

PRIRUČNIK ZA PRIPREMU PRIJEMNOG ISPITA IZ VEŠTAČKE INTELIGENCIJE ZA MASTER STUDIJE SOFTVERSKO INŽENJERSTVO I VEŠTAČKA INTELIGENCIJA

PROF. JELENA JOVANOVIĆ

PROF. BOJAN TOMIĆ

PROF. ZORAN ŠEVARAC

1.

MAŠINSKO UČENJE

PREGLED TEMA

- Šta je mašinsko učenje?
- Zašto (je potrebno/bitno) m. učenje?
- Oblasti primene m. učenja
- Oblici m. učenja
- Osnovni koraci i elementi procesa m. učenja
 - Podaci
 - Atributi (features)
 - Odabir algoritma m. učenja
 - Validacija i testiranje kreiranog modela

ŠTA JE MAŠINSKO UČENJE ?

Mašinsko učenje se odnosi na sposobnost softverskog sistema da:

- *generalizuje* na osnovu prethodnog *iskustva*, i
- da koristi kreirane generalizacije kako bi pružio odgovore na pitanja koja se odnose na entitete/pojave koje pre nije sretao

Pod iskustvom se podrazmeva skup podataka o pojavama / entitetima koji su predmet učenja

ŠTA JE MAŠINSKO UČENJE ?

Za kompjuterski program se kaže da uči

iz iskustva ***E*** (*experience*),

vezanog za zadatak ***T*** (*task*), i

meru performansi ***P*** (*performance*),

ukoliko se njegove performanse na zadatku ***T***, merene

metrikama ***P***, unapređuju sa iskustvom ***E***

Tom Mitchell (1997)

ŠTA JE MAŠINSKO UČENJE ?

Primer: program koji označava poruke kao spam i non-spam

- Zadatak (***T***): klasifikacija email poruka na spam i non-spam
- Iskustvo (***E***): email poruke označene kao spam i non-spam; podaci o oblicima interakcije korisnika sa spam i non-spam porukama
- Performanse (***P***): procenat email poruka korektno klasifikovanih kao spam, odnosno non-spam

ZAŠTO MAŠINSKO UČENJE ?

1) Neke vrste zadataka ljudi rešavaju vrlo lako, a pri tome nisu u mogućnosti da precizno (algoritamski) opišu kako to rade

Primeri: prepoznavanje slika, zvuka, govora

2) Za neke vrste zadataka mogu se definisati algoritmi za rešavanje, ali su ti algoritmi vrlo složeni i/ili zahtevaju velike baze znanja

Primeri: automatsko prevođenje

ZAŠTO MAŠINSKO UČENJE ?

3) U mnogim oblastima se kontinuirano prikupljaju podaci sa ciljem da se iz njih “nešto sazna”; npr.:

- u medicini: podaci o pacijentima i korišćenim terapijama
- u sportu: o odigranim utakmicama i igri pojedinih igrača
- u marketingu: o korisnicima/kupcima i tome šta su kupili, za šta su se interesovali, kako su proizvode ocenili,...

Analiza podataka ovog tipa zahteva pristupe koji će omogućiti da se otkriju pravilnosti, zakonitosti u podacima koje nisu ni poznate, ni očigledne, a mogu biti korisne

GDE SE PRIMENJUJE MAŠINSKO UČENJE ?

Brojne oblasti primene

- Kategorizacija teksta prema temi, iskazanim osećanjima i/ili stavovima i sl.
- Mašinsko prevođenje teksta
- Razumevanje govornog jezika
- Prepoznavanje lica na slikama
- Segmentacija tržišta
- Uočavanje paternu u korišćenju različitih aplikacija
- Autonomna vozila (self-driving cars)
- ...

OBlici MAŠINSKOG UČENJA

Osnovni oblici mašinskog učenja:

- Nadgledano učenje (supervised learning)
- Nenadgledano učenje (unsupervised learning)
- Učenje uz podsticaje (reinforced learning)

NADGLEDANO UČENJE

Obuhvata skup problema i tehnika za njihovo rešavanje u kojima program koji uči dobija:

- skup ulaznih podataka (x_1, x_2, \dots, x_n) i
- skup željenih/tačnih vrednosti, tako da za svaki ulazni podatak x_i , imamo željeni/tačan izlaz y_i

Zadatak programa je da “nauči” kako da novom, neobeleženom ulaznom podatku dodeli tačnu izlaznu vrednost

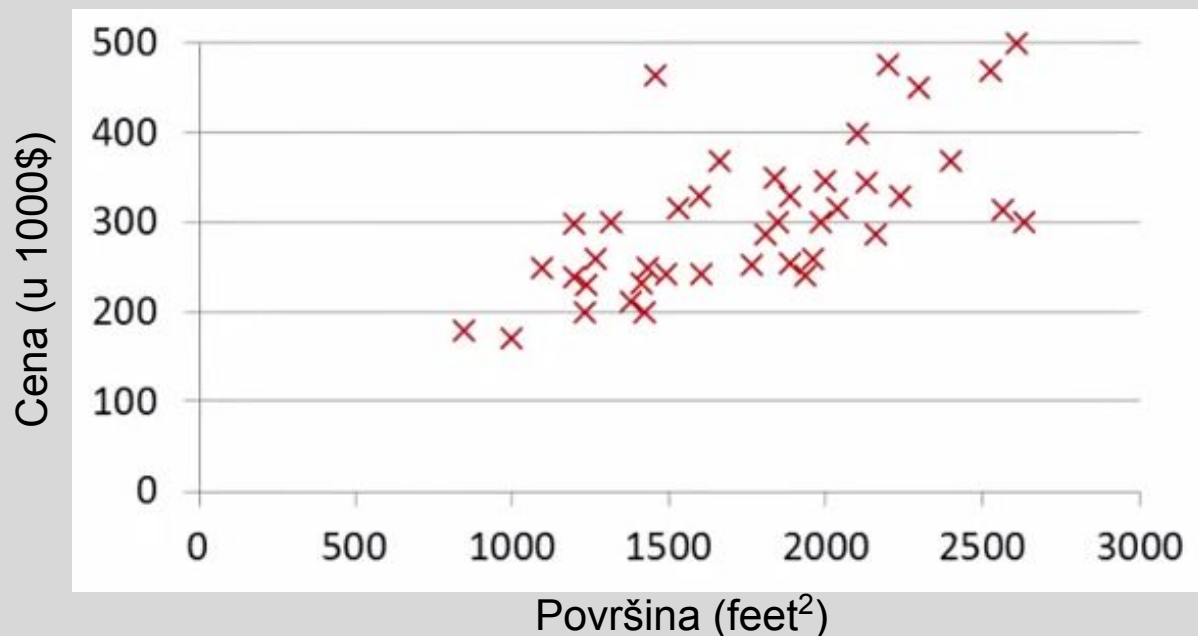
Izlazna vrednost može biti:

