1 Uvod

Prišel bo čas, ko bomo morali pozabiti vse, kar smo se naučili.
(Ramana Maharshi)

- Strojno učenje (machine learning)
- odkrivanje zakonitosti v podatkovnih bazah (knowledge discovery in databases)
- podatkovno rudarjenje (data mining)

Osnovni princip strojnega učenja je avtomatsko opisovanje (modeliranje) pojavov iz podatkov.

Odkrivanje znanja iz podatkov

1. Razumevanje problemskega področja:
   - metodologije,
   - cilji,
   - kriteriji uspešnosti.
2. Razumevanje podatkov:
   - spoznavanje,
   - preverjanje kvalitete,
   - iskanje izjem.
3. Priprava podatkov:
   - zbiranje,
   - vrednotenje, vizualizacija,
   - poenotenje,
   - čiščenje, filtriranje,
   - transformiranje (diskretizacija, preslikave...).
4. Modeliranje, strojno učenje:
   - izbor ustrezne metode za strojno učenje,
   - gradnja in interno vrednotenje modelov,
   - ponavljanje postopkov.
5. Vrednotenje rezultatov:
   - vrednotenje glede na različne kriterije,
   - sprejem konkretnih modelov,
   - ocena celotnega procesa in odločitev o naslednjem koraku.
6. Uporaba:
   - kdo in kdaj bo uporabljal rezultate,
   - problem prenosljivosti modela (na nove podatke),
   - praktična uporaba znanja.

Rezultat učenja iz podatkov so lahko

- pravila,
- funkcije,
- relacije,
- sistemi enačb,
- verjetnostne porazdelitve ipd.

Naučeni modeli poskušajo razlagati podatke, iz katerih so bili modeli zgenerirani, in se lahko uporabijo za odločanje pri opazovanju modeliranega procesa v bodočnosti (napovedovanje, diagnosticiranje, nadzor, preverjanje, simulacije itd.).

Pregled metod strojnega učenja

- klasifikacija:
  - Odločitvena drevesa in pravila
  - Bayesov klasifikator
  - Klasifikator z najbližjimi sosedi
  - Diskriminantne funkcije:
    - Fisherjeva linearna dis. funkcija,
    - logistična regresija,
    - metoda podpornih vektorjev (SVM),
  - Umetne nevronske mreže
- regresija:
  - Regresijska drevesa
  - Linearna regresija
  - Lokalno utežena regresija:
  - Metoda podpornih vektorjev
  - Umetne nevronske mreže
- asociacije in logične relacije:
  - asociativne nevronske mreže,
  - povezovalna pravila,
  - induktivno logično programiranje,
- odkrivanje zakonitosti - sistemi enačb,
- nenadzorovano učenje - razvrščanje,
- spodbujevano učenje.

Aplikacije strojnega učenja

- Diagnostika proizvodnega procesa,
- Medicinska diagnostika in prognostika,
- Ocenjevanje zavarovancev in posojilojemalcev,
- Klasifikacija slik (5×10^7 galaksij, 2×10^9zvezd)
- Napovedovanje strukture kemičnih spojin in proteinov,
- Igranje iger.

2 Učenje in inteligenca

1. Kaj je učenje
    - Definicija učenja
    - Umetna inteligenca
2. Naravno učenje
    - Prirojeno in naučeno
    - Spomin
    - Vrste naravnega učenja
3. Inteligenca in zavest: znanost in filozofija
4. Zakaj strojno učenje?

2.1 Kaj je učenje

Učenje pomeni adaptivne spremembe v sistemu, ki mu omogočajo bolj učinkovito opravljanje iste naloge.

Herbert A. Simon

- učenje = pridobivanje znanja
- naravno učenje : strojno učenje
- *programirano učenje*, računalniško podprto

Inteligenca:
sposobnost prilagajanja okolju in reševanja problemov.

Znanje je dano vnaprej ali pa je rezultat učenja.

inteligenca =
strojna oprema + predznanje + učenje + ?.

Umetna inteligenca

Razvoj sistemov, ki se obnašajo inteligentno in ki so sposobni reševati relativno težke probleme.

- strojno učenje,
- predstavitev znanja,
- računalniško zaznavanje,
- razumevanje naravnega jezika,
- avtomatsko sklepanje in dokazovanje izrekov,
- igranje iger,
- logično programiranje,
- kvalitativno modeliranje,
- razvoj ekspertnih sistemov,
- hevristično reševanje problemov,
- robotika,
- kognitivno modeliranje.

2.2 Naravno učenje

Učenje je napredek dejavnosti po izkušnji.
Zavest se otroku razvije z učenjem.
Učljiva je morala in inteligenca.

Anton Trstenjak

- novorojenček se uči gledati, poslušati,
- dojenček se uči pomena besed, povezave med vidom in tipom, motorike
- otrok se uči hoditi, govoriti besede in stavke,
- učenec se uči brati, pisati, računati, abstraktno razmišljati, logično sklepati,
- dijak pridobiva opisno znanje in hevristike za reševanje problemov;
- študent si z učenjem prestrukturira svoje znanje, pridobiva specializirano opisno znanje in specializirane hevristike
- na delovnem mestu z delom pridobiva izkušnje

- *psihologija učenja*
- *pedagoška psihologija*

Prirojeno in naučeno

- zorenje
- kritična obdobja za učenje
- prirojene veščine: nagoni
- Čim višje na razvojni lestvici se nahaja živalska vrsta, tem pomembnejšo vlogo v njenem življenju igra učenje. Čim višji je končni nivo učne zmožnosti, ki ga neka vrsta doseže, tem počasnejše je učenje v otroštvu.
- črv 13 nevronov, čebela okoli 900 nevronov,
- gorilo so naučili razumeti več kot tisoč besed.

# nevronov	10^{11}
# nevronov v možganski skorji	2×10^{11}
# poslanih impulzov na nevron	10 impulzov/sek
hitrost pretoka inf. po kanalih	30 bit/sek
# kemijskih reakcij v možganih	$10^5 - 10^6/sek$
# povezav / nevron	10^4
vseh sinaps	$10^{14} - 10^{15}$
# bolečinskih točk na koži	1.2×10^6
# živčnih vlaken pri očesnem živcu	1.2×10^6
kratkoročni spomin (zavest)	7 kazalcev (naslovov)
dolgoročni spomin - samo sinapse	$> 10^{15}$ *bitov*
dostop do dolgoročnega spomina	2 sek
zunanja optim. delovna temp.	$18°C$
# aminokislin	20
dolžina verige aminokislin	več tisoč
# možnih različnih proteinov	$> 20^{1000}$
genski zapis	$> 10^9$ bitov = 1G bit
# proizvedenih proteinskih molekul	15000/sek/nevron

Spomin

1. Ustvarjanje novih povezav z drugimi nevroni
2. Spreminjanje jakosti povezav na sinapsah
3. Proizvodnja proteinov (univerzalna koda)

- človek si zapomni prav vse, kar se mu pripeti v življenju (gospod "S")
- problematično je "naslavljanje spomina"
- racionalizacija (podlaga generalizaciji)

Vrste naravnega učenja:

- glede kompleksnosti učenja,
- glede gradiva, ki se ga učenec uči,
- glede strategije učenja.

Vrste učenja po kompleksnosti učnega procesa:

Vtiskavanje

Pogojevanje ali asociiranje: pogojni refleks; asociacija

Verjetnostno učenje

Zapominjanje

Učenje s poskusi in napakami

Posnemanje: vzročno povezovanje in sklepanje

Učenje z razumevanjem in vpogledom: zapominjanje, abstraktno (simbolično) mišljenje, logično sklepanje in vzročno povezovanje; vpogled in integracija

Vrste učenja glede na gradivo učenja:

- senzorno učenje,
- motorično učenje in
- besedno (semantično) učenje.

spontana generalizacija

diferenciacija (specializacija)

Pojmi so razredi ali kategorije izkušenj, ki jih iz določenih razlogov enako obravnavamo oziroma imajo neko skupno svojstvo. Razlikujejo se po

- konkretnosti/abstraktnosti in
- po zapletenosti (npr. konjunktivni in disjunktivni).

Večina učenja pri otrocih sestoji iz pridobivanja že obstoječih pojmov, tako da jim učitelj podaja **primere in protiprimere ter delni opis pojma**.

Le redko mora učenec tvoriti **nove pojme**, kar je seveda težavnejša naloga.

Iskanje v širino

Iskanje v globino

Konservativno osredotočenje: Hipoteza postane prvi (pozitivni) učni primer. Zatem učenec sistematično spreminja eno spremenljivko, dokler ne dobi konsistentno hipotezo.

Osredotočeno ugibanje: Hipoteza postane prvi (pozitivni) učni primer. Zatem učenec spreminja več spremenljivk naenkrat.

Razumeti sebe je kot poskusiti ugrizniti svoje zobe.

Alan Watts

Znanje je pomembno. Toda še veliko pomembnejša je njegova koristna uporaba. Ta je odvisna od srca in uma človeka.

Dalaj Lama

Identičnost med človekom in strojem se ne doseže s tem, da prenesemo človeška svojstva na stroj, marveč s tem, da prenesemo mehanične omejitve na človeka.

Mortimer Taube

Učenje, inteligenca, zavest

inteligenca = strojna oprema + predznanje + učenje + ?.

- Količina inteligence,
- Zavest.

Zavest

Če odvzameš vse misli, ostane čista zavest.
Ramana Maharshi

- zavedanje samega sebe, razlikovanje sebe od drugih, zavedanje svojih težav, svojih nalog in svojih odgovornosti

- kaj je zavest?

- nekateri kvantni fiziki povezujejo zavest s kolapsom valovne funkcije,

- materialistična razlaga: zavest je rezultat kompleksnosti sistema in je nastala v nekem časovnem trenutku evolucije.

Mentalna vsebina	zavedanje sebe	
	DA	NE
DA	budno stanje	sanjanje
NE	meditacija	spanje

Več nivojev zavesti:

- čista zavest,
- nadzavest ali spremenjeno stanje zavesti,
- običajna zavest (budno stanje),
- podzavest
- nezavest (najverjetneje samo truplo).

Meje simbolične izračunljivosti, izpeljivosti, opisljivosti

Resnice se ne da poučevati.
Osho

Teorija izračunljivosti: skoraj zanemarljiv delež problemov, ki si jih lahko formalno zastavimo, rešljiv algoritmično.

Danes uporablja znanost za opisovanje realnosti:
- matematična logika,
- računalniški programski jeziki,
- rekurzivne funkcije in
- formalne gramatike.

Vsi ti formalizmi so po izrazni moči med seboj enakovredni.

Diskretni svet: Q - **racio**nalna števila
Zvezni svet: R - **real**na števila

$$|R| = 2^{|Q|} >> |Q|$$

Vsi formalizmi imajo enake omejitve: lahko (delno) opišejo pojave znotraj diskretnega sveta in tako rekoč zanemarljivo majhen del zveznega sveta.

Torej, če je svet dejansko zvezen, potem je najverjetneje neopisljiv.

Nekaj pomembnih dosežkov umetne inteligence:

- Lenatov (1983) AM (Automatic Matematician),
- uspehi pri igranju iger, kot so dama in šah,
- umetne nevronske mreže kot model možganov,
- generiranje novega znanja iz podatkov.

Naučljivost je tesno povezana z izračunljivostjo in vse omejitve v zvezi z izračunljivostjo veljajo tudi za naučljivost.

Možnost umetne inteligence

Učljiva je morala in inteligenca.
Anton Trstenjak

Za razumevanje možnosti umetne inteligence, so pomembni vidiki:

Vpliv učenja na inteligenco

Hitrejši je inteligentnejši

Meje inteligence Turingovega stroja

Oponašanje inteligence

Torej, če izpustimo zavest, je v principu lahko stroj dovolj inteligenten, da nam pričara občutek umetne inteligence.

Seveda pa takemu stroju manjka zavest.

(Ne)zmožnost umetne zavesti?

- Inteligenca: objektivno merljiva
- Zavest: subjektivna, objektivno nepreverljiva
- Zavestni sistem se da oponašati.

Zavest je ključno povezana s pojmi:
življenje, **inteligenca** in **svobodna volja**:

Zavest = življenje?

Več inteligence omogoča večjo zavest?

Zavest implicira svobodno voljo?

Znanost in filozofija

Znanost brez vere je hroma, vera brez znanosti je slepa.
Albert Einstein

Znanost modelira empirične podatke; če hipoteza dovolj natančno opisuje podatke, se sčasoma privzame kot (naravni) zakon.

Znanost se ne sprašuje: **zakaj** vesolje obstaja in čemu je življenje namenjeno.

Zaradi ignoriranja teh dveh vprašanj mnogi privzamejo kot temeljni princip, da je vesolje in življenje nastalo **po naključju**.

SCIENCE	SPIRITUALITY
scientists	mystics
logical, rational mind **head**	intuitive mind, inner sense **heart**
objectivity measurable, describable	**subjectivity** nonmeasurable, undescribable
describing reality	conscious sensing, awareness of reality
how?	why?
studies matter	studies consciousness
life appeared by chance	life is chance
doubt, verification	faith

SCIENCE	SPIRITUALITY
logic, experiments, statistics	relaxation, meditation, ceremonials
analysis, differentiates, parts	synthesis, joins, whole
separation, space-time dimension	all is one, spectral dimension
causality, thinking of past and future	no causality, **now!**

SCIENCE	SPIRITUALITY
discrete, rational world ($\mathcal{Q}$)	continuous, real ($\mathcal{R}$), irrational world
objective, **indirect** experience	subjective, **direct** experience
theory, approximation of reality	**practice**, reality itself
active, violent free will	passive, harmonious free will
subordination, control	cooperation
taking, profit, ego	giving, sharing

SCIENCE	SPIRITUALITY
knowledge	**wisdom**
Scientific theories: quantum, relativity, thermodynamics, evolution ...	Spiritual virtues: love, humility, compassion, patience, courage, sincerity ...
Scientific branches: mathematics, physics, chemistry, biology ...	Spiritual movements: yoga, tao, zen, sufism, institutional religions, new age ...

- Filozofija (v prvotnem pomenu besede): razum in srce.
- Inteligenca je sposobnost.
- Zavest je povezana z etiko življenja.
- Inteligenca brez srca je nezavedna inteligenca, sposobna uničevati in uničiti okolje in sebe.
- **Dalaj Lama** v Ljubljani: *"Potrebujemo izobrazbo in čut moralne etike - to dvoje mora iti skupaj."*
- Umetna (in naravna) inteligenca je orodje.
- Odgovornost pa ostaja na tvoji (za)vesti.

Zakaj strojno učenje

človek	računalnik
pozabljiv utrudljiv težko prenaša znanje	zanesljiv, ponovljiv 24 ur/dan trivialno prenosljivo znanje
omejen spomin ? počasen ?	velike baze podatkov ? hiter ?
široko znanje paralelelno procesiranje se uči iz napak dinamično znanje	ozko specializirano znanje zaporedno procesiranje isto napako ponavlja statično znanje

Zakaj strojno učenje

- Človek si želi spoznati samega sebe.
- Učenje ljudi se zdi zelo počasno.
- Avtomatsko generiranje baz znanja za ekspertne sisteme.
- Računalniki manj togi in bolj prilagodljivi novim situacijam in problemom.
- Nadzor nad učečimi se sistemi.

3 Osnovne strojnega učenja

Definicija strojnega učenja:

1. učenje kot modeliranje,
2. princip najkrajšega opisa
3. inkrementalno učenje,
4. princip večkratne razlage,
5. ocenjevanje verjetnosti

Definicija strojnega učenja

Učenje kot modeliranje

- *učni algoritem*,
- *izvajalni algoritem*.

Avtomatsko zgenerirano znanje = *model* (*hipoteza*, *teorija*)

Strojno učenje = opisovanje ali *modeliranje* podatkov.

množica podatkov + predznanje $\longrightarrow$ opis, (model, hipoteza, teorija)

Predznanje = prostor možnih modelov + kriterij optimalnosti +
začetna hipoteza + množica hevristik + ?

strojno učenje = **optimizacija:**

Dano: prostor možnih rešitev + kriterijska funkcija

Poišči: rešitev, ki minimizira kriterijsko funkcijo

Vrste modelov:

Diskretne funkcije: *klasifikacijski problemi*
mnogolične funkcije
zaloga vrednosti = verjetnostna distribucija

Zvezne funkcije: zvezni (urejeni) razred
regresijski problemi.
interval zaupanja

Relacije: splošnejše od funkcij
večja kompleksnost
ILP in sistemi enačb

Princip najkrajšega opisa

It is vain to do with more what can be done with fewer.
William of Ockham

Nature is pleased with simplicity, and affects not the pomp of superfluous causes.
Isaac Newton

hipoteza, ki "čim bolj ustreza" vhodnim podatkom in predznanju

Princip *Ockhamove britve* (Occam's Razor Principle):
Najpreprostejša razlaga je najbolj zanesljiva (verjetna).

Kriteriji:

- maksimizirati klasifikacijsko točnost hipoteze,
- minimizirati povprečno ceno klasifikacijskih napak,
- minimizirati velikost hipoteze,
- maksimizirati prileganje hipoteze vhodnim podatkom,
- maksimizirati razumljivost hipoteze,
- minimizirati časovno zahtevnost klasifikacije,
- minimizirati število parametrov, potrebnih za klasifikacijo,
- minimizirati ceno pridobivanja vrednosti parametrov,
- **maksimizirati verjetnost hipoteze**

Princip najkrajšega opisa

$\mathcal{H}$ - množica možnih hipotez
$H \in \mathcal{H}$ - hipoteza,
B - predznanje
E - vhodni podatki

Optimalna hipoteza:

$$H_{opt} = arg \max_{H \in \mathcal{H}} P(H|E, B)$$

Princip najkrajšega opisa

$P(H|B)$ - (apriorna) verjetnost hipoteze

Apriorna količina informacije hipoteze H:

$$I(H|B) = -\log_2 P(H|B) \ \ [bit]$$

Aposteriorna količina informacije hipoteze H:

$$I(H|E,B) = -\log_2 P(H|E,B) \ \ [bit]$$

Optimalna hipoteza:

$$H_{opt} = arg \ \min_{H \in \mathcal{H}} I(H|E,B)$$

Princip najkrajšega opisa

Po Bayesovem teoremu velja:

$$I(H|B,E) = I(E|H,B) + I(H|B) - I(E|B)$$

$I(E|B)$ je konstanta, neodvisna od hipoteze:

$$H_{opt} = arg \min_{H \in \mathcal{H}} (I(E|H,B) + I(H|B))$$

Komunikacijski kanal:

Oddajnik $\longrightarrow$ OPIS PODATKOV $\longrightarrow$ sprejemnik

sporočilo = niz ničel in enic
dolžina sporočila = število bitov za zakodiranje opisa
predpostavimo optimalno kodiranje
dekodiranje: opis dekoderja je lahko del sporočila

Naloga oddajnika: minimizirati dolžino sporočila

Oddajnik lahko:

- zakodira podatke in jih pošlje sprejemniku ali
- zakodira hipotezo (model) ter zakodira še izpeljave podatkov

Kompresivna hipoteza: dolžina opisa modela + izpeljave < dolžina opisa podatkov:

$$I(H|B) + I(E|H, B) < I(E|B)$$

B - predznanje imata oba, oddajnik in sprejemnik

Inkrementalno učenje

Nature never says wheter the guesses are correct. Scientific success consists of eventually offering a correct guess and never deviating from it thereafter.

(Osherson et al., 1986)

Inkrementalno učenje: spreminjanje teorije po vsakem podatku
Problem: najmanjša potrebna sprememba trenutne teorije
Zapominjanje/pozabljanje učnih primerov
Učni algoritem **nikoli** z gotovostjo ne ve, če se je naučil optimalno teorijo.

Princip večkratne razlage

If more than one theory is consistent with the observations, keep all theories.

Epicurus

For each particular model find what its prediction is and then weight its prediction with the probability of that model, and the weighted sum of those predictions is the optimal prediction.

Peter Cheeseman

Obdrži vse konsistentne (verjetne) hipoteze z vhodnimi podatki!

Princip MDL:
iskanje (v povprečju) najboljše hipoteze
(usmerjanje učnega algoritma)

Princip večkratne razlage:
kombinacija več verjetnih hipotez
(usmerjanje izvajalnega algoritma)

$\mathcal{R}$ - množica možnih rešitev
$r \in \mathcal{R}$ - ena konkretna rešitev problema
$P(r|H)$ - verjetnost rešitve r, če je hipoteza H pravilna

Rešitev, ki jo ponuja optimalna hipoteza:

$$r_{opt1} = arg \max_{r \in \mathcal{R}} P(r|H_{opt})$$

Toda to ni optimalna rešitev!

Optimalni izvajalni algoritem uporablja vse (dovolj) verjetne hipoteze:

$$r_{opt} = arg \max_{r \in \mathcal{R}} \sum_{H \in \mathcal{H}} (P(r|H)P(H|E,B))$$

Navidezno nasprotje: kombinacija več teorij = teorija

spremenjen prostor možnih teorij: $n \longrightarrow >> 2^n$

Ocenjevanje verjetnosti

oceniti verjetnost iz majhne množice vhodnih podatkov
Apriorna gostota verjetnosti: beta porazdelitev $\beta(a, b)$:

$$p(x) = \begin{cases} \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1} & 0 \leq x \leq 1 \\ 0 & sicer \end{cases}$$

$$B(a, b) = \int_0^1 x^{a-1}(1-x)^{b-1} dx$$

$a > 0$ in $b > 0$ interpretiramo: a uspešnih in b neuspešnih
matematično upanje slučajne spremenljivke p porazdeljene po $\beta(a, b)$:

$$Exp(p) = \frac{a}{a+b}$$

Ocenjevanje verjetnosti

r uspešnih in n vseh primerov, potem ocenimo verjetnost uspeha:

$$p = \frac{r + a}{n + a + b} \qquad \longleftarrow \beta(a + r, b + n - r)$$

- začetna porazdelitev $\beta(0, 0)$: *relativna frekvenca*

$$p = \frac{r}{n}$$

- začetna porazdelitev $\beta(1, 1)$: *Laplaceov zakon zaporednosti* k = število možnih izidov:

$$0 < p = \frac{r + 1}{n + k} < 1$$

Ocenjevanje verjetnosti

- $m = a + b$, $p_0 = \frac{a}{a+b}$: *m-ocena verjetnosti*

$$p = \frac{r + mp_0}{n + m} = \frac{n}{n + m} \times \frac{r}{n} + \frac{m}{n + m} \times p_0$$

$m = k$, $p_0 = 1/k \longrightarrow$ Laplace

Ocenjevanje učenja:

1. klasifikacijska točnost,
2. tabela napačnih klasifikacij,
3. cena napačne klasifikacije,
4. Brierjeva mera,
5. informacijska vsebina,
6. senzitivnost in specifičnost, krivulja ROC,
7. priklic in preciznost,
8. srednja kvadratna in absolutna napaka,
9. korelacijski koeficient.

Ocenjevanje učenja

primeri rešenih problemov: *učni* in *testni* primeri

Klasifikacijska točnost (classification accuracy)

M_R - število možnih razredov

N - število vseh možnih primerov problemov na danem področju

$N^{(p)}$ - število pravilnih rešitev primerov

N_t - število vseh testnih primerov

$$T = \frac{N^{(p)}}{N} \times 100\% \sim T_t = \frac{N_t^{(p)}}{N_t} \times 100\%$$

Klasifikacijska točnost na N_u učnih primerih = zgornja meja:

$$T_u = \frac{N_u^{(p)}}{N_u} \times 100\%$$

Če $T_u >> T_t$:*preveč prilagojeno* učni množici (overfitting)

večinski razred: spodnja meja klasifikacijske točnosti
$N_u^{(i)}$ - število učnih primerov iz i-tega razreda

$$T_v = \max_i \frac{N_u^{(i)}}{N_u}$$

$T_t < T_v$: klasifikator je neuporaben

Problem prognostike ponovitve raka na dojki: $T_v = 80\%$
Večina parametrov je nepomembnih za prognozo.
Apriorni verjetnosti razredov: $P_1 = 0.8$ in $P_2 = 0.2$
Klasifikacijska točnost: $P_1^2 + P_2^2 = 68\%$

Tabela napačnih klasifikacij

	klasificiran kot			
pravi razred	C1	C2	C3	vsota
C1	12.3	2.4	8.5	23.2
C2	5.5	58.7	2.1	66.3
C3	0.0	2.0	8.5	10.5
vsota	17.8	63.1	19.1	100.0

Cena napačne klasifikacije

$N_t^{(ij)}$ - število testnih primerov iz i-tega razreda, ki jih dana teorija klasificira v j-ti razred.

$$N_t^{(p)} = \sum_{i=1}^{M_R} N_t^{(ii)}$$

C_{ij} - cena napačne klasifikacije

Lahko:$C_{ij} \neq C_{ji}$ Ponavadi: $C_{ii} = 0$, $i = 1, ..., M_R$
Povprečna cena napačne klasifikacije:

$$C_t = \frac{\Sigma_{i,j}(C_{ij} N_t^{(ij)})}{N_t}$$

Brierjeva mera

M_R število razredov
$r^{(j)}$ razred j-tega testnega primera
$P'_j(r_i), i = 1..M_R$ napovedana verjetnostna distribucija
$C_j(r^{(j)}) = 1$ in $C_j(r_i) = 0, r_i \neq r^{(j)}$ Ciljna distribucija
N_t število testnih primerov:

$$Brier = MSE_P = \frac{\Sigma_{j=1}^{N_t} \Sigma_{i=1}^{M_R} (C_j(r_i) - P'_j(r_i))^2}{N_t}$$

kvaliteta klasifikatorja: $1 - Brier/2$.

Informacijska vsebina odgovora

Odgovor: verjetnostna distribucija po vseh razredih
Apriorne verjetnosti in aposteriorne verjetnosti razredov

domena	T_u	T_v	M_r	$H(R)$	Inf
rak na dojki	80%	80%	2	0.72 bit	0.0 bit
primarni tumor	45%	25%	22	3.64 bit	1.6 bit

Apriorna verjetnost i-tega razreda: $P(r_i) = \frac{N_u^{(i)}}{N_u}$

Informacija, da zvemo, da primer pripada i-temu razredu:

$$H(r_i) = -\log_2 P(r_i) \quad [bit]$$

Entropija razredov: $H(R) = -\sum_{i=1}^{R} (P(r_i) \log_2 P(r_i))$

$H(R)$ doseže maksimum, ko so vsi razredi enako verjetni:

$$P(r_i) = \frac{1}{M_R}, \qquad i = 1, ..., M_R$$

ter minimum, ko so vsi primeri iz istega razreda r_j:

$$P(r_j) = 1, \qquad in \qquad P(r_i) = 0, \quad i \neq j$$

Povprečna informacijska vsebina odgovora (information score):
$r^{(j)}$ - pravilni razred danega j-tega testnega primera
$P'(r^{(j)})$ - aposteriorna verjetnost tega razreda

$$Inf = \frac{\sum_{j=1}^{N_t} Inf_j}{N_t} \qquad [bit]$$

in

$$Inf_j = \begin{cases} -\log_2 P(r^{(j)}) + \log_2 P'(r^{(j)}), & P'(r^{(j)}) \geq P(r^{(j)}) \\ -(-\log_2(1 - P(r^{(j)})) + \log_2(1 - P'(r^{(j)}))), & P'(r^{(j)}) < P(r^{(j)}) \end{cases}$$

Imejmo dva možna razreda: $P(r_1) = 0.8$ in $P(r_2) = 0.2$
Klasifikator vrne $P'(r_1) = 0.6$ in $P'(r_2) = 0.4$

- Če je pravilni razred r_1:
  klasifikacijska točnost: odgovor je pravilen
  informacijska vsebina: odgovor je nepravilen (zavajajoč)

- Če je pravilni razred r_2:
  klasifikacijska točnost: odgovor je nepravilen
  informacijska vsebina: odgovor je pravilen (koristen)

Meje: $0 \leq Inf \leq H(R)$

Relativna informacijska vsebina odgovora:

$$RInf = \frac{Inf}{H(R)} \times 100\%$$

Senzitivnost in specifičnost

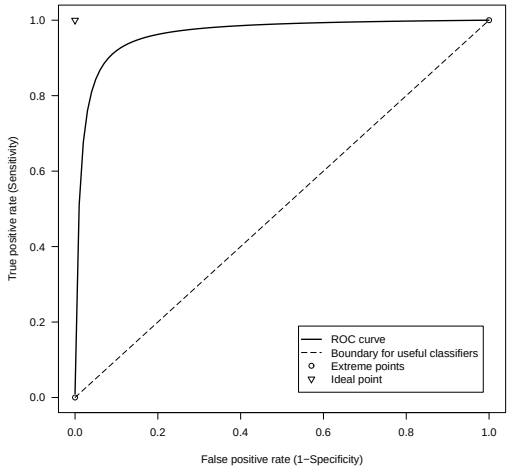
	klasificiran kot		
pravi razred	P	N	vsota
P	TP	FN	POS=TP+FN
N	FP	TN	NEG=FP+TN
vsota	PP=TP+FP	PN=FN+TN	n = TP+FP+FN+TN

$$Senzitivnost = \frac{TP}{TP + FN} = \frac{TP}{POS}$$

$$Specificnost = \frac{TN}{TN + FP} = \frac{TN}{NEG}$$

$$Tocnost = \frac{TP + TN}{TN + FP + FN + TN} = \frac{TP + TN}{n}$$

Krivulja ROC (Reciever Operating Characteristic curve)



ploščina pod krivuljo ROC (Area Under the ROC Curve, AUC)

Srednja kvadratna napaka (mean squared error)

Pri regresijskih problemih:
napovedana vrednost: $\hat{f}(i)$ želena vrednost: $f(i)$

$$E = \frac{1}{N}\sum_{i=1}^{N}(f(i) - \hat{f}(i))^2$$

Relativna srednja kvadratna napaka:

$$0 \leq RE = \frac{N \times E}{\Sigma_i(f(i) - \overline{f})^2} \leq 1 \quad kjer\ je\ \ \overline{f} = \frac{1}{N}\sum_i f(i)$$

$\hat{f}(i) = \overline{f} \longrightarrow RE = 1$ trivialna f. ($RE > 1 \rightarrow$ neuporabna)
$\hat{f}(i) = f(i) \longrightarrow RE = 0$ idealna funkcija

Srednja absolutna napaka

mean absolute error (MAE)

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |f(i) - \hat{f}(i)|$$

relativna srednja absolutna napaka:

$$RMAE = \frac{N \times MAE}{\Sigma_i |f(i) - \overline{f}|}$$

$$0 \leq RMAE \leq 1$$

Korelacijski koeficient

$$K = \frac{S_{f\hat{f}}}{S_f S_{\hat{f}}}$$

kjer so:

$$S_{f\hat{f}} = \frac{\Sigma_{i=1}^{N}[(f(i) - \overline{f})(\hat{f}(i) - \overline{\hat{f}})]}{N-1}$$

$$S_f = \frac{\Sigma_{i=1}^{N}(f(i) - \overline{f})^2}{N-1}$$

$$S_{\hat{f}} = \frac{\Sigma_{i=1}^{N}(\hat{f}(i) - \overline{\hat{f}})^2}{N-1}$$

smiselne samo pozitivne vrednosti:$0 < K \leq 1$

Ocenjevanje učenja:

1. Prečno preverjanje,
2. Metoda razmnoževanja učnih primerov,
3. Primerjanje uspešnosti različnih algoritmov,
4. Kombiniranje algoritmov strojnega učenja

Prečno preverjanje

- metoda *izvrži enega* (leave-one-out)
  N primerov: moramo zgraditi $N + 1$ hipotez
- "izvrži N/K primerov": *prečno preverjanje reda* K
  (K-fold cross-validation)
  K določa število hipotez; v praksi $K = 10$

- standardna napaka ($1 - \alpha = 68\%$)
- interval zaupanja (npr. $1 - \alpha = 95\%$)

Primerjanje uspešnosti različnih učnih algoritmov

Učenje: *učna množica* (learning/training set)
Nastavljanje parametrov: *nastavitvena množica* (validation set)
Testiranje: *testna množica* (testing set)
Pogosta napaka: testna množica se uporabi kot nastavitvena...

Parametrični testi značilnosti odstopanj:
ocenimo stopnjo zaupanja v ocenjene razlike uspešnosti

Primerjanje uspešnosti različnih učnih algoritmov

- Dva algoritma na eni domeni
  - Prečno preverjanje: enosmerni t-test
  - Izloči enega ali neodvisna testna množica: enosmerni z-test
  - Bonferronijeva korekcija (N primerjav): $\alpha_B \approx \alpha/N$
- Two algorithms on several domains: nonparametric Wilcoxon signed rank test
- Več algoritmov na več domenah: Friedmanov test
  - En algoritem proti ostalim: test Bonferroni-Dunn
  - Vsak z vsakim: Nemenyijev test

Kombiniranje algoritmov strojnega učenja

- različne učne algoritme
- isti učni algoritem poganjamo večkrat z različnimi nastavitvami parametrov ali nad različno pripravljenimi vhodnimi podatki

Kombiniranje napovedi različnih hipotez:

- Glasovanje (angl. voting)
- Uteženo glasovanje
- Glasovanje, uteženo z zanesljivostjo napovedi
- Kombiniranje po metodi naivnega Bayesa
- Naučeno kombiniranje z meta-učenjem (stacking)
- Lokalno uteženo glasovanje

Povezovanje algoritmov

Vzporedno

Zaporedno: prvi algoritem se uči ciljne napovedi. Vsak naslednji algoritem pa se uči napovedovanja napak prejšnjega algoritma.

Kombinirano... (problem prevelikega prileganja)

Bagging

- "bootstrap agreggating"(Breiman, 1996): zaporedje hipotez iz različnih učnih množic
- n krat naključno izberemo primer z vračanjem
- nekaterih primerov množica sploh ne vsebuje (36.8%)
- Vse hipoteze glasujejo za napoved.
- Dober pri nestabilnih učnih algoritmih z visoko varianco.
- Z večanjem števila modelov ne pride do prevelikega prileganja.

Naključni gozdovi

(angl. random forests) Izboljša točnost drevesnih algoritmov.

- Pri izbiri najboljšega atributa v vsakem vozlišču naključno izbere relativno majhno število atributov, ki vstopajo v izbor za najboljši atribut.
- Število tako zgrajenih dreves je 100 ali več.
- Klasifikacija: glasovanje vseh dreves.
- Robustna, saj zmanjša varianco drevesnih algoritmov.
- Slabost: razlaga odločitev je otežena.

Boosting

- uteževanje učnih primerov glede na njihovo težavnost
- predpostavka: zna obravnavati utežene učne primere;
- sicer učno množico generiramo kot pri baggingu, le da verjetnostna distribucija ni uniformna.
- V prvi iteraciji učni algoritem zgradi hipotezo na normalnih učnih primerih. Hipoteza napove odvisno spremenljivko za vse učne primere. Primerom s pravilno napovedjo zmanjšamo uteži in primerom z napačno napovedjo povečamo uteži.
- $e =$ napaka hipoteze na učnem primeru, utež primera se pomnoži z $e/(1-e)$. Na koncu normalizacija.

Boosting

- Pri napovedovanju se uporabijo vse hipoteze, vendar napoved vsake hipoteze utežimo z njeno točnostjo na uteženi učni množici, iz katere je bila zgenerirana.
- utež $-log(f/(1-f))$, kjer je f napaka hipoteze
- Boosting pogosto dosega boljšo točnost od bagginga in ga lahko uporabimo tudi na stabilnih učnih algoritmih z majhno varianco.
- Vendar lahko včasih pride do prevelikega prileganja učni množici (overfitting).

Nevronske mreže: Umetni nevron je abstrakcija biološkega.

$$x_{out} = f\Big(\sum_i w_i x_i + w_{bias}\Big)$$

Pragovna funkcija f je lahko definirana z:

$$f(X) = \begin{cases} 1 & X > 0 \\ 0 & X \leq 0 \end{cases}$$

Bolj pogosto se uporablja zvezna in zvezno odvedljiva funkcija:

$$f(X) = \frac{1}{1 + e^{-X}}$$

Topologija: usmerjene večnivojske umetne nevronske mreže (angl. feedforward multilayered artificial neural networks).

Povezavam med nevroni pravimo tudi *sinapse*.
Matrika vseh uteži definira funkcijo nevronske mreže.

- klasifikacijski problem: n_0 izhodnih nevronov,
- regresijski problem: en sam izhodni nevron

Dvonivojska nevronska mreža: linearni problemi.

Najpogosteje: tri (in štiri) nivojske nevronske mreže.
S številom skritih nivojev in s številom nevronov v vsakem skritem nivoju je določena zmožnost opisovanja funkcij.

Naivni Bayesov klasifikator

Predpostavlja pogojno neodvisnost atributov pri danem razredu.

$$P(r_k|V) = P(r_k)\frac{P(V|r_k)}{P(V)}$$

Predpostavka pogojne neodvisnosti:

$$P(V|r_k) = P(v_1 \,\&\, ... \,\&\, v_a|r_k) = \prod_{i=1}^{a} P(v_i|r_k)$$

$$P(r_k|V) = \frac{P(r_k)}{P(V)} \prod_{i=1}^{a} P(v_i|r_k)$$

Še enkrat Bayesovo pravilo:

$$P(v_i|r_k) = P(v_i)\frac{P(r_k|v_i)}{P(r_k)}$$

$$P(r_k|V) = P(r_k)\frac{\Pi_{i=1}^{a} P(v_i)}{P(V)} \prod_{i=1}^{a} \frac{P(r_k|v_i)}{P(r_k)}$$

Faktor:

$$\frac{\Pi_{i=1}^{a} P(v_i)}{P(V)}$$

je neodvisen od razreda, zato ga lahko izpustimo:

$$P(r_k|V) = P(r_k) \prod_{i=1}^{a} \frac{P(r_k|v_i)}{P(r_k)}$$

5 Iskalne metode

H_z – začetna hipoteza
$\longrightarrow$ – en operator iz množice operatorjev
$\{H' \mid H_z \longrightarrow H'\}$ – množica naslednikov
$\mathcal{H} = \{H \mid H_z \longrightarrow^* H\}$ – prostor hipotez
E – učni primeri
B – predznanje
$q(E, B, H)$ – ocena kvalitete hipoteze
$\forall H',\ H \longrightarrow^* H' :\ max(E, B, H) \geq \hat{q}(E, B, H')$ – kval. nasl.

Naloga:

$$H_0 = arg \max_{H \in \mathcal{H}} q(E, B, H)$$

Osnovne strategije preiskovanja:

- izčrpno preiskovanje (globina, širina, iterativno poglabljanje),
- omejeno izčrpno iskanje - razveji in omeji (branch and bound),
- metoda “najprej najboljši”,
- požrešno iskanje,
- iskanje v snopu,
- lokalna optimizacija,
- gradientno iskanje,
- simulirano ohlajanje,
- genetski algoritmi.

```
function V_širino(E : Ins; B : Any; H_z : hypo) : hypo;
var
  H_0 : hypo; (* trenutno najboljša hipoteza *)
  H : hypo; (* trenutno obravnavana hipoteza *)
  H' : hypo; (* naslednik trenutno obravnavane hipoteze *)
  H_m : hypo; (* najboljši naslednik *)
  Open : list of hypo; (* hipoteze z nepregl. nasl. *)
  Successors : list of hypo; (* seznam naslednikov *)
```

begin
$H_0 := H_z$;
Open := $[H_z]$;
while Open $\neq$ [] **and not** timeout **do**
H := head(Open);
Open := tail(Open); (* zbrišemo prvi element *)
Successors := {H' | H $\longrightarrow$ H'};
Open := conc(Open,Successors); (* stik seznamov *)
$H_m := arg\max_{H' \in Successors} q(E, B, H')$;
if q(E,B,H_m) > q(E,B,H_0) **then** $H_0 := H_m$; **end if**
end while
return(H_0);
end

```
function It_poglabljanje(E : Ins; B : Any; H_z : hypo) : hypo;
var H_0, H_m: hypo; (* trenutno najboljša hipoteza *)
 Globina : integer; (* trenutna globina iteracije *)
begin
 H_0 := H_z;
 Globina := 1;
 while not timeout do
   H_m := V_globino(E,B,H_z,Globina);
   if q(E,B,H_m) > q(E,B,H_0) then H_0 := H_m; end if
   Globina := Globina + 1;
 end while
 return(H_0);
end
```

function V_globino(E : Ins; B : Any; H_z : hypo;
Globina : integer) : hypo;
var
(* H_z : hypo; - trenutno obravnavana hipoteza *)
H_0 : hypo; (* trenutno najboljša hipoteza *)
H' : hypo; (* naslednik trenutno obravnavane hipoteze *)
H_m : hypo; (* najboljši naslednik *)
Successors : list of hypo; (* seznam naslednikov *)

```
begin
 if Globina = 0 then return(H_z) else
   H_0 := H_z;
   Successors := {H' | H_z ⟶ H'};
   for H' ∈ Successors do
     H_m := V_globino(E,B,H',Globina-1);
     if q(E,B,H_m) > q(E,B,H_0) then H_0 := H_m; end if
   end for
   return(H_0);
 end if
end
```

```
function Omejeno_v_širino(E: Ins; B: Any; H_z: hypo): hypo;
var
  H_0 : hypo; (* trenutno najboljša hipoteza *)
  H : hypo; (* trenutno obravnavana hipoteza *)
  H' : hypo; (* naslednik *)
  H_m : hypo; (* najboljši naslednik *)
  Open : list of hypo; (* seznam hipotez z nepregl. nasl. *)
  Successors : list of hypo; (* seznam naslednikov *)
```

begin
$H_0 := H_z$; Open := $[H_z]$;
while Open $\neq$ [] **and not** timeout **do**
H := head(Open); Open := tail(Open);
if max(E,B,H) > q(E,B,H_0) **then**
Successors := {H' | H $\longrightarrow$ H'}
Open := conc(Open,Successors);
$H_m := arg\max_{H' \in Successors} q(E, B, H')$;
if q(E,B,H_m) > q(E,B,H_0) **then** $H_0 := H_m$; **end if**
end if (* else prune *)
end while
return(H_0);
end

```
procedure Omej_v_glob(E:Ins;B:Any;H_z:hypo; var H_0:hypo);
var
  (* H_z : hypo; - trenutno obravnavana hipoteza *)
  (* H_0 : hypo; - trenutno najboljša hipoteza *)
 H' : hypo; (* naslednik *)
 Successors : list of hypo; (* seznam naslednikov *)
begin
 if q(E,B,H_z) > q(E,B,H_0) then H_0 := H_z; end if
 if max(E,B,H_z) > q(E,B,H_0)then
   Successors := {H' | H_z ⟶ H'};
   for H' ∈ Successors do Omejeno_izčrpno_v_globino(E,B,H',H_0);
   end for
 end if (* else prune *)
end
```

Primer:

$$q(E,B,H) = \begin{cases} splošnost(H) & \forall e \in E:\ B\ \&\ H \models e \\ -\infty & \exists e \in E:\ B\ \&\ H \not\models e \end{cases}$$

$$\forall e \in E:\ B\,\&\,H \models e \Rightarrow \forall H', H \longrightarrow^* H':\ q(E,B,H) > q(E,B,H')$$

$$H_{max} = arg \max_{H} max(E, B, H)$$

function Najprej_najboljši(E : Ins; B : Any; H_z : hypo) : hypo;
var
H_0 : hypo; (* trenutno najboljša hipoteza *)
H_{max} : hypo; (* trenutno obravnavana hipoteza *)
H' : hypo; (* naslednik *)
H_m : hypo; (* najboljši naslednik *)
Open : list of hypo; (* seznam hipotez z nepregl. nasl. *)
Successors : list of hypo; (* seznam naslednikov *)

begin
$H_0 := H_z$; Open := $[H_z]$;
while Open $\neq$ [] **and not** timeout **do**
$H_{max} := arg\max_{H \in Open} max(E, B, H)$;
Open := Open$\backslash\{H_{max}\}$;
if max(E,B,H_{max}) $\leq$ q(E,B,H_0) **then** Open := []; **else**
Successors := {H' | $H_{max} \longrightarrow$ H'}
Open := conc(Open,Successors);
$H_m := arg\max_{H' \in Successors} q(E, B, H')$;
if q(E,B,H_m) > q(E,B,H_0) **then** $H_0 := H_m$; **end if**
end if
end while
return(H_0);
end

Požrešno iskanje (samo najboljši)

V prostoru hipotez:

$$max_1(E, B, H) = \max\left(max(E, B, H), q(E, B, H)\right)$$

$$H_{max} = arg \max_{H',\ H \longrightarrow H'} max_1(E, B, H')$$

V prostoru delnih hipotez:
$\mathcal{D} \supset \mathcal{H}$

$\forall H', D \longrightarrow^* H', H' \in \mathcal{H}, D \in \mathcal{D} :\ max_2(E, B, D) \geq \hat{q}(E, B, H')$

function Požrešni(E : Ins; B : Any; H_z : hypo) : hypo;
var
H_0 : hypo; (* trenutno obravnavana hipoteza *)
H_{max}, H' : hypo; (* najboljši naslednik in naslednik*)
Successors : list of hypo; (* seznam naslednikov *)
begin
$H_0 := H_z$;
repeat
Successors := {H' | $H_0 \longrightarrow$ H'};
$H_{max} := arg\max_{H' \in Successors} max_1(E, B, H')$;
if $max_1(E, B, H_{max}) > q(E, B, H_0)$ **then** $H_0 := H_{max}$;**endif**
until $H_0 \neq H_{max}$;
return(H_0);
end

```
function Iskanje_v_snopu(E:Ins;B:Any;H_z:hypo;M:integer): hypo;
var
  H, H': hypo; (* trenutno obravnavana hipoteza in naslednik *)
  Beam, Beam1 : set of hypo; (* tren. in nova množica hipotez *)
  Successors : set of hypo; (* množica naslednikov *)
begin
  Beam1 := [H_z];
  repeat
    Beam := Beam1;
    Successors := {H' | H ⟶ H', H ∈ Beam};
    Beam1 := M najbolje ocenjenih iz množice Beam ∪ Successors;
  until Beam = Beam1;
  return(arg max_{H∈Beam} q(E,B,H));
end
```

Lokalna optimizacija:

- Začetna hipoteza $H_z \in \mathcal{H}$ je naključno generirana.
- Išče samo v prostoru hipotez $\mathcal{H}$.
- Večje število možnih operatorjev.
- Poženemo večkrat in obdržimo najboljšo hipotezo.

Gradientno iskanje:

$$q'(E, B, H) = dq(E, B, H(p))/dH(p)$$

$$p = p + \eta \frac{dq(E, B, H(p))}{dp}$$

function Gradientno(E : Ins; B : Any; η : real) : parameter;
var
p_0 : parameter; (* trenutna vrednost parametra *)
p_{new} : parameter; (* nova vrednost parametra *)
begin
p_{new} := random_parameter;
repeat
$p_0 := p_{new}$;
$p_{new} := p_{new} + \eta q'(E, B, H(p_{new}))$;
until $q(E, B, H(p_0) > q(E, B, H(p_{new})) + \varepsilon$;
return(p_0);
end

Simulirano ohlajanje

Termodinamika: verjetnost, da se atom nahaja v stanju z energijo E_i:

$$P(E_i) \propto e^{\frac{-E_i}{T}}$$

Čim počasneje ohlajamo snov, tem večja je verjetnost, da bo stanje optimalnejše.

- Naključna začetna hipoteza.
- Stohastična izbira naslednika.
- Temperatura se po vsaki iteraciji zmanjša.

Lastnosti simuliranega ohlajanja:

- Verjetnost izbire $\propto$ kvaliteta naslednika
- Čim večja je temperatura, tem bolj naključno iskanje
- $T = 0$ – lokalna optimizacija.
- $q(E, B, H') > q(E, B, H)$, potem $P(H_0 := H') = 1$, sicer

$$P(H_0 := H') = e^{\frac{q(E,B,H')-q(E,B,H)}{T}}$$

- temperatura pada po geometrijskem zaporedju:

$$T' := \lambda T, \qquad \lambda < 1$$

- Na koncu deterministična lokalna optimizacija

function Sim_ohlajanje(E: Ins; B: Any; T_{max}, λ, T_{min}: real): hypo;

var

T : real; (* trenutna temperatura *)

H_0 : hypo; (* trenutno obravnavana hipoteza *)

H' : hypo; (* naslednik *)

H_{max} : hypo; (* najboljši naslednik *)

Successors : list of hypo; (* seznam naslednikov *)

begin
H_0 := random_hypo; T := T_{max};
repeat
H' := random_successor(H_0);
if $random < e^{\frac{q(E,B,H')-q(E,B,H)}{T}}$ **then** $H_0 := H'$; **end if**
$T := \lambda T$;
until $T \leq T_{min}$;
repeat (* lokalna optimizacija *)
Successors := {H' | $H_0 \longrightarrow$ H'};
$H_{max} := arg\max_{H' \in Successors} q(E, B, H')$;
if $q(E, B, H_{max}) > q(E, B, H_0)$ **then** $H_0 := H_{max}$;**endif**
until $H_0 \neq H_{max}$;
return(H_0);
end

Metoda markovskih nevronskih mrež

- obstajajo boljši nasledniki:

$$P(H_0 := H') = \frac{e^{\frac{q(E,B,H')-q(E,B,H_0)}{T}}}{\sum_{H'',H_0 \to H'',q(E,B,H'')>q(E,B,H_0)} e^{\frac{q(E,B,H'')-q(E,B,H_0)}{T}}}$$

- vsi nasledniki slabši:

$$P(H_0 := H') = \frac{e^{\frac{q(E,B,H')-q(E,B,H_0)}{T}}}{\sum_{H'',H_0 \to H'' \vee H''=H_0} e^{\frac{q(E,B,H'')-q(E,B,H_0)}{T}}}$$

```
function MNN(E : Ins; B : Any; T_max, λ, T_min : real ) : hypo;
var
  R, Q, Sum : real; (* metanje kocke na intervalu [0..1] *)
  T : real; (* trenutna temperatura *)
  H_0 : hypo; (* trenutno obravnavana hipoteza *)
  H' : hypo; (* naslednik *)
  H_max : hypo; (* najboljši naslednik *)
  Successors : list of hypo; (* seznam naslednikov *)
  i : integer; (* kazalec na Successors *)
```

begin
H_0 := random_hypo; T := T_{max};
repeat
Successors := $\{H' \mid H_0 \longrightarrow H' \wedge$ q(E,B,H') > q(E,B,H_0)$\}$;
if Successors = {} **then** Successors := $\{H' \mid H_0 \longrightarrow H' \vee$ H' = $H_0\}$; **end if**
R := random; (* metanje kocke *) Q := 0; i := 0;
Sum := $\Sigma_{H' \in Successors}\, e^{\frac{q(E,B,H')-q(E,B,H_0)}{T}}$; (* normalizacija *)
while Q < R **do** (* iskanje izbranega *)
i := i + 1;
$Q := Q + \frac{e^{\frac{q(E,B,Successors[i])-q(E,B,H_0)}{T}}}{Sum}$;
end while;
$T := \lambda T$; H_0 := Successors[i]; (* izbrani naslednik *)
until $T \leq T_{min}$;

(* lokalna optimizacija *)
repeat
Successors := {H' | $H_0 \longrightarrow$ H'};
$H_{max} := arg\max_{H' \in Successors} q(E, B, H')$;
if $q(E, B, H_{max}) > q(E, B, H_0)$ **then** $H_0 := H_{max}$; **end if**
until $H_0 \neq H_{max}$;
return(H_0);
end

Genetski algoritmi

- ideje evolucije in naravne selekcije
- hipoteza ustreza enemu *osebku*
- en osebek je zakodiran kot niz simbolov – *genov*
- naključno izbrana množico osebkov – populacija

- stohastično generiranje naslednje generacije:
  - **reprodukcija:** čim boljši je osebek $H \in \mathcal{P}$, tem večja je verjetnost, da bo prispeval naslednike v naslednji populaciji. Pričakovano število naslednikov:

$$n \frac{q(E, B, H)}{\Sigma_{H' \in \mathcal{P}} q(E, B, H)}$$

  - **križanje** (angl. crossover): osebek nastane iz dveh osebkov, tako da od vsakega podeduje del genetskega zapisa, ki je naključno izbran, Križanje se izvede $n \times P_c$ krat.
  - **mutacija:** vrednost enega gena se naključno (z majhno verjetnostjo P_m) spremeni.

- Ključno je kodiranje osebkov.
- Parametri, ki jih mora izbrati uporabnik, so:
  - G - število generacij,
  - n - število osebkov v populaciji,
  - P_c - verjetnost križanja in
  - P_m - verjetnost mutacije.

```
function Gen(E: Ins; B: Any; G, n: integer; P_c, P_m: real): hypo;
var
  H : array[1..n] of hypo; (* trenutna populacija *)
  H' : array[1..n] of hypo; (* naslednja populacija *)
  R, Q, Sum : real; (* metanje kocke na intervalu [0..1] *)
  i, j, k, l, i_1, i_2, cross: integer;
begin
(* začetna populacija *)
  for i := 1 to n do H_i := random_hypo; end for
  for j := 1 to G do (* G generacij *)
(* reprodukcija; križanje; mutacija *)
  end for
  return(arg max_{H_i} q(E, B, H_i));
end
```

(* reprodukcija *)
for i := 1 **to** n **do**
R := random(1); (* metanje kocke *)
Q := 0;
k := 0;
Sum := $\Sigma_l\, q(E, B, H_l)$; (* normalizacija *)
while Q < R **do** (* iskanje prednika *)
k := k + 1;
Q := Q + q(E,B,H_k)/Sum;
end while;
$H'_i := H_k$; (* izbrani prednik *)
end for;
H := H'; (* nova populacija *)

(* križanje *)

for k := 1 **to** round(n*P_c) **do**
  i_1 := random(n); (* prvi prednik *)
  i_2 := random(n); (* drugi prednik *)
  cross := random(hypo_length); (* točka križanja *)
  H'_{i_1} := crossover(H_{i_1}, H_{i_2}, cross); (* prvi naslednik *)
  H'_{i_2} := crossover(H_{i_2}, H_{i_1}, cross); (* drugi naslednik *)
end for;

(* mutacija *)

for i := 1 **to** n **do**
for k := 1 **to** hypo_length **do**
if random(1) < P_m **then** $H_i'[k]$:= random_value; **end if**
end for
end for

H := H'; (* nova populacija *)

6 Mere za ocenjevanje atributov

1. Mere nečistoče
2. Mere, ki temeljijo na količini informacije:
   (a) informacijski prispevek, razmerje, razdalja
   (b) povprečna absolutna teža evidence,
   (c) MDL,
   (d) J-ocena,
3. Ostale mere pri klasifikaciji:
   (a) χ^2, G,ORT,
   (b) Gini index, ReliefF.

Atributna predstavitev je podana z:

- množico atributov A = $\{A_i, i = 0..a\}$;
- za vsak diskretni atribut A_i imamo množico možnih vrednosti

$$\mathcal{V}_i = \{V_1, ...V_{n_i}\}$$

- za vsak zvezi atribut A_i imamo interval možnih vrednosti

$$\mathcal{V}_i = [Min_i, ...Max_i]$$

- razred je podan z atributom A_0: če rešujemo klasifikacijski problem, potem je A_0 diskretni atribut, če pa rešujemo regresijski problem, je A_0 zvezni atribut;
- en učni primer je vektor vrednosti atributov

$$u_j =< r^{(j)}, v^{(1,j)}, ..., v^{(a,j)} >$$

  pri tem je razred označen z $r^{(j)} = v^{(0,j)}$;
- množica učnih primerov je podana kot množica vektorjev

$$\mathcal{U} = \{u_j, j = 1...n\}$$

Vpeljimo notacijo:

n – število učnih primerov

$n_{k.}$ – število učnih primerov iz razreda r_k

$n_{.j}$ – število učnih primerov z j–to vrednostjo atributa A_i

n_{kj} – število učnih primerov iz razreda r_k in z j–to vrednostjo

$p_{kj} = n_{kj}/n_{..}$

$p_{k.} = n_{k.}/n_{..}$

$p_{.j} = n_{.j}/n_{..}$

$p_{k|j} = p_{kj}/p_{.j} = n_{kj}/n_{.j}$

Mere, ki temeljijo na količini informacije

Količina informacije:

$$I(X_i) = -\log_2 P(X_i)$$

Povprečna pričakovna količina informacije–*entropija*:

$$H(X) = -\sum_i P(X_i)\log_2 P(X_i)$$

Entropija je mera nečistoče.
Dokaz:
1. Maksimum:

$$H(X) = -\sum_i^{n_0-1} P(X_i)\log_2 P(X_i) - \left(1 - \sum_i^{n_0-1} P(X_i)\right)\log\left(1 - \sum_i^{n_0-1} P(X_i)\right)$$

$$\frac{\partial H}{\partial P(X_k)} = -\log P(X_k) - \log e + \log\left(1 - \sum_i^{n_0-1} P(X_i)\right) + \log e$$

Prvi odvod je 0 v točki: $P(X_k) = 1/n_0$
Drugi odvod je v tej točki negativen (povsod je negativen):

$$\frac{\partial^2 H}{\partial P(X_k)^2} = -\log e\left(\frac{1}{P(X_k)} + \frac{1}{\log\left(1 - \Sigma_i^{n_0-1} P(X_i)\right)}\right) < 0$$

2. Minimum:

$$H(X) \geq 0 \wedge H(X) = 0 \Leftrightarrow (P(X_i) = 0 \vee P(X_i) = 1)$$

3. H je simetrična.
4. H je konkavna.
5. H je zvezna in zvezno odvedljiva funkcija.

Vpeljimo naslednje entropije:
H_R – entropija razredov: $H_R = -\Sigma_k\, p_{k.} \log p_{k.}$
H_A – entropija vrednosti danega atributa: $H_A = -\Sigma_j\, p_{.j} \log p_{.j}$
H_{RA} – entropija produkta dogodkov razred–vrednost atributa:

$$H_{RA} = -\sum_k \sum_j p_{kj} \log p_{kj}$$

$H_{R|A}$ – pogojna (pričakovana) entropija razreda pri danem A:

$$H_{R|A} = H_{RA} - H_A = -\sum_j p_{.j} \sum_k \frac{p_{kj}}{p_{.j}} \log \frac{p_{kj}}{p_{.j}}$$

$$H_{R|A} = -\sum_j p_{.j} \sum_k p_{k|j} \log p_{k|j}$$

Ker velja $H_X \geq 0$, velja $H_{RA} \geq H_{R|A}$.

Informacijski prispevek atributa

$$Gain(A) = H_R + H_A - H_{RA} = H_R - H_{R|A}$$

Lastnosti pomembnosti atributa:

$$Gain(A) \geq 0$$

$$max(Gain(A)) = H_R$$

Če atributu A neko vrednost razbijemo na dve vrednosti, in tako dobimo novi atribut A' velja:

$$Gain(A') \geq Gain(A)$$

Razmerje informacijskega prispevka

$$GainR(A) = \frac{Gain(A)}{H_A}$$

$$Gain(A_i) \geq \frac{\sum_{j=1}^{a} Gain(A_j)}{a}$$

Mera razdalje dogodkov

$$1 - D(R, A) = \frac{Gain(A)}{H_{RA}}$$

$D(R, A)$ je razdalja:

1. $D(R, A) \geq 0$
2. $D(R, A) = 0 \Leftrightarrow R = A$
3. $D(R, A) = D(A, R)$
4. $D(R, A_1) + D(A_1, A_2) \geq D(R, A_2)$

J–ocena

Pomembnost ene vrednosti atributa bi lahko bila:

$$q_1(V_j) = \phi(P(r_1), ..., P(r_{n_0})) - \phi(P(r_1|V_j), ..., P(r_{n_0}|V_j))$$

Vendar ima tako definirana mera več slabosti:

- lahko je negativna in
- ne razlikuje med enakimi in permutiranimi distribucijami, npr.:

$$P(r_1|V_j) = P(r_1), ..., P(r_{n_0}|V_j) = P(r_{n_0}) \Rightarrow q_1(V_j) = 0$$

$$P(r_1|V_j) = P(r_{n_0}), ..., P(r_{n_0}|V_j) = P(r_1) \Rightarrow q_1(V_j) = 0$$

J–ocena

Informacijska vsebina pogoja je pričakovana količino informacije, ki jo dobimo od V_j.

$$J_j = p_{.j} \sum_k p_{k|j} \log \frac{p_{k|j}}{p_{k.}}$$

J–ocena ima lepe lastnosti:

Nenegativnost: $J_j \geq 0$.

Zgornja meja J–ocene: odvod enak 0 tudi:

$$k_0 : p_{k_0|j} = 1 \wedge \forall k \neq k_0 : \ p_{k|j} = 0$$

$$J_j \leq p_{.j} \left(\min \left\{ \max_k - \log p_{k.}, \ - \log p_{.j} \right\} \right) \leq -p_{.j} \log p_{.j} \leq 0.53 \ bit$$

Relacija z informacijskim prispevkom: $\Sigma_j \, J_j = Gain(A)$

Porazdeljena po zakonu χ^2: $2nJ_j \log_e 2 = 1.3863nJ_j$

Gini–index

Apriorni Gini index:

$$Gini\_prior = \sum_k \sum_{l \neq k} p_{k.} p_{l.} = 1 - \sum_k p_{k.}^2$$

Apriorni Gini index je mera nečistoče.
Dokaz:
1. Maksimum:

$$Gini\_prior = 1 - \sum_k^{n_0-1} p_{k.}^2 - \left(1 - \sum_k^{n_0-1} p_{k.}\right)^2$$

$$\frac{\partial Gini\_prior}{\partial p_{k.}} = -2p_{k.} + 2\left(1 - \sum_k^{n_0-1} p_{k.}\right)$$

Prvi odvod je 0 v točki: $p_{k.} = 1/n_0$
Drugi odvod je v tej točki negativen (povsod je negativen):

$$\frac{\partial^2 Gini\_prior}{\partial p_{k.}^2} = -2 + 2(-1) = -4 < 0$$

2. Minimum:

$$Gini\_prior \geq 0 \wedge Gini\_prior = 0 \Leftrightarrow \exists! k : \; p_{k.} = 1$$

3. $Gini\_prior$ je simetrična funkcija.
4. $Gini\_prior$ je konkavna funkcija.
5. $Gini\_prior$ je zvezna in zvezno odvedljiva funkcija.

Pomembnost: razlika med apriornim in pričakovanim Gini–indexom:

$$Gini(A) = \sum_j p_{.j} \sum_k p_{k|j}^2 - \sum_k p_{k.}^2$$

$Gini(A)$ ima vse lastnosti, ki izvirajo iz definicije mere nečistoče:

Nenegativnost: $Gini(A) \geq 0$

Maksimalna vrednost:

$$Gini(A) = Gini\_prior \Leftrightarrow \forall j : \ \exists! k : \ p_{k|j} = 1$$

Večanje števila vrednosti atributa: $Gini(A)$ kvečjemu naraste.

Večanje števila razredov: $Gini(A)$ kvečjemu naraste.

ReliefF

A_1	A_2	A_3	R
0	0	0	0
0	0	1	0
0	1	1	1
0	1	0	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	0	0

```
function RELIEF(I: array[1..n] of instance): array[1..a] of real;
var inst,att : integer; W : array[1..a] of real;
  M,H : instance; (* najbližji pogrešek, zadetek *)
begin
 for att := 1 to a do W[att] := 0.0; end for
 for inst := 1 to n do
   primeru I[inst] poišči najbližji pogrešek M in zadetek H;
   for att := 1 to a do
     W[att] := W[att] - diff(att,I[inst],H)/n + diff(att,I[inst],M)/n;
    end for;
  end for;
 return(W);
end;
```

$$diff(A_i, u_j, u_k) = \begin{cases} \frac{|v^{(i,j)} - v^{(i,k)}|}{Max_i - Min_i}, & A_i \; je \; zvezni \\ 0, & v^{(i,j)} = v^{(i,k)} \wedge A_i \; je \; diskretni \\ 1, & v^{(i,j)} \neq v^{(i,k)} \wedge A_i \; je \; diskretni \end{cases}$$

Osnovni RELIEF:

- ocenjevanje zveznih in diskretnih atributov
- dvorazredni problemi
- kompleksnost: $O(n^2 \times a) \longrightarrow O(n \times m \times a), \quad m \in [30..200]$

Relacija z gini indeksom:

RELIEF oceni naslednjo razliko verjetnosti:

$$RELIEF(A_i) = P(diff(A_i, \cdot, \cdot) = 1 | najbližji\ primer\ iz\ nasprot.\ razr.) -$$

$$-P(diff(A_i, \cdot, \cdot) = 1 | najbližji\ primer\ iz\ istega\ razreda)$$

izpustimo pogoj bližine in dobimo funkcijo, ki je zelo podobna Gini–indeksu:

$$Relief(A_i) = P(diff(A_i, u_j, u_k) = 0 | r^{(j)} = r^{(k)}) -$$

$$-P(diff(A_i, u_j, u_k) = 0 | r^{(j)} \neq r^{(k)}) =$$

$$= constant \times \sum_j p_{.j}^2 \times Gini'(A_i)$$

Dokaz:

Zapišimo:

$$P_{eqval} = P(diff(A_i, u_j, u_k) = 0)$$

$$P_{samecl} = P(r^{(j)} = r^{(k)})$$

$$P_{samecl|eqval} = P(r^{(j)} = r^{(k)} | diff(A_i, u_j, u_k) = 0)$$

Če uporabimo Bayesovo pravilo, dobimo:

$$Relief(A_i) = \frac{P_{samecl|eqval} P_{eqval}}{P_{samecl}} - \frac{(1 - P_{samecl|eqval}) P_{eqval}}{1 - P_{samecl}}$$

$$P_{samecl} = \sum_k p_{k.}^2$$

$$P_{samecl|eqval} = \sum_j \left(\frac{p_{.j}^2}{\Sigma_j p_{.j}^2} \times \sum_k p_{k|j}^2 \right)$$

$$\begin{aligned} Relief(A_i) &= \frac{P_{eqval} \times Gini'(A_i)}{P_{samecl}(1 - P_{samecl})} \\ &= \frac{\Sigma_j p_{.j}^2 \times Gini'(A_i)}{\Sigma_k p_{k.}^2 (1 - \Sigma_k p_{k.}^2)} \\ &= constant \times \sum_j p_{.j}^2 \times Gini'(A_i) \end{aligned}$$

kjer je $Gini'(A)$ zelo podoben Gini–indeksu:

$$Gini'(A) = \sum_j \left(\frac{p_{.j}^2}{\Sigma_j p_{.j}^2} \times \sum_k p_{k|j}^2 \right) - \sum_k p_{k.}^2 \qquad (1)$$

Edina razlika je, da v Gini–indeksu nastopa faktor:

$$\frac{p_{.j}}{\Sigma_j p_{.j}} = p_{.j}$$

ReliefF vsebuje naslednje razširitve:

Neznane vrednosti atributov:

- prvi primer (u_j) nima vrednosti za dani atribut:

$$diff(A_i, u_j, u_k) = 1 - p_{v^{(i,k)}|r^{(j)}}$$

- če oba primera nimata vrednosti za dani primer:

$$diff(A_i, u_j, u_k) = 1 - \sum_{l=1}^{n_i} (p_{V_l|r^{(j)}} \times p_{V_l|r^{(k)}})$$

Šumni podatki: ReliefF poišče k najbližjih zadetkov in k najbižjih pogreškov ter njihove prispevke povpreči.

Več razredni problemi: ReliefF poišče k najbližjih pogreškov iz vseh razredov. Prispevki posameznih razredov so dodatno obteženi z apriornimi verjetnostmi razredov

function ReliefF(I: array[1..n] of instance; k : integer):
array[1..a] of real;
var
inst,att, kk, razr, r, rr : integer;
W : array[1..a] of real;
M : array[1..k,1..n_0] of instance;

```
begin
 for att := 1 to a do W[att] := 0.0; end for;
 for inst := 1 to n do for razr := 1 to n_0 do
    primeru I[inst] poišči k najbližjih primerov M[kk,razr] iz r_razr;
    r := arg_rr r_rr = r^(inst);
    for att := 1 to a do for kk := 1 to k do
     if razr = r then W[att] := W[att] - diff(att,I[inst],M[kk,razr])/(n*k)
      else W[att] := W[att] + p_razr./(1-p_r.) * diff(att,I[inst],M[kk,razr])/(n*k)
      end if;
     end for; end for;
  end for; end for;
 return(W);
end;
```

Mere za ocenjevanje atributov v regresiji

1. Razlika variance
2. Regresijski ReliefF (RReliefF)
3. MDL

Razlika variance pri regresijskih problemih

Pri regresijskih problemih namesto mere nečistoče uporabljamo *varianco* zveznega razreda:

$$s^2 = \frac{1}{n} \sum_{k=1}^{n} (r^{(k)} - \overline{r})^2$$

$$\overline{r} = \frac{1}{n} \sum_{k=1}^{n} r^{(k)}$$

Če v dvorazrednem klasifikacijskem problemu enemu razredu priredimo vrednost zveznega razreda 0, drugemu razredu pa vrednost zveznega razreda 1, potem velja:

$$Gini\_prior = 2s^2$$

Dokaz:

$$Gini\_prior = p_{1.}p_{2.} + p_{2.}p_{1.} = \frac{2n_{1.}n_{2.}}{n^2}$$

$$\overline{r} = \frac{n_{1.} \times 0 + n_{2.} \times 1}{n} = \frac{n_{2.}}{n}$$

$$\begin{aligned}
s^2 &= \frac{1}{n}\sum_{k=1}^{n}(r^{(k)} - \overline{r})^2 \\
&= \frac{1}{n}\left(\sum_{k=1}^{n_{1.}}\left(0 - \frac{n_{2.}}{n}\right)^2 + \sum_{k=1}^{n_{2.}}(1 - \frac{n_{2.}}{n})^2\right) \\
&= \frac{1}{n}\left(n_{1.}\left(\frac{n_{2.}}{n}\right)^2 + n_{2.}\left(\frac{n_{1.}}{n}\right)^2\right) \\
&= \frac{n_{1.}n_{2.}(n_{1.} + n_{2.})}{n^3} \\
&= \frac{n_{1.}n_{2.}}{n^2} \\
&= \frac{Gini_prior}{2}
\end{aligned}$$

Pomembnost atributa je (nenegativna)
pričakovana razlika variance:

$$ds^2(A_i) = \frac{1}{n}\sum_{k=1}^{n}(r^{(k)} - \overline{r})^2 - \sum_{j=1}^{n_i}\left(p_{.j}\frac{1}{n_{.j}}\sum_{k=1}^{n_{.j}}(r_j^{(k)} - \overline{r}_j)^2\right)$$

$r_j^{(k)}$ – vrednost zveznega razreda k-tega primera, ki ima j-to vrednost atributa A_i

$$\overline{r}_j = \frac{1}{n_{.j}}\sum_{k=1}^{n_{.j}} r_j^{(k)}$$

7 Predobdelava učnih primerov

1. Atributna predstavitev kompleksnih struktur,
2. diskretizacija zveznih atributov,
3. mehka diskretizacija zveznih atributov,
4. binarizacija atributov,
5. spreminajnje diskretnih atributov v zvezne,
6. obravnavanje neznanih vrednosti,
7. vizualizacija,
8. izbira podmnožice atributov.

Diskretizacija zveznih atributov

Vsaka diskretizacija pomeni kvečjemu izgubo informacije.

- optimalno število intervalov
- optimalne meje za vse intervale

Vrste metod za diskretizacijo

Učne primere najprej uredimo po vrednostih zveznega atributa.

- *od spodaj navzgor* (angl. bottom-up)
  n primerov: $O(n^2)$ ali
- *od zgoraj navzdol* (angl. top-down)
  k intervalov: $O(kn)$

Možne meje med intervali

$\phi(P(r_1), ..., P(r_{n_0}))$ - mera nečistoče

Ocena kvalitete atributa A pri uporabi meje z dvema podintervaloma:

$$q(A) = \phi(P(r_1), ..., P(r_{n_0})) - \sum_{j=1}^{2} P(V_j)\phi(P(r_1|V_j), ..., P(r_{n_0}|V_j))$$

Max zaporednih primerov iz istega razreda; meja: $x \in [0, Max]$
Funkcija $q(x)$ je konveksna:

$$\frac{\partial^2 q}{\partial x^2} > 0$$

in ima maksimum na eni od mejnih točk: $x = 0$ ali $x = Max$.

Dokaz:

$$\frac{\partial q}{\partial x} = -\frac{\phi(p_{1|1},...,p_{n_0-1|1})}{n} + \frac{n_{.1}+x}{n(n_{.1}+x)^2}\sum_{k=1}^{n_0-1} n_{k1}\frac{\partial\phi(p_{1|1},...,p_{n_0-1|1})}{\partial p_{k|1}} +$$

$$\frac{\phi(p_{1|2},...,p_{n_0-1|2})}{n} - \frac{n-n_{.1}-x}{n(n-n_{.1}-x)^2}\sum_{k=1}^{n_0-1} n_{k2}\frac{\partial\phi(p_{1|2},...,p_{n_0-1|2})}{\partial p_{k|2}}$$

$$\frac{\partial q}{\partial x} = -\frac{\phi(p_{1|1},...,p_{n_0-1|1})}{n} + \frac{1}{n(n_{.1}+x)}\sum_{k=1}^{n_0-1} n_{k1}\frac{\partial\phi(p_{1|1},...,p_{n_0-1|1})}{\partial p_{k|1}} +$$

$$\frac{\phi(p_{1|2},...,p_{n_0-1|2})}{n} - \frac{1}{n(n-n_{.1}-x)}\sum_{k=1}^{n_0-1} n_{k2}\frac{\partial\phi(p_{1|2},...,p_{n_0-1|2})}{\partial p_{k|2}}$$

$$\frac{\partial^2 q}{\partial x^2} = \frac{2-2}{n(n_{.1}+x)^2}\sum_{k=1}^{n_0-1} n_{k1}\frac{\partial\phi(p_{1|1},...,p_{n_0-1|1})}{\partial p_{k|1}} -$$

$$-\frac{1}{n(n_{.1}+x)^3}\sum_{k=1}^{n_0-1} n_{k1}^2\frac{\partial^2\phi(p_{1|1},...,p_{n_0-1|1})}{\partial p_{k|1}^2} +$$

$$+\frac{2-2}{n(n-n_{.1}-x)^2}\sum_{k=1}^{n_0-1} n_{k2}\frac{\partial\phi(p_{1|2},...,p_{n_0-1|2})}{\partial p_{k|2}} -$$

$$-\frac{1}{n(n-n_{.1}-x)^3}\sum_{k=1}^{n_0-1} n_{k2}^2\frac{\partial^2\phi(p_{1|2},...,p_{n_0-1|2})}{\partial p_{k|2}^2}$$

$$\frac{\partial^2 q}{\partial x^2} = -\frac{1}{n(n_{.1}+x)^3}\sum_{k=1}^{n_0-1} n_{k1}^2\frac{\partial^2\phi(p_{1|1},...,p_{n_0-1|1})}{\partial p_{k|1}^2} -$$

$$-\frac{1}{n(n-n_{.1}-x)^3}\sum_{k=1}^{n_0-1} n_{k2}^2\frac{\partial^2\phi(p_{1|2},...,p_{n_0-1|2})}{\partial p_{k|2}^2}$$

Torej je funkcija q konveksna:

$$\frac{\partial^2 q}{\partial x^2} > 0$$

in ima maksimum na eni od mejnih točk: $x = 0$ ali $x = Max$.

Mere za usmerjanje diskretizacije

Pričakovano zmanjašanje nečistoče q z dodajanjem novih mej kvečjemu naraste.

Informacijski prispevek in *Gini-indeks* nista primerna za diskretizacijo.

Bolj primerne so funkcije, ki ne precenjujejo večvrednostnih atributov, npr. $1 - D$, MDL, $ReliefF$.

Mehka diskretizacija zveznih atributov

Meja med dvema intervaloma = meja med dvema odločitvama.
intervala [0.00..0,30] in [0.31..1.00]:
primera 0.00 in 0.30 se obravnavata enako
primera 0.30 in 0.31 pa različno.

Mehka meja: verjetnostna distribucijo, da primer pripada eni oziroma drugi strani meje.

Druga možnost: mehke vrednosti originalnega zveznega atributa.

Primeru se pripiše vrednost diskretnega atributa z verjetnostjo, ki je enaka ploščini pod verjetnostno distribucijo na danem intervalu.

Binarizacija atributov

binarna odločitvena drevesa:

- rezultirajoče drevo manjše in s tem optimalnejše:
  - zmanjšanje problema podvajanja (replication problem).
  - preprečuje preveliko razdrobljenost učne množice,
- izognemo se problemu precenjevanja večvrednostnih atributov,
- nekatere ocene znajo ocenjevati samo binarne atribute (ORT).

Pri binarizaciji **zveznega atributa** uporabimo samo prvi korak diskretizacije atributa od zgoraj navzdol.

Pri binarizaciji **diskretnega atributa**:

- Za vsako vrednost atributa A generiramo en binarni atribut tako, da vse ostale vrednosti združimo v eno vrednost.

- Generiramo toliko binarnih atributov, kolikor je možnih različnih razbitij na dve disjunktni podmnožici:

$$\frac{1}{2}\sum_{k=1}^{n-1}\binom{n}{k} = 2^{n-1} - 1$$

Dvorazredni problemi

Vrednosti atributa $V_j, j = 1..n$ uredimo po naraščajočih pogojnih verjetnostih prvega razreda:

$$P(r_1|V_1) \leq P(r_1|V_2) \leq ... \leq P(r_1|V_n)$$

Velja, da je pomembnost atributa binarne verzije atributa A:

$$q(A) = \phi(P(r_1), ..., P(r_{n_0})) - \sum_{j=1}^{2} P(\mathcal{V}_j)\phi(P(r_1|\mathcal{V}_j), ..., P(r_{n_0}|\mathcal{V}_j))$$

največja za podmnožici $\mathcal{V}_j, j = 1, 2$, za kateri velja:

$$\mathcal{V}_1 = \{V_1, ..., V_l\}, \quad \mathcal{V}_2 = \{V_{l+1}, ..., V_n\}$$

za nek $0 < l < n$.

Večrazredni problemi

Požrešni algoritem za iskanje suboptimalne binarizacije atributa:

1. Izberi eno vrednost, ki maksimizira kvaliteto atributa, če ga obravnavamo kot binarnega z vsemi ostalimi vrednostmi združenimi v drugo vrednost.
2. Ponavljaj, dokler se kvaliteta atributa ne spreminja več:
   (a) Če v drugi množici vrednosti obstaja vrednost, ki premaknjena v prvo množico poveča kvaliteto atributa, premakni vrednost, ki maksimizira kvaliteto binarnega atributa.
   (b) Če v prvi množici vrednosti obstaja vrednost, ki premaknjena v drugo množico poveča kvaliteto atributa, premakni vrednost, ki maksimizira kvaliteto binarnega atributa.

Spreminajnje diskretnih atributov v zvezne

linearna in druge regresije, diskriminantne funkcije in nevronske mreže predpostavljajo: **vsi atributi zvezni**.

Binarni atribut – poseben primer zveznega atributa.

Večvrednostne diskretne atribute spremenimo v toliko binarnih atributov, kolikor ima originalni atribut različnih vrednosti.

Slaba stran: dobimo veliko med seboj odvisnih atributov.

Obravnavanje neznanih vrednosti

Če vrednost atributa A_i manjka, ga učna metoda

- ignorira,
- atributu doda ločeno vrednost (unknown),
- poskuša nadomestiti manjkajočo vrednost z najbolj verjetno vrednostjo ali
- uporabi verjetnostno distribucijo po posameznih vrednostih atributa.

- pogojne verjetnosti pri danem razredu primera;
- poseben učni algoritem, ki se nauči preslikave vrednosti ostalih atributov in razreda v atribut brez vrednosti.

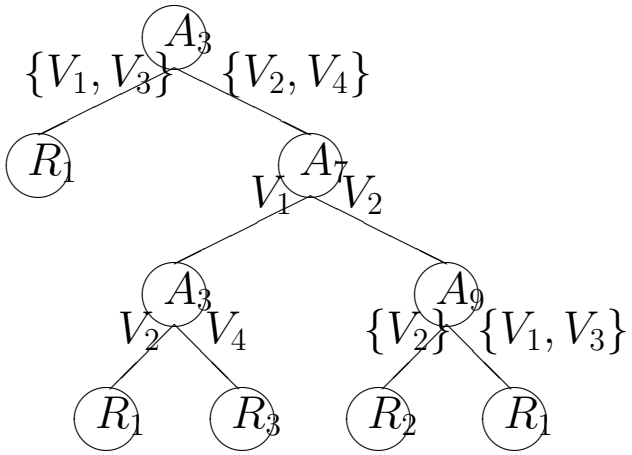
Obravnavanje posebnih vrednosti

- poljubna vrednost (“don’t care”)
- nesmiselna vrednost (“inapplicable”)

9 Symbolic learning

- decision trees
- decision rules
- association rules
- regression trees
- inductive logic programming (ILP)
- Naive and Semi-naive Bayesian classifiers
- Bayesian belief networks

Učenje odločitvenih dreves



Osnovni algoritem je sledeč (Hunt in sod., 1966):

Če je izpolnjen ustavitveni pogoj,
    potem postavi list, ki vključuje vse učne primere;
    sicer
        izberi "najboljši" atribut A_i;
        označi naslednike z vrednostmi atributa A_i;
        za vsako vrednost V_j atributa A_i **ponovi**:
            rekurzivno zgradi poddrevo z ustrezno
            podmnožico učnih primerov;

Pri tem je **ustavitveni pogoj** lahko:

- bodisi dovolj "čista" učna množica,
- bodisi je premalo učnih primerov,
- ali pa je zmanjkalo (dobrih) atributov.

Selection of "best" attribute:
information gain, gain-ratio, distance, MDL, Gini-index, ReliefF ...

Drevo predstavlja simbolični opis in povzetek zakonitosti.

- število učnih primerov v listu kaže na zanesljivost ocene verjetnostne porazdelitve razredov.
- prazen list (NULL): uporabimo naivni Bayesov klasifikator
- gradnja **binarnega** odločitvenega drevesa
  - ni dobro, da se učna množica razdeli prehitro na majhne podmnožice
  - **diskretni** atribut: lokalno optimizira kvaliteto binarne verzije atributa
  - **zvezni** atribut: upoštevamo samo vse možne meje
  - dobimo manjša drevesa, ponavadi tudi boljšo klasifikacijsko točnost

- obravnavanje **neznanih** in **poljubnih** vrednosti
  - primeri imajo utež
    - $* = 1$ – normalen primer
    - $* < 1$ – primer je nezanesljiv zaradi manjkajoče vrednosti
    - $* > 1$ – primer ima poljubno vrednost (don't care)
  - pogojna verjetnost, da primer ustreza veji:

$$P(V_i|R) = \frac{N_{V_i,R} + mP(R)}{N_R + m}$$

- **rezanje** odločitvenega drevesa
  - zaradi nezanesljivosti nižjih nivojev drevesa
  - drevo se *naknadno poreže* (postpruning):

    Za vsa vozlišča od spodaj navzgor **ponavljaj**:
    oceni pričakovano napako v poddrevesih;
    oceni pričakovano napako v vozlišču;
    če je napaka poddreves večja,
    potem poreži poddrevesa in postavi v list.

- a ocenjevanje napake lahko uporabljamo m-oceno verjetnosti.
- MDL princip:
  * ocenjuje dolžino kodiranja drevesa in porazdelitev razredov učnih primerov v listih
  * Če je dolžina kodiranja porazdelitve v vozlišču manjša od vsote dolžin kodiranj poddreves plus 1, potem poddrevesa porežemo.
  * MDL ne zahteva nastavitve nobenega parametra
  * skoraj optimalno poreže drevesa

Naivni Bayesov klasifikator

pogojna neodvisnost atributov pri danem razredu:

$$P(r_k|V) = P(r_k) \prod_{i=1}^{a} \frac{P(r_k|v_i)}{P(r_k)}$$

apriorne verjetnosti: Laplaceov zakon zaporednosti:

$$P(r_k) = \frac{N_k + 1}{N + n_0}$$

pogojne verjetnosti: m-ocena:

$$P(r_k|v_i) = \frac{N_{k,i} + mP(r_k)}{N_i + m}$$

Lastnosti Naivnega Bayesa

- Bayesov klasifikator vedno uporabi **vse znane atribute**
- **neznane vrednosti**: primere izpustimo
- **zvezni atribut** najprej (*mehko*) diskretiziramo
- pogojna neodvisnost pogosto sprejemljiva
  - Simptomi pri pacientih so odvisni od bolezni.
  - Prikazovalnik cifer LCD: okvare žarnic neodvisne
- **Ocenjevanje verjetnosti** relativno zanesljivo:
  ne pride do prevelikega prilagajanja učni množici.

Lastnosti Naivnega Bayesa

- Če neodvisnost ni popolna:
  še vedno dovolj "rezerve" (predpostavka neodvisnosti ne "pokvari" vrstnega reda verjetnosti razredov).
- **Močne odvisnosti** med atributi: naivni Bayes odpove
- **Razlaga odločitev** naivnega Bayesa:

  $$-\log_2 P(r_k|V) = -\log_2 P(r_k) - \sum_{i=1}^{a} (\log_2 P(r_k|v_i) - \log_2 P(r_k))$$

  apriorna količino informacije, da primer razvrstimo v razred r_k, minus vsota informacijskih prispevkov posameznih atributov.
- **Inkrementalno učenje**

10 Numerične metode

1. metoda najbližjih sosedov (k-NN),
2. učenje diskriminantnih funkcij,
3. linearna regresija,
4. metode podpornih vektorjev (SVM)

Metode najbližjih sosedov

Učenje pomeni shranitev učnih primerov - lazy learning.

Predikcija: poišči podmnožico *podobnih* primerov.

1. k-najbližjih sosedov
2. z razdaljo uteženih k-najbližjih sosedov
3. lokalno utežena regresija.

K-najbližjih sosedov

novemu u_x, poiščemo k najbližjih u_1, ..., u_k

Pri klasifikaciji napovemo večinski razred

$$r_x = argmax_{r \in \{V_1,...,V_{n_0}\}} \sum_{i=1}^{k} \delta(r, r^{(i)})$$

kjer je

$$\delta(a,b) = \begin{cases} 1, & a = b \\ 0, & a \neq b \end{cases}$$

Pri regresiji napovemo povprečno vrednost razreda:

$$r_x = \frac{1}{k} \sum_{i=1}^{k} r^{(i)}$$

Parameter k

k običajno nastavimo na liho število

Če v učnih podatkih ni napak, potem najbolje 1-NN.

Če so napake, s k povprečimo napovedi več bližnjih primerov.

Eksperimentalno določamo optimalni k.

k ne določa velikosti okolice novega primera,
ampak se okolica dinamično spreminja

Metrika:

k-NN je občutljiv na izbrano metriko.

Evklidska razdalja: $D(u_l, u_j) = \sqrt{\sum_{i=1}^{a} d(v^{(i,l)}, v^{(i,j)})^2}$

kjer za zvezni atribut A_i velja:

$$d(v^{(i,l)}, v^{(i,j)}) = |v^{(i,l)} - v^{(i,j)}|$$

in za diskretnega:

$$d(v^{(i,l)}, v^{(i,j)}) = \begin{cases} 0, & v^{(i,l)} = v^{(i,j)} \\ 1, & v^{(i,l)} \neq v^{(i,j)} \end{cases}$$

Uteževanje vpliva atributov na celotno razdaljo:

$$D(u_l, u_j) = \sqrt{\sum_{i=1}^{a} q(A_i) d(v^{(i,l)}, v^{(i,j)})^2}$$

Z razdaljo uteženih k-najbližjih sosedov

Razdalja vpliva linearno, kvadratično, eksponentno itd.

Kvadratični vpliv razdalje pri klasifikaciji:

$$r_x = argmax_{r \in \{V_1, \ldots, V_{n_0}\}} \sum_{i=1}^{k} \frac{\delta(r, r^{(i)})}{D(u_x, u_i)^2}$$

in pri regresiji:

$$r_x = \frac{\Sigma_{i=1}^{k} \left[r^{(i)} / D(u_x, u_i)^2 \right]}{\Sigma_{i=1}^{k} \left[1 / D(u_x, u_i)^2 \right]}$$

k nepotreben: na napoved vplivajo vsi učni primeri.

Lokalno utežena regresija

Namesto povprečenja uporabimo poljubno regresijsko funkcijo skozi k najbližjih sosedov.

Najpogosteje linearna lokalno utežena regresija, ki je linearna regresija, uporabljena na k najbližjih sosedih:

- nevarnost prevelikega prileganja;
- časovna zahtevnost!

11 Umetne nevronske mreže

Intelligence emerges from the interaction of large numbers of simple processing units.

David Rumelhart & John McClelland

1. Uvod
   (a) Preprost primer nevronske mreže
   (b) Porazdeljeni pomnilnik
   (c) Lastnosti nevronskih mrež
   (d) Primeri uporabe
   (e) Relacija z metodami umetne inteligence

2. Vrste nevronskih mrež
   (a) Topologija
   (b) Namen
   (c) Pravilo učenja
   (d) Kombinacijska funkcija
3. Hopfieldova nevronska mreža
4. Bayesova nevronska mreža
5. Perceptron
   (a) Dvonivojski perceptron
   (b) Večnivojski perceptron

Preprost primer nevronske mreže

Zgled matričnega računa.
Naloga: razpoznavati naslednja dva vzorca:

$$X_1 = (1, 1, 1)^T$$

in

$$X_2 = (1, -1, -1)^T$$

Sestavimo matriko:

$$M = X_1 X_1^T + X_2 X_2^T = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 2 \\ 0 & 2 & 2 \end{bmatrix}$$

Definirajmo še odločitveno funkcijo:

$$f(X) = \begin{cases} 1, & X > 0 \\ 0, & X = 0 \\ -1, & X < 0 \end{cases}$$

Sedaj velja:

$$f(MX_1) = f((2,4,4)^T) = (1,1,1)^T = X_1$$

$$f(MX_2) = f((2,-4,-4)^T) = (1,-1,-1)^T = X_2$$

Poskusimo z delno poznanimi vektorji:

$$X_1' = (1, 1, 0)^T$$

$$X_2' = (1, 0, -1)^T$$

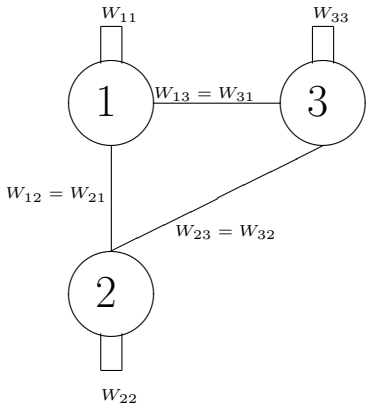
$$f(MX_1') = f((2, 2, 2)^T) = (1, 1, 1)^T = X_1$$

$$f(MX_2') = f((2, -2, -2)^T) = (1, -1, -1)^T = X_2$$

Poskusimo še z napačnimi vrednostmi:

$$X_1'' = (1, 1, -1)^T = X_2''$$

$$f(MX_1'') = f((2, 0, 0)^T) = (1, 0, 0)^T$$



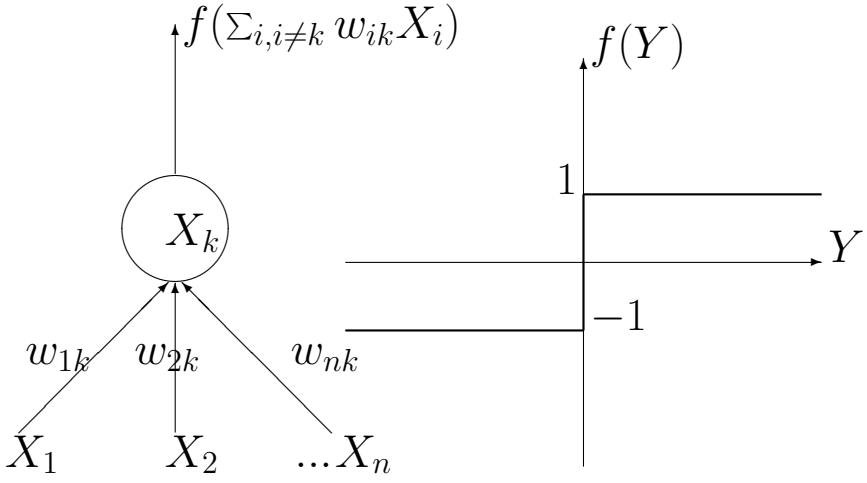
Vsak nevron povezan z vsakim z dvosmerno sinapso.
Stanje nevrona lahko 1 ali –1.
Vsaka vez ima pridruženo realno utež.

Pred učenjem so vse uteži enake 0.
Za vsak učni primer:
    Če imata nevrona enake vrednosti,
      se utež vezi poveča za 1,
      sicer se zmanjša za 1

Izvajanje: nevroni izračunavajo izhod po pravilu:

$$Y_i = f(\sum_j W_{ji} X_j)$$

X_j – trenutno stanje j-tega nevrona,
W_{ji} – utež sinapse med j-tim in i-tim nevronom
Y_i – novo stanje i-tega nevrona
f – izhodna funkcija



- sinhrono/asinhrono izvajanje
- konvergenca učenja/izvajanja
- vzbujevalna/zaviralna sinapsa

Porazdeljeni pomnilnik

Lastnosti porazdeljene predstavitve so:

- avtomatska posplošitev
- prilagodljivost na spreminjajoče se okolje
- čim večja je porazdeljenost, tem bolj natančno lahko shranimo iste podatke z enakim številom nevronov
- konstruktivni značaj: vsak procesni element kombinira več osnovnih atributov
- robustnost: odstranitev enega nevrona povzroči delni padec v natančnosti za več podatkov in ne popolne izgube enega

- spontano spominjanje pozabljenih podatkov med ponovnim učenjem drugih podatkov
- vsebinsko naslavljanje (ni razlike med podatkom in njegovim naslovom)
- moduli : podatek je lokalno shranjen z globalnega gledišča, vendar globalno shranjen z lokalnega gledišča

Lastnosti umetnih nevronskih mrež

- Biološka podobnost
- Paralelizem
- Večsmerno izvajanje
- Robustnost glede na
  - okvaro nevronov
  - okvaro sinaps
  - pomanjkljive vhodne podatke
- Učenje: sinapsa predstavlja povezanost med aktivnostima dveh nevronov

- Relacija med strojno in programsko opremo:
  - ni programske opreme
  - "relaksator" namesto "računalnik"
  - Strojna oprema je lahko fiksna ali pa se spreminja. Konfiguracija je lahko naključna, kar je analogno z možgani.
- Matematična podlaga: linearna algebra
- Slabost: razlaga odločitve

Primeri uporabe

- kontrola kvalitete izdelave trdih diskov
- diagnosticiranje pacientov na osnovi signala EKG
- razpoznavanje ročno zapisanih znakov
- preverjanje podpisov na čekih
- identifikacija objektov z radarjem.
- računalniško razpoznavanje govora
- računalniški vid (razpoznavanje invariantno glede na velikost in translacijo in delno invariantno glede na deformacijo)

- analiza in procesiranje signalov akustične emisije (nova generacija merilnih instrumentov)
- približne rešitve NP-polnih problemov (v linearnem ali celo konstantnem času)
- vodenje dinamičnih sistemov
- kognitivno modeliranje

Analogija z možgani

- Možgani so vsebinsko naslovljivi
- Ni možno lokalno ločiti funkcije pomnilnika od procesorjev.
- Pomnilnik v možganski skorji je porazdeljenega tipa.
- Spomin v možganih: sinaptične povezave.
- Hitrost nevronov v možganih je velikostnega reda milisekunde. Percepcija, procesiranje jezika, intuitivno razmišljanje in dostop do spomina zahtevajo v možganih približno desetinko sekunde, torej kakih 100 korakov.
- Učinkovitost počasi pada s številom uničenih nevronov.

- Človeška percepcija je do določene mere invariantna glede na velikost, translacijo in rotacijo.
- Avtomatično generira skrčeno predstavitev.
- Ohranjajo topologijo prostorskih senzorjev (samoorganiziranje).
- 10^{10} nevronov in 10^4 povezav na nevron: po principu preprostega pragovnega elementa imamo zadostno kapaciteto, da z natančnostjo shranimo vse, kar doživimo v 100 letih.
- Geni ne določajo vseh povezav v možganih V možganih ni niti "hardware-a" v strogem pomenu besede niti "software-a" v običajnem pomenu besede.
- Rekurzija ni domača človekovemu načinu razmišljanja.

Razlike:

- Ali vzbujevalne ali zaviralne vezi.
- Velikost signalov je podana s frekvenco impulzov.
- Današnji modeli nevronskih mrež ne upoštevajo globalne komunikacije.

Relacija z metodami umetne inteligence

umetna inteligenca	**umetne nevronske mreže**
simbolični nivo	subsimbolični nivo
razlaga odločitve	včasih tudi
eksplicitno shranjena pravila	dinamično kreirana pravila
zaporedno procesiranje	vzporedno procesiranje
logični, zavestni nivo	intuitivni, podzavestni nivo
psihološka analogija	tudi
ni podobnosti z biološkimi sist.	je podobnost
leva možganska polobla	desna možganska polobla

Vrste nevronskih mrež

Nevronske mreže delimo po naslednjih kriterijih:

- topologija nevronske mreže,
- namen nevronske mreže,
- pravilo učenja in
- funkcija kombiniranja vhodov nevrona v izhod.

Funkcija kombiniranja vhodov nevrona v izhod

Funkcija aktivacije:

Utežena vsota:

$$A(X_j) = \sum_i W_{ij} X_i + C_j$$

Funkcija *sigma-pi*:

$$sp(X_1, ..., X_n) = \sum_{S_i \in P} W_i \prod_{k \in S_i} X_k$$

P – potenčna množica množice indeksov vseh nevronov

Naivni Bayes

Izhodna funkcija je lahko

– deterministična binarna,

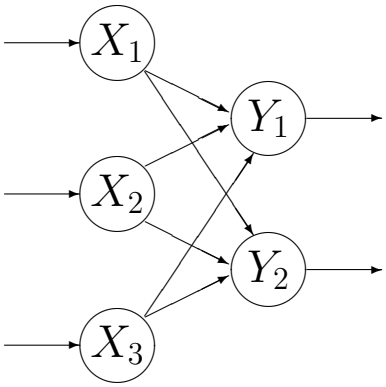
– deterministična zvezna:

$$f(X) = \frac{1}{1 + e^{-X}}$$

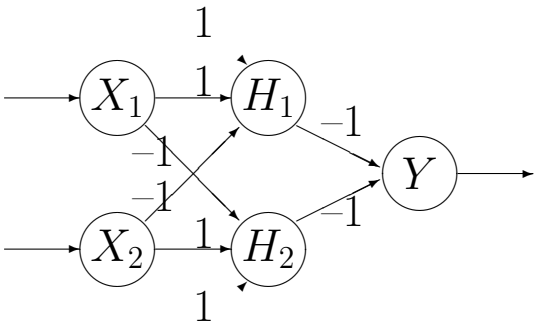
– stohastična.

Topologija nevronske mreže

- Brez nivojev
- Dvonivojske usmerjene nevronske mreže



- Večnivojske usmerjene nevronske mreže



X_1	X_2	Y
1	1	−1
1	−1	1
−1	1	1
−1	−1	−1

- Dvosmerni asociativni pomnilnik

Namen nevronske mreže

- Avtoasociativni pomnilnik
- Heteroasociativni pomnilnik
- Časovni asociativni pomnilnik
- Klasifikacija
- Razvrščanje (clustering)
- Samoorganizacija in sortiranje

Pravilo učenja

- Hebbovo pravilo
- Posplošeno Hebbovo pravilo
- Pravilo delta
- Posplošeno pravilo delta (pravilo vzvratnega razširjanja napake – backpropagation)
- Tekmovalno pravilo
- Pozabljanje

Perceptron

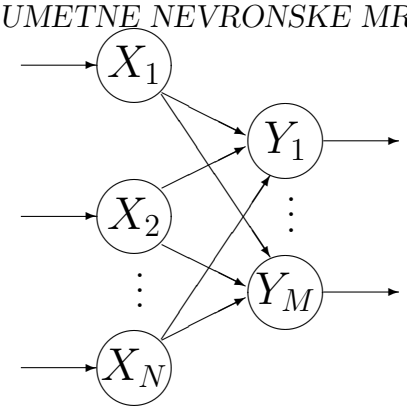
Dvonivojski perceptron

Dvonivojska usmerjena nevronska mreža:

$$Y = WX$$

$$Y_i = \sum_{j=1}^{N} W_{ji} X_j + \theta_i$$

Dvonivojske nevronske mreže lahko rešijo samo linearne probleme



Lahko se omejimo na samo en izhodni nevron.

$$Y = W^T X$$

$Y \geq 0$, potem klasificira v prvi razred
$Y < 0$, potem klasificira v drugi razred

Učenje

1. Pred začetkom učenja so uteži izbrane naključno (lahko vse 0).
2. Vhodni nevroni dobijo vhodni vzorec.
3. Zatem se v enem koraku izračuna izhod.
4. Če je n-ti vhodni vzorec $X(n)$ pravilno klasificiran,

se vrednosti uteži ne spremenijo, sicer se uteži spremenijo:

4.1. če je pravilni razred prvi in je $W^T(n)X(n) < 0$:

$$W(n+1) = W(n) + \eta X(n)$$

4.2. če je pravilni razred $X(n)$ drugi in je $W^T(n)X(n) \geq 0$:

$$W(n+1) = W(n) - \eta X(n)$$

η – *stopnja učenja* (learning rate)

Pravilo delta

Upošteva razliko med izhodom Y in želenim izhodom d:

$$W(n+1) = W(n) + \eta\Big(d(n) - Y(n)\Big)X(n)$$

$$W(n+1) = W(n) + \eta\Big(d(n) - W^T(n)X(n)\Big)X(n)$$

Temu pravilu pravimo tudi *gradientno pravilo.*
Kvadrat napake $E(n)$ je namreč podan z:

$$E(n) = \Big(d(n) - W^T(n)X(n)\Big)^2$$

in je odvod napake enak

$$\frac{dE(n)}{dW(n)} = -2\Big(d(n) - W^T(n)X(n)\Big)X(n)$$

Paketna **varianta pravila delta:**

Izračuna razliko med želenim in dejanskim izhodom za vse učne primere, $X(1)$, ..., $X(m)$.

Napaka je podana z: $E(n) = \Sigma_{i=1}^{m} \Big(d(i) - W^T(n)X(i)\Big)^2$

odvod napake pa z:

$$\frac{dE(n)}{dW(n)} = -2 \sum_{i=1}^{m} \Big(d(i) - W^T(n)X(i)\Big)X(i)$$

Paketno pravilo delta je:

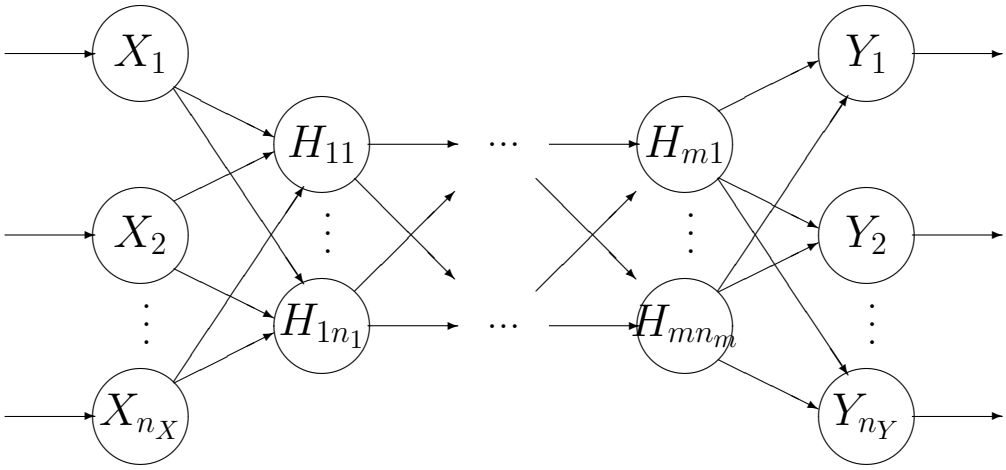
$$W(n+1) = W(n)+\eta\frac{dE(n)}{dW(n)} = W(n)+\eta \sum_{i=1}^{m} \Big(d(i)-W^T(n)X(i)\Big)X(i)$$

Paketno pravilo upošteva več informacije naenkrat.

Neprimerno za (hardware-sko) implementacijo z nevronskimi mrežami.

Večnivojski perceptron

Z večnivojskimi mrežami lahko rešimo tudi nelinearne probleme.



Posplošeno pravilo delta **ali** *pravilo vzvratnega razširjanja napake* **(backpropagation of error):**

1. Na začetku so uteži naključne.
2. Na vhodu mreža dobi vhodni vzorec in izračuna izhod.
3. Zatem se izračuna razlika med dejanskim in želenim izhodom.
4. Najprej se spremenijo uteži med zadnjim in predzadnjim nivojem kot pri osnovnem pravilu delta.

5.1 Zatem se izračunajo želene vrednosti nevronov na predzadnjem nivoju.

5.2 Izračuna se razlika med želenimi in dejanskimi vrednostmi nevronov na predzadnjem nivoju.

5.3 Rekurzivno se nadaljuje spreminjanje uteži vse do vhodnega nivoja nevronov.

Računanje izhoda (razširjanje naprej)

Izhodna funkcija mora biti zvezna in zvezno odvedljiva:

$$f(X) = \frac{1}{1 + e^{-X}}$$

Odvod funkcije f je:

$$f'(X) = \frac{e^{-X}}{(1 + e^{-X})^2} = f(X)(1 - f(X))$$

Nevron na prvem skritem nivoju izračunava svoj izhod pri učnem primeru $X(n)$ po enačbi:

$$H_{1i}(n) = f\Big(\sum_{j=1}^{n_X} W_{ji}^{(1)}(n) X_j(n) \Big)$$

Definirajmo še: $A_{1i}(n) = \Sigma_{j=1}^{n_X} W_{ji}^{(1)}(n) X_j(n)$
Na k-tem skritem nivoju, $k > 1$, nevroni izračunavajo svoj izhod po enačbi

$$H_{ki}(n) = f(A_{ki}(n))$$

kjer je

$$A_{ki}(n) = \sum_{j=1}^{n_k} W_{ji}^{(k)}(n) H_{k-1,j}(n)$$

Izhodni nevroni pa izračunavajo svoj izhod po enačbi:

$$Y_i(n) = f(A_i(n))$$

kjer je

$$A_i(n) = \sum_{j=1}^{n_m} W_{ji}^{(m+1)}(n) H_{m,j}(n)$$

Popravek na izhodnih nevronih:

Napaka i-tega izhodnega nevrona za dani n-ti učni primer $X(n)$:

$$e_i(n) = d(n) - Y_i(n)$$

Napako za celotno mrežo pa izračunamo kot polovico vsote kvadratov napak po vseh izhodnih nevronih:

$$E(n) = \frac{1}{2} \sum_{i=1}^{n_Y} e_i^2(n)$$

Parcialni odvod napake celotne mreže po uteži $W_{ji}^{(m+1)}$:

$$\frac{\partial E(n)}{\partial W_{ji}^{(m+1)}(n)} = \frac{\partial E(n)}{\partial e_i(n)} \frac{\partial e_i(n)}{\partial Y_i(n)} \frac{\partial Y_i(n)}{\partial A_i(n)} \frac{\partial A_i(n)}{\partial W_{ji}^{(m+1)}(n)}$$

Posamezni odvodi so:

$$\frac{\partial E(n)}{\partial e_i(n)} = e_i(n)$$

$$\frac{\partial e_i(n)}{\partial Y_i(n)} = -1$$

$$\frac{\partial Y_i(n)}{\partial A_i(n)} = f'(A_i(n))$$

$$\frac{\partial A_i(n)}{\partial W_{ji}^{(m+1)}(n)} = H_{mj}(n)$$

Popravek uteži $W^{(m+1)}$ pri izhodnih nevronih je:

$$W_{ji}^{(m+1)}(n+1) = W_{ji}^{(m+1)}(n) - \eta \frac{\partial E(n)}{\partial W_{ji}^{(m+1)}(n)}$$

$$W_{ji}^{(m+1)}(n+1) = W_{ji}^{(m+1)}(n) - \eta \frac{\partial E(n)}{\partial A_i(n)} H_{mj}(n)$$

$$W_{ji}^{(m+1)}(n+1) = W_{ji}^{(m+1)}(n) - \eta e_i(n) f'(A_i(n)) H_{mj}(n)$$

Vzvratno razširjanje napake

Izračunamo parcialne odvode napake po izhodnih vrednostih skritega nevrona:

$$\frac{\partial E(n)}{\partial A_{mi}(n)} = \frac{\partial E(n)}{\partial H_{mi}(n)} \frac{\partial H_{mi}(n)}{\partial A_{mi}(n)}$$

$$\frac{\partial H_{mi}(n)}{\partial A_{mi}(n)} = f'(A_{mi}(n))$$

Prvi faktor pa izračunamo po verižnem pravilu:

$$\frac{\partial E(n)}{\partial H_{mi}(n)} = \sum_{j=1}^{n_Y} \frac{\partial E(n)}{\partial A_j(n)} \frac{\partial A_j(n)}{\partial H_{mi}(n)}$$

$$\frac{\partial E(n)}{\partial H_{mi}(n)} = \sum_{j=1}^{n_Y} \frac{\partial E(n)}{\partial A_j(n)} \frac{\partial \left(\sum_{l=1}^{n_m} W_{lj}^{(m+1)}(n) H_{m,l}(n) \right)}{\partial H_{mi}(n)}$$

$$\frac{\partial E(n)}{\partial H_{mi}(n)} = \sum_{j=1}^{n_Y} \frac{\partial E(n)}{\partial A_j(n)} W_{ij}^{(m+1)}(n)$$

Torej dobimo za zadnji (m-ti) skriti nivo:

$$\frac{\partial E(n)}{\partial A_{mi}(n)} = f'(A_{mi}(n)) \sum_{j=1}^{n_Y} \frac{\partial E(n)}{\partial A_j(n)} W_{ij}^{(m+1)}(n)$$

$$\frac{\partial A_{mi}(n)}{\partial W_{ji}^{(m)}(n)} = H_{m-1,j}(n)$$

Torej pravilo delta zanj je:

$$W_{ji}^{(m)}(n+1) = W_{ji}^{(m)}(n) - \eta \frac{\partial E(n)}{\partial W_{ji}^{(m)}(n)}$$

$$W_{ji}^{(m)}(n+1) = W_{ji}^{(m)}(n) - \eta \frac{\partial E(n)}{\partial A_{mi}(n)} H_{m-1,j}(n)$$

$$W_{ji}^{(m)}(n+1) = W_{ji}^{(m)}(n) - \eta f'(A_{mi}(n)) \left(\sum_{l=1}^{n_Y} \frac{\partial E(n)}{\partial A_l(n)} W_{il}^{(m+1)}(n) \right) H_{m-1,j}(n)$$

Ostali skriti nivoji:

V splošnem pa za k-ti skriti nivo,$1 < k < m$, velja:

$$\frac{\partial A_{ki}(n)}{\partial W_{ji}^{(k)}(n)} = H_{k-1,j}(n)$$

in je torej pravilo delta zanj:

$$W_{ji}^{(k)}(n+1) = W_{ji}^{(k)}(n) - \eta f'(A_{ki}(n)) \left(\sum_{l=1}^{n_{k+1}} \frac{\partial E(n)}{\partial A_{k+1,l}(n)} W_{il}^{(k+1)}(n) \right) H_{k-1,j}(n)$$

In za prvi skriti nivo ($k = 1$) velja:

$$\frac{\partial A_{1i}(n)}{\partial W_{ji}^{(1)}(n)} = X_j(n)$$

pravilo delta pa je zanj sledeče:

$$W_{ji}^{(1)}(n+1) = W_{ji}^{(1)}(n) - \eta f'(A_{1i}(n)) \left(\sum_{l=1}^{n_2} \frac{\partial E(n)}{\partial A_{2,l}(n)} W_{il}^{(2)}(n) \right) X_j(n)$$

Slabosti posplošenega pravila delta

1. Ne konvergira vedno k optimalni mreži (lahko obtiči v lokalnem minimumu).
2. Problematična je izbira topologije mreže.
3. Preveliko prileganja učni množici.
4. Nastavitev parametra η vpliva na stabilnost.
5. Zahteva zelo veliko število prehodov preko učnih primerov.
6. Pravilo delta nima biološke analogije z možgani.

Prve tri probleme se da omiliti...

1. Vpeljemo momentni člen:

$$W_{ji}(n+1) = W_{ji}(n) - \Delta W_{ji}(n+1) - \alpha \Delta W_{ji}(n)$$

2. Metoda eliminacije uteži funkciji napake doda člen, ki "kaznuje" velike uteži.

   - Učenje začnemo s preveliko mrežo in med učenjem se odstranijo odvečni nevroni.
   - Hkrati se zaradi avtomatske nastavitve optimalne velikosti mreže izognemo tudi prevelikemu prileganju učni množici.
   - Zato pa metoda vpelje še dodatne parametre za kontrolo učenja, ki jih ni trivialno nastaviti.