraj


checking_status
= <0
= 0<=X<200
= >=200
= no checking
duration
duration
good (63.0/14.0)
good (394.0/46.0)
<= 22
> 22
<= 20
> 20
good (154.0/62.0)
bad (120.0/47.0)
good (146.0/42.0)
bad (123.0/60.0)



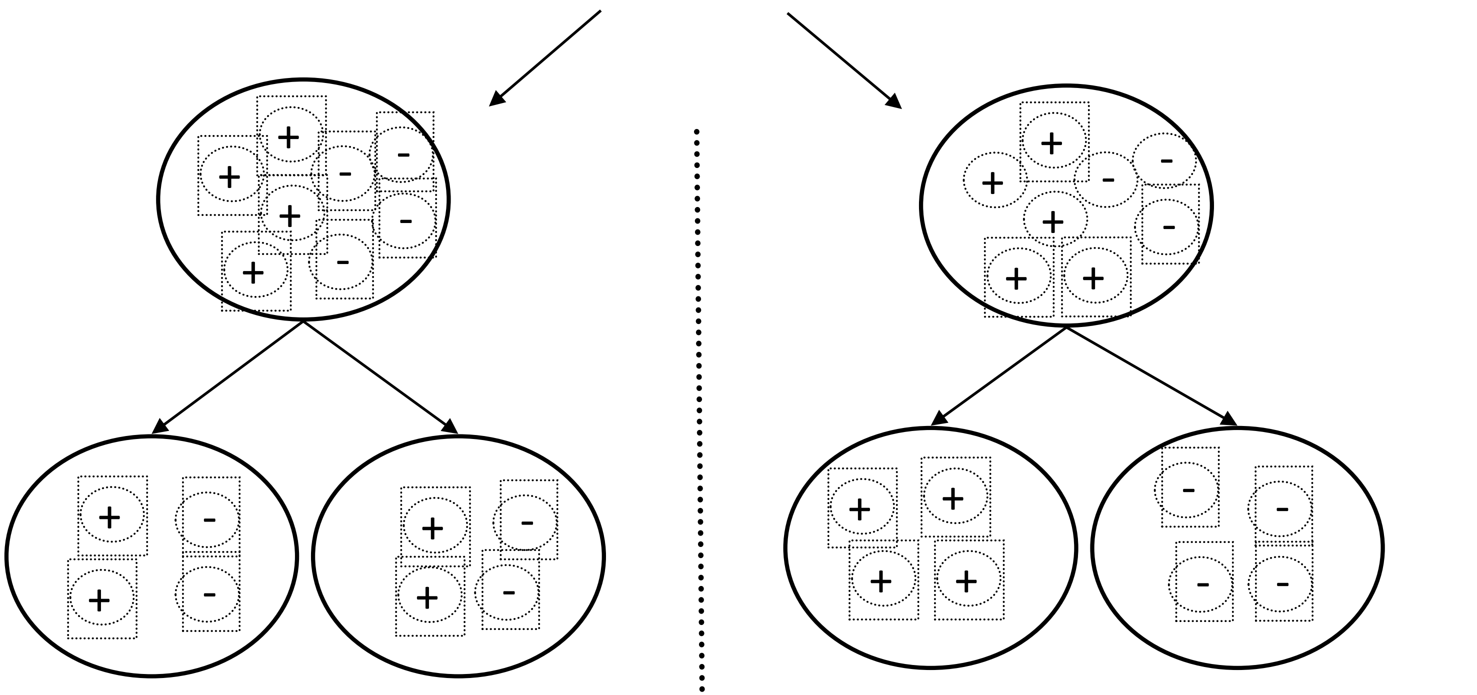
checking_status
= <0
= 0<=X<200
= >=200
= no checking
duration
savings_status
good (63.0/14.0)
good (394.0/46.0)
<= 11
> 11
= <100
= 100<=X<500
= 500<=X<1000
= >=1000
= no known savings
good (39.0/7.0)
duration
duration
bad (47.0/23.0)
good (11.0/3.0)
good (14.0/4.0)
good (45.0/7.0)
<= 30
> 30
<= 22
> 22
purpose
bad (49.0/13.0)
good (91.0/25.0)
bad (61.0/19.0)
= new car
= used car
= furniture/equipment
= radio/tv
= domestic appliance
= repairs
= education
= vacation
= retraining
= business
= other
bad (53.0/18.0)
good (16.0/2.0)
good (53.0/22.0)
good (39.0/19.0)
bad (3.0)
bad (3.0)
bad (6.0/2.0)
good (0.0)
good (2.0/1.0)
good (9.0/2.0)
good (2.0/1.0)

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)

Velika nesigurnost



Velika nesigurnost

Niska nesigurnost
Dobra podjela !

Entropija

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

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

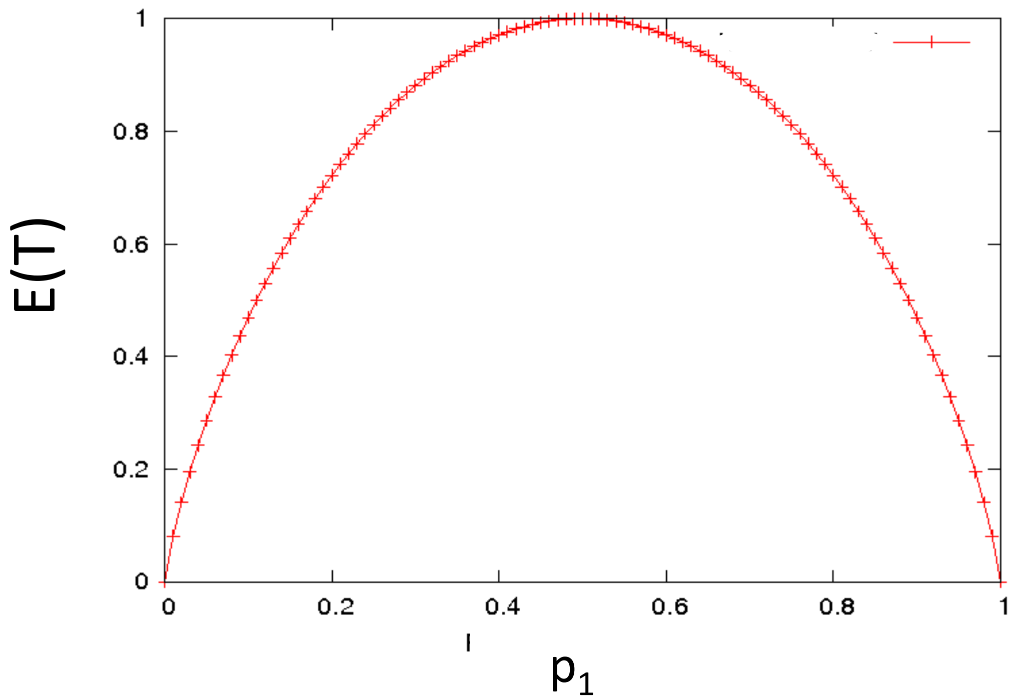
- p_1 udio primjera klase 1 u skupu *T*
- p_0 udio primjera klase 0 u skupu *T*

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–(0.5·1 + 0.5·1) = 0
Porin (X2) =1–(0.75·0.918 + 0.25·0) = <u>0.311</u>
Porin (X3) =1–(0.75·0.918 + 0.25·0) = <u>0.311</u>

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 $(x \in d_2) > x_i$
  - Izmjeri *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 perfektno 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) nepouzdane

Prevencija 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 atribute 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,...}*)

Mogući pristupi

- Ako čvor *n* "testira" varijablu *V*, za primjer koji ima nedostajuću vrijednost za V:
  a) pridijeli najčešću vrijednost za primjere u koji su u čvoru *n*
  b) pridijeli najčešću vrijednost za *V* koju imaju primjeri iste klase koji su u čvoru *n*
  c) Dodijeli vjerojatnost p_i svakoj mogućoj vrijednosti v_i varijable *V,* te proslijedi udio p_i svakoj grani *i* koja ide iz čvora

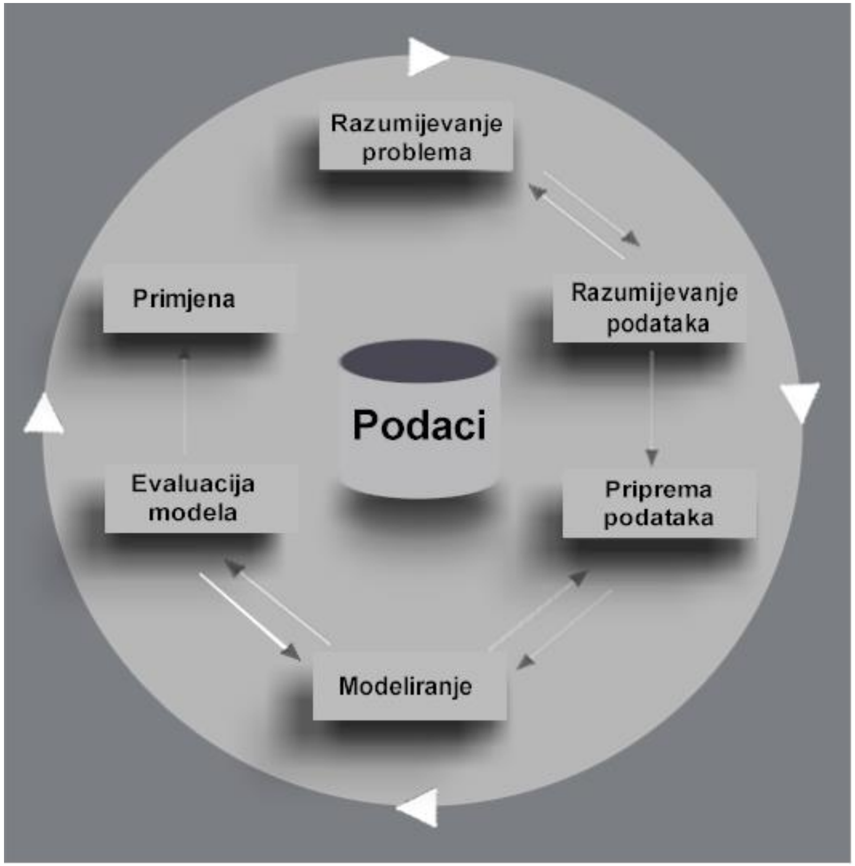
- Kada se klasificiraju novi primjeri – koristi se ista shema kao i kod učenja !

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, Naivni Bayes i Stabla Odlučivanja vrlo često se koriste u dubinskoj analizi podataka
- bazični algoritmi u strojnom učenju (*baseline* modeli)

DM – kao standardni proces (CRISP-DM)



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

Literatura

Literatura:

Machine learning (ch 3,6,8); T. Mitchel

The Elements of Statistical Learning (ch 2); Hastie, Tibshirani, Friedman

Data Mining and Analysis: Fundamental Concepts
and Algorithms (ch 3); Zaki & Meira