- labela (tj. nominalna vrednost) – reč je o *klasifikaciji*
- realan broj – reč je o *regresiji*

NADGLEDANO UČENJE

Primer linearne regresije: predikcija cena nekretnina na osnovu njihove površine

Podaci za učenje: površine (x) i cene (y) nekretnina u nekom gradu



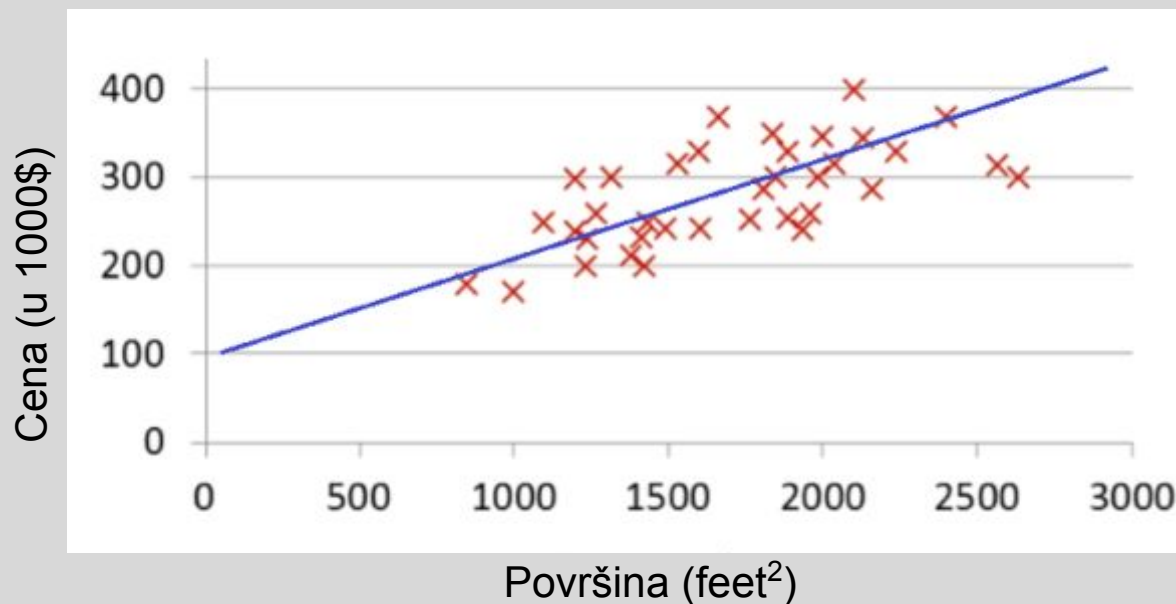
NADGLEDANO UČENJE

Primer linearne regresije (nastavak)

Funkcija koju treba „naučiti“ u ovom slučaju (samo jedan atribut) je:

$$h(x) = a + bx$$

a i b su koeficijenti koje program u procesu „učenja“ treba da *proceni* na osnovu datih podataka



NENADGLEDANO UČENJE

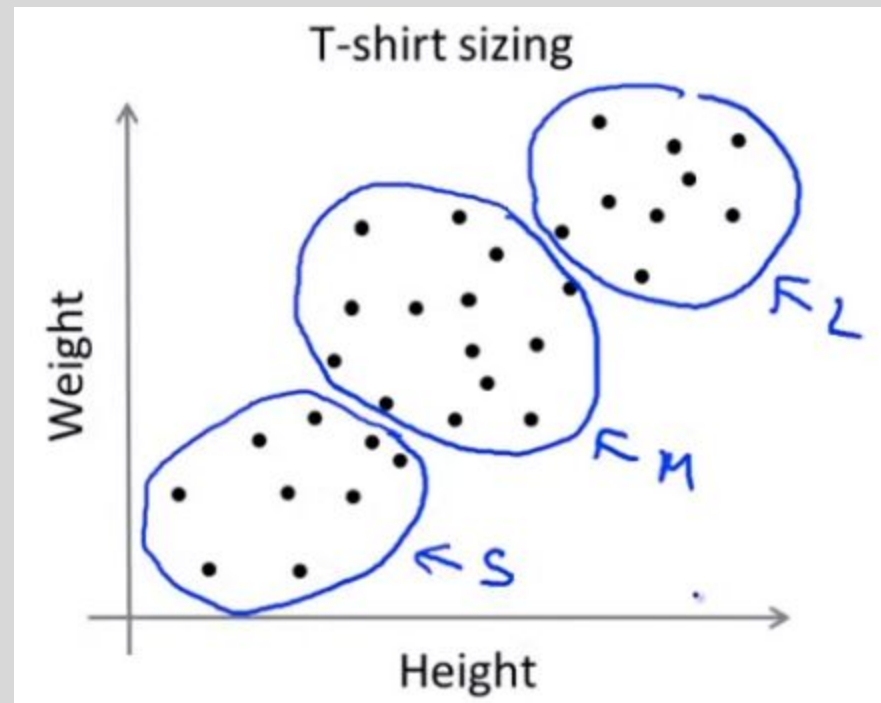
Kod nenadgledanog učenja

- nemamo informacija o željenoj izlaznoj vrednosti
- program dobija samo skup ulaznih podataka (x_1, x_2, \dots, x_n)

Zadatak programa je da otkrije paterne tj. skrivene strukture/zakovitosti u podacima

NENADGLEDANO UČENJE

Primer: određivanje konfekcijskih veličina na osnovu visine i težine ljudi



UČENJE UZ PODSTICAJE

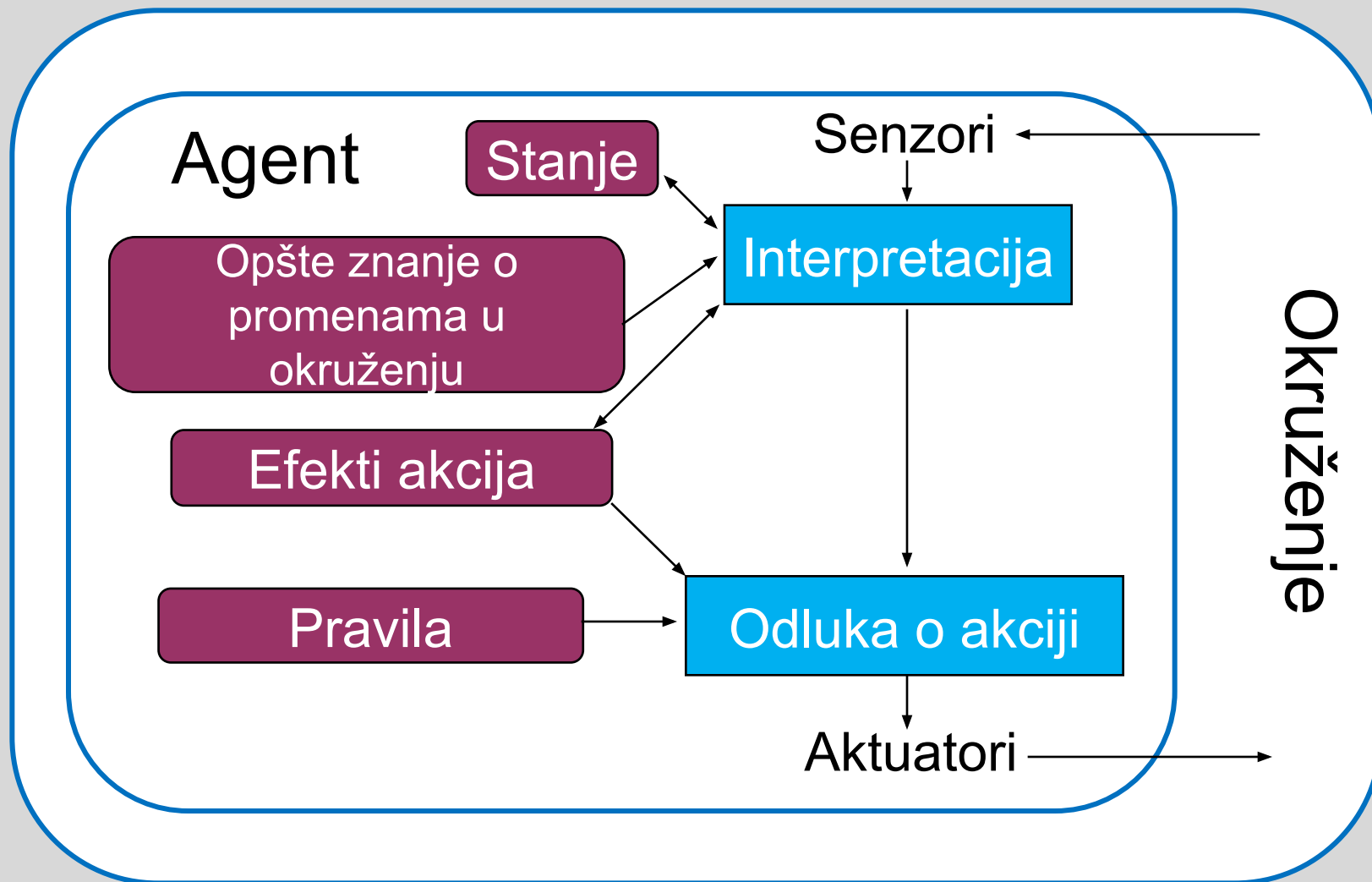
Ovaj oblik učenja podrazumeva da program (agent) deluje na okruženje izvršavanjem niza akcija

Ove akcije utiču na stanje okruženja, koje povratno utiče na agenta pružajući mu povratne informacije koje mogu biti “nagrade” ili “kazne”

Cilj agenta je da nauči kako da deluje u datom okruženju tako da vremenom max. nagrade (ili min. kazne)

Primer: autonomna vozila

ILUSTRACIJA AGENTA KOJI UČI UZ PODSTICAJE



1.1. OSNOVNI KORACI I ELEMENTI PROCESA MAŠINSKOG UČENJA

OSNOVNI KORACI PROCESA M. UČENJA

- 1)** *Prikupljanje podataka* potrebnih za formiranje dataset-ova za obuku, (validaciju) i testiranje modela m. učenja
- 2)** Priprema podataka, što tipično podrazumeva “čišćenje” i *transformaciju podataka*
- 3)** Analiza rezultujućih dataset-ova, i njihovo, eventualno, dalje unapređenje kroz *selekciju/transformaciju atributa*
- 4)** *Izbor 1 ili više algoritama* m. učenja
- 5)** *Obuka, konfiguracija i evaluacija* kreiranih modela
- 6)** Izbor modela koji će se koristiti (na osnovu rezultata koraka 5) i njegovo *testiranje*

PODACI

Podaci su potrebni za trening, validaciju i testiranje modela

- Tipična podela podataka kojima raspolažemo je 60% za trening, 20% za validaciju i 20% za testiranje
- Izbor uzoraka za trening, validaciju i testiranje treba da se uradi na slučajan način (random selection)

Za nadgledano učenje, moramo imati “obeležene” podatke

- Npr. obeležiti slike koje sadrže lice, elektronsku poštu koja je nepoželjna, e-mail adrese koje su lažne, i sl.

PODACI

Izvori podataka:

- Javno dostupne kolekcije podataka
 - Pogledati npr. <https://www.kaggle.com/datasets>
- Podaci dostupni posredstvom Web API-a
 - Pogledati npr. <https://developer.twitter.com/en/docs/twitter-api>
- Sve veće tržište gde je moguće kupiti podatke
 - Pogledati npr. <https://about.datarade.ai/>

PODACI

Preporuka: predavanje Peter Norvig*-a na temu značaja podataka za mašinsko učenje:

The Unreasonable Effectiveness of Data

URL: <http://www.youtube.com/watch?v=yvDCzhbjYWs>

*Peter Norvig je autor jedne od najpoznatijih knjiga u domenu Veštačke inteligencije i trenutno na poziciji Director of Research u Google-u

ATRIBUTI (FEATURES)

Osnovna ideja:

- pojave/entitete prepoznavamo uočavajući njihove osobine (ili izostanak nekih osobina) i uviđajući odnose između različitih osobina
- omogućiti programu da koristi osobine pojava/entiteta za potrebe njihove identifikacije/grupisanja

Izazov:

- odabrati attribute koji najbolje opisuju neki entitet/pojavu, tj. omogućuju distinkciju entiteta/pojava različitog tipa

ATRIBUTI (FEATURES)

Primeri:

- Za elektronsku poštu: naslov (tj. polje subject), reči napisane velikim slovom, dužina email-a, prva reč i sl.
- Za stan: površina, lokacija, broj soba, tip grejanja i sl.
- Za tweet poruke: prisustvo linkova, prisustvo hashtag-ova, vreme slanja, pošiljalac, ...

ODABIR ALGORITMA

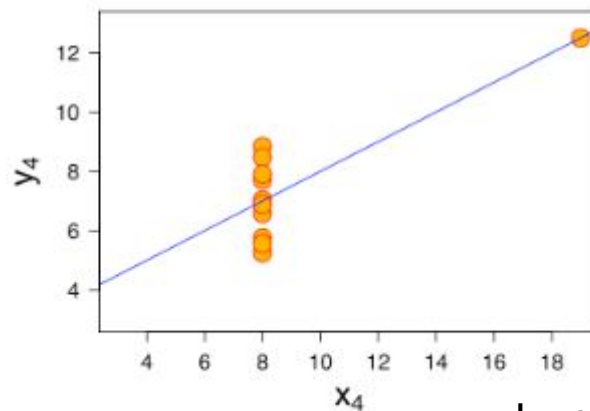
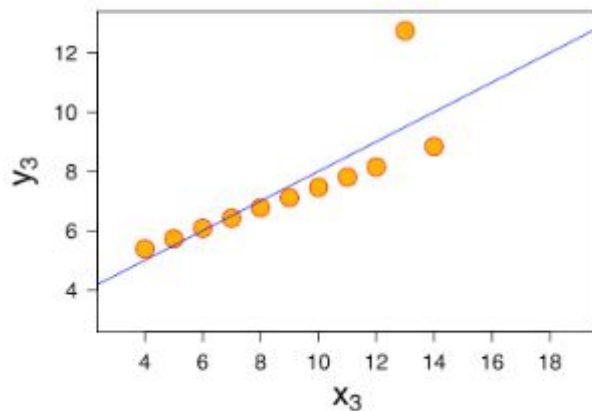
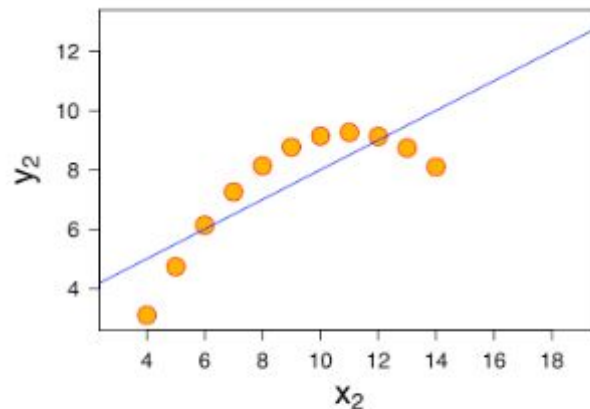
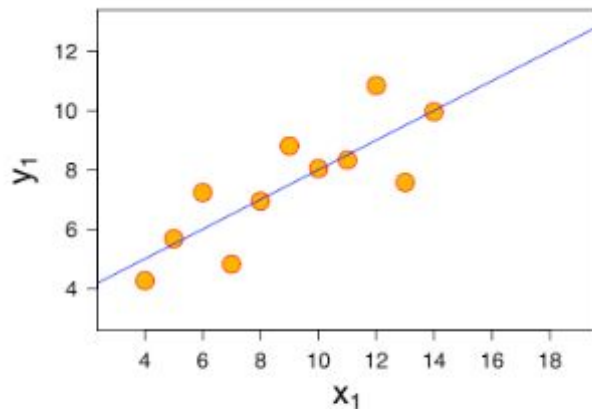
Generalno, zavisi od:

- vrste problema koji rešavamo,
- karakteristika skupa atributa (features)
 - tip atributa i stepen homogenosti tipova i opsega vrednosti atributa
 - stepen međuzavisnosti (korelisanosti) atributa
- obima podataka koji su nam na raspolaganju

ODABIR ALGORITMA

Primer:

pokušaj aproksimacije četiri različita skupa podataka primenom iste linearne funkcije (tj. linearne regresije)



ODABIR ALGORITMA

Primer (nastavak):

Očigledno, linearna funkcija neće biti najbolje rešenje za sve skupove podataka; za mnoge će nam biti potrebna neka složenija funkcija (npr. polinomijalna funkcija).

U zavisnosti od odabira kompleksnosti funkcije, imaćemo bolju ili lošiju aproksimaciju podataka.

TESTIRANJE

Pošto se algoritam obučavao na podacima za trening, njegova uspešnost na tim podacima nije indikator stvarne uspešnosti

Za procenu uspešnosti modela, potrebni su podaci koje model nije imao prilike da “vidi” u fazi učenja

Reč je o podacima za testiranje, za koje se obično izdvaja 20-30% ukupnih podataka

Uspešnost modela se utvrđuje različitim metrikama: tačnost, preciznost, odziv, ...

TRAIN/VALIDATE/TEST

Pored treniranja i testiranja modela, najčešće se radi i validacija modela kako bi se:

- a) izabrao najbolji algoritam između više kandidata
- b) odredila optimalna konfiguracija parametara modela
- c) izbegli problemi *over/under-fitting-a*

U ovim slučajevima, ukupan dataset deli se u odnosu 60/20/20 na podatke za trening, validaciju i testiranje

Podaci za validaciju koriste se za poređenje performansi

- različitih algoritama (tačka (a) iznad)
- izabranog algoritma sa različitim vrednostima parametara (tačka (b) iznad)

UNAKRSNA VALIDACIJA (*CROSS-VALIDATION*)

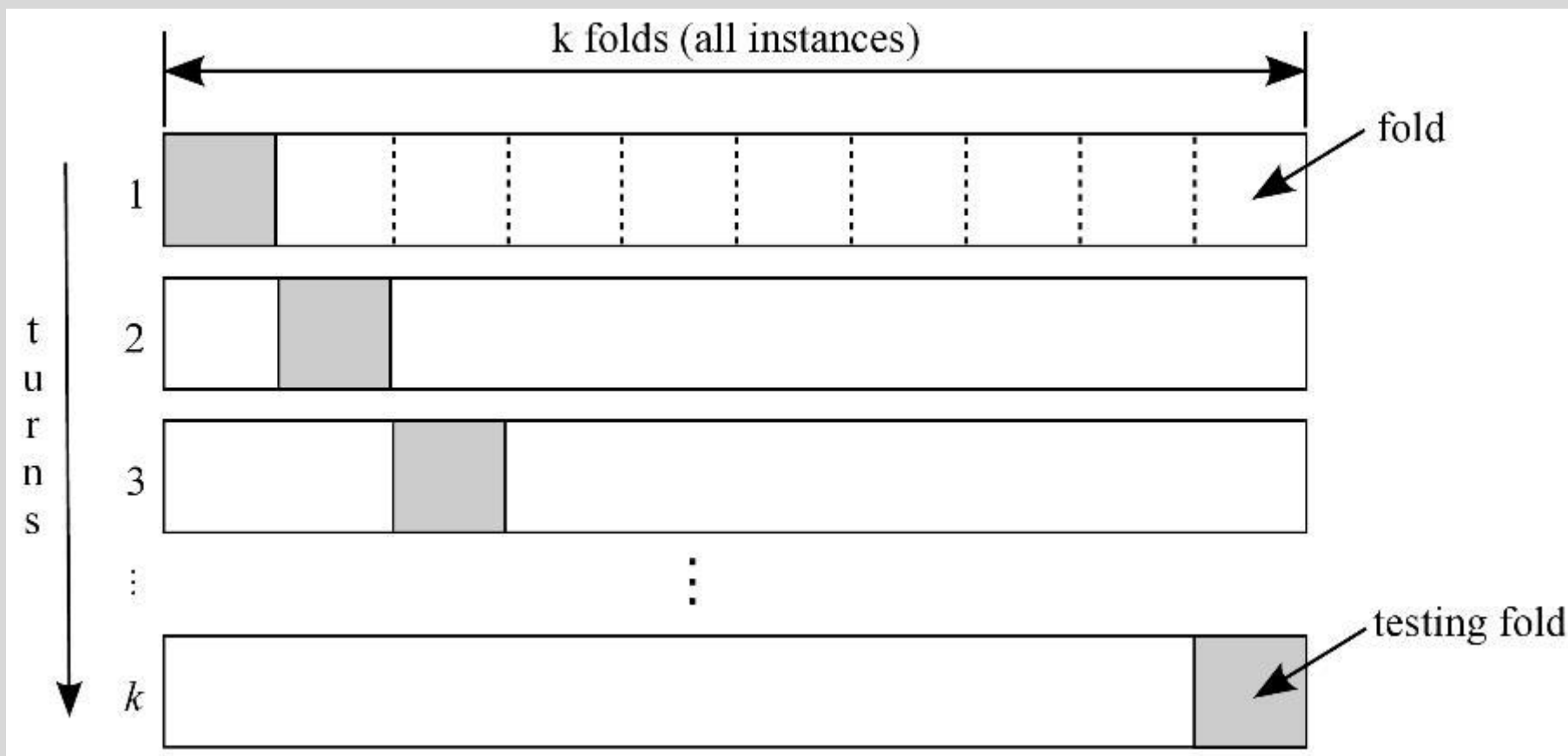
Čest pristup za efikasno korišćenje raspoloživih podataka

Kako funkcioniše:

- raspoloživi skup podataka za trening se podeli na K delova ili podskupova (*folds*)
 - najčešće se uzima 10 podskupova (*10 fold cross validation*)
- zatim se obavlja K iteracija treninga + validacije modela ; u svakoj iteraciji:
 - uzima se 1 deo podataka za potrebe validacije, a ostatak (K-1 deo) se koristi za učenje
 - bira se uvek različiti podskup koji će se koristiti za validaciju

Sledeći slajd ilustruje postupak

UNAKRSNA VALIDACIJA (*CROSS-VALIDATION*)



UNAKRSNA VALIDACIJA (*CROSS-VALIDATION*)

Pri svakoj iteraciji računaju se performanse modela (npr. mera tačnosti modela)

Na kraju se računa prosečna uspešnost na nivou svih K iteracija – tako izračunate mere uspešnosti daju bolju sliku o performansama modela

Ukoliko su rezultati u svih K iteracija vrlo slični, smatra se da je procena uspešnosti modela pouzdana

ANALIZA GREŠKE

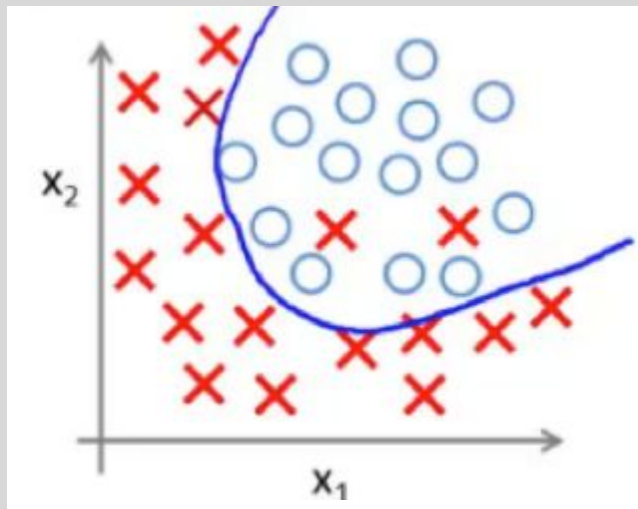
Podrazumeva “ručno” pregledanje primera na kojima je model pravio greške i uočavanje paterna u tim primerima

Pomaže da se stekne osećaj zbog čega model greši, i šta bi se moglo uraditi da se greške otklone; Npr.

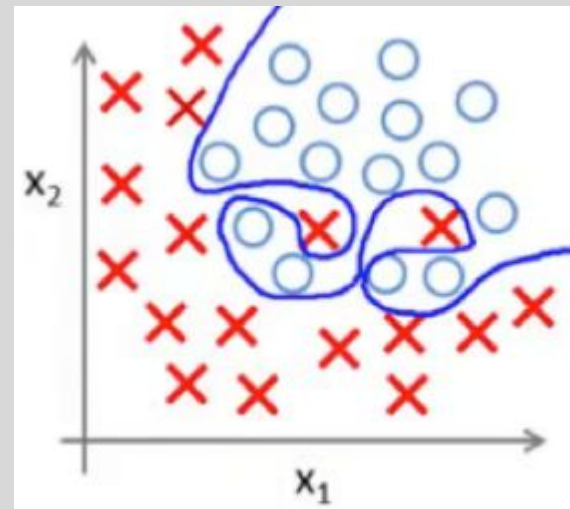
- identifikovati suvišne attribute
- identifikovati attribute koji nedostaju
- drugačije podesiti parametre modela
- ...

PROBLEM PREVELIKOG PODUDARANJA SA PODACIMA (OVER-FITTING)

Odnosi na situaciju u kojoj model savršeno nauči da prepozna instance iz trening seta, ali nije u mogućnosti da prepozna instance koje se i malo razlikuju od naučenih



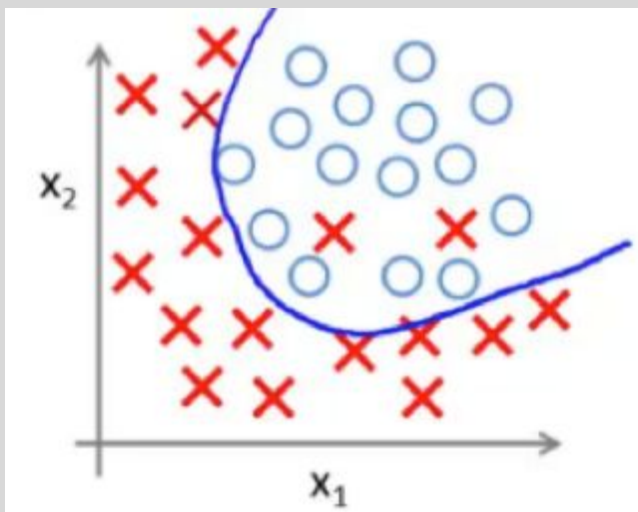
poželjno rešenje



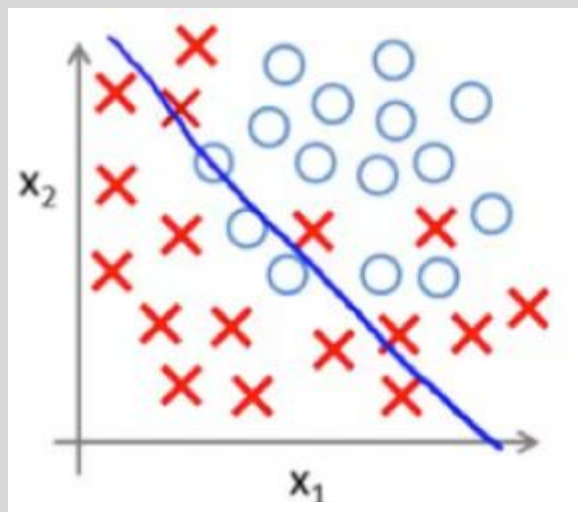
over fitting

PROBLEM NEDOVOLJNOG PODUDARANJA SA PODACIMA (UNDER-FITTING)

Under-fitting se odnosi na slučaj kad model ne uspeva da aproksimira podatke za trening, tako da ima slabe performanse čak i na trening setu

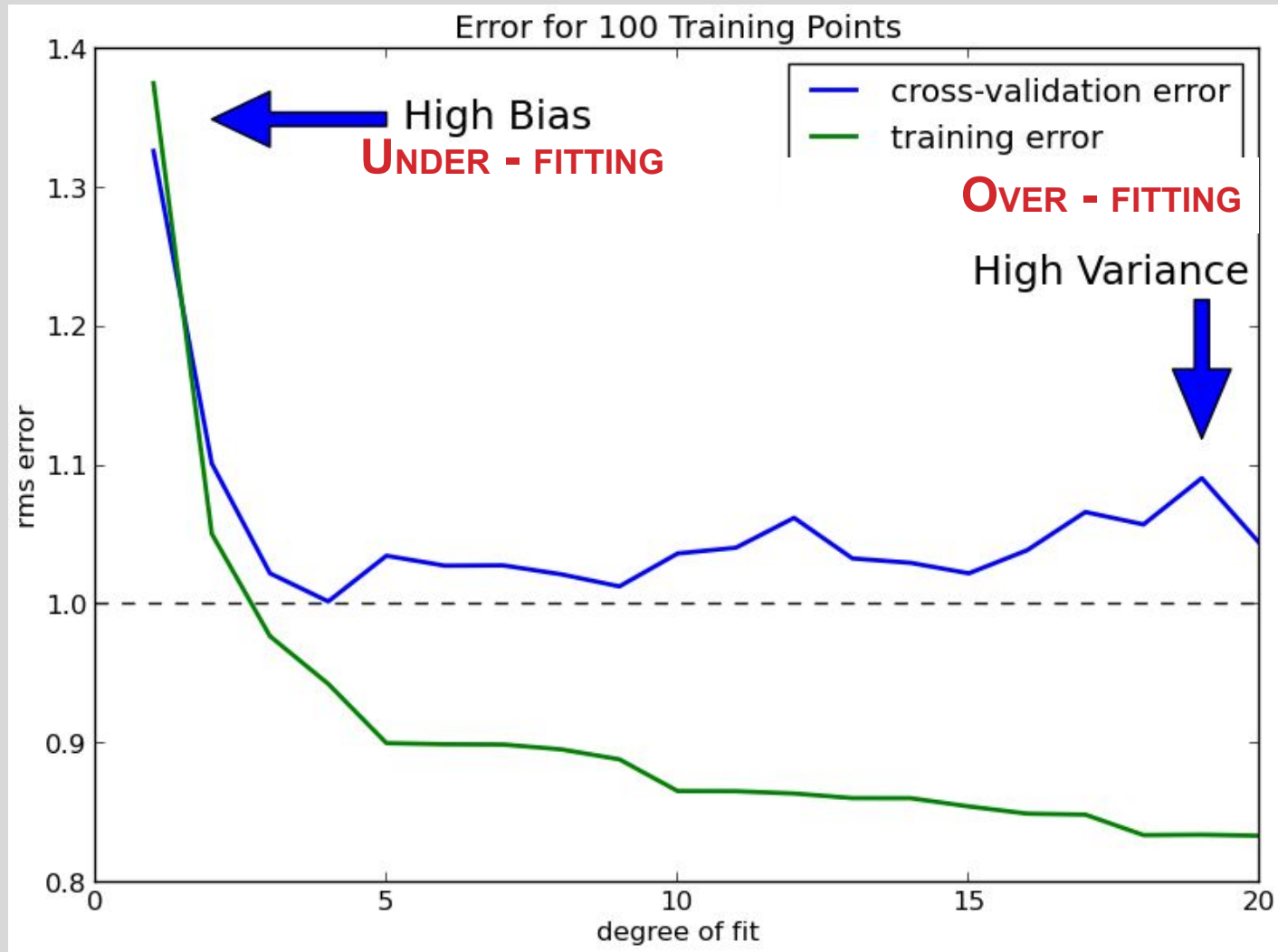


poželjno rešenje



under fitting

OVER-FITTING VS. UNDER-FITTING



2.

KLASIFIKACIJA

PREGLED TEMA

- Šta je klasifikacija?
- Binarna i više-klasna klasifikacija
- Algoritmi klasifikacije
- Naive Bayes (NB) algoritam
- Ilustracija NB algoritma: klasifikacija teksta
- Mere uspešnosti klasifikatora

ŠTA JE KLASIFIKACIJA?

- Zadatak određivanja klase kojoj neka instanca pripada
 - instanca je opisana vrednošću atributa
 - skup mogućih klasa je poznat i dat
- Klase su date kao nominalne vrednosti, npr.
 - klasifikacija email poruka: spam, non-spam
 - klasifikacija novinskih članaka: politika, sport, kultura i sl.

BINARNA I VIŠE-KLASNA KLASIFIKACIJA

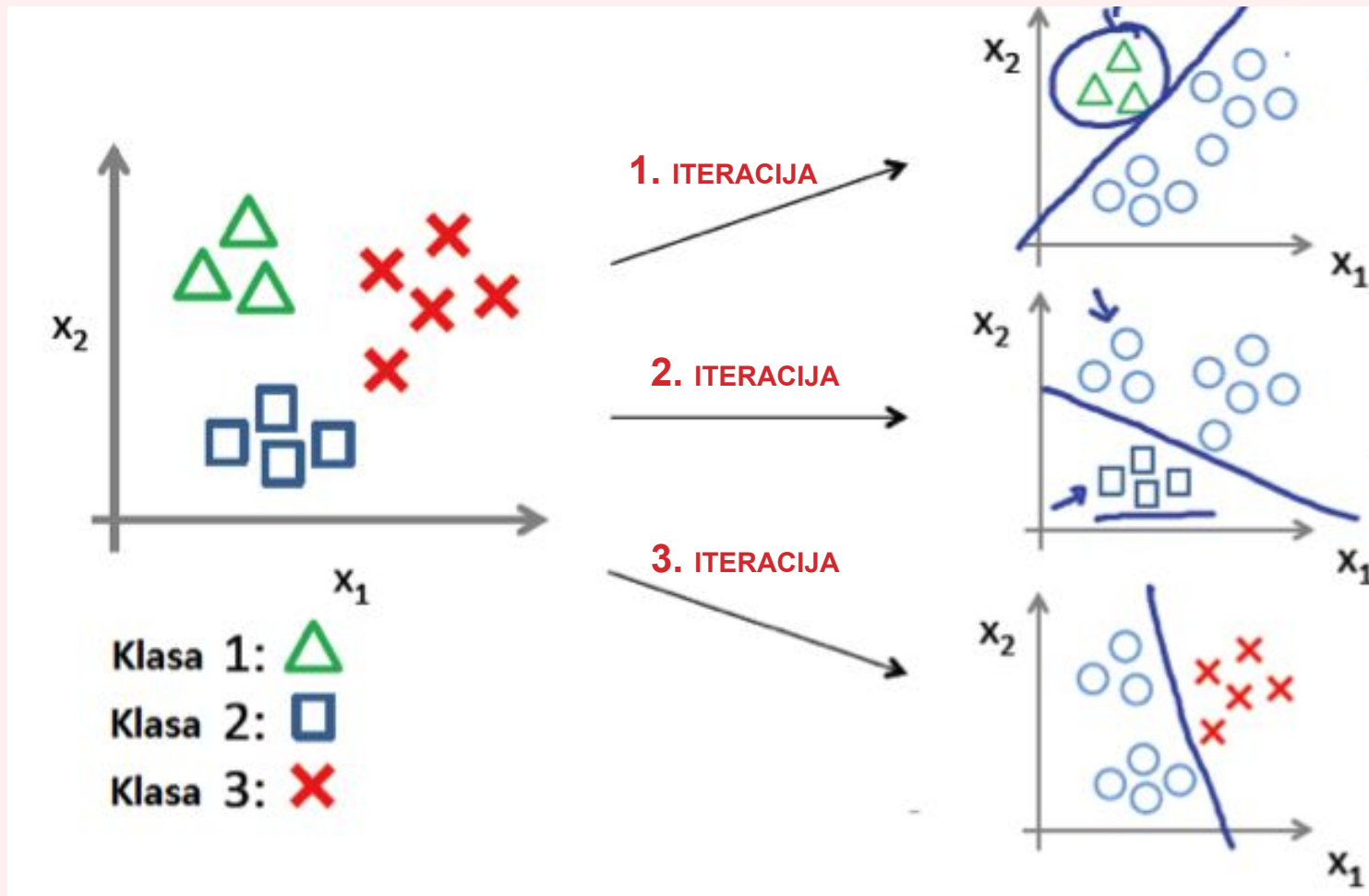
Zavisno od broja klasa, razlikujemo:

- *binarnu* klasifikaciju - postoje dve klase
- *više-klasnu* klasifikaciju - postoji više klasa u koje se instance mogu svrstati

Princip rada algoritma u oba slučaja je gotovo isti:

u slučaju postojanja više klasa, algoritam iterativno uči, tako da u svakoj iteraciji “nauči” da jednu od klasa razgraniči od svih ostalih

VIŠE-KLASNA KLASIFIKACIJA



ALGORITMI KLASIFIKACIJE

Postoje brojni pristupi/algoritmi za klasifikaciju:

- Logistička regresija
- Naive Bayes
- Algoritmi iz grupe Stabala odlučivanja
- Algoritmi iz grupe Neuronskih mreža
- k-Nearest Neighbor (kNN)
- Support Vector Machines (SVM)
- ...

2.1. NAIVE BAYES

ZAŠTO BAŠ NAIVE BAYES?

Naive Bayes (NB) se navodi kao algoritam koji treba među prvima razmotriti pri rešavanju zadataka klasifikacije

Razlozi:

- Jednostavan je
- Ima dobre performanse
- Vrlo je skalabilan
- Može se prilagoditi za gotovo bilo koji problem klasifikacije

PODSEĆANJE: BAYES-OVO PRAVILO

$$P(H|E) = P(E|H) * P(H) / P(E)$$

- H – hipoteza (*hypothesis*)
- E – opažaj (*evidence*) vezan za hipotezu H , tj. podaci na osnovu kojih bi trebalo da potvrdimo ili odbacimo hipotezu H
- $P(H)$ – verovatnoća hipoteze H (*prior probability*)
- $P(E)$ – verovatnoća opažaja tj. stanja na koje ukazuju prikupljeni podaci
- $P(E|H)$ – (uslovna) verovatnoća opažaja E ukoliko važi hipoteza H
- $P(H|E)$ – (uslovna) verovatnoća hipoteze H ukoliko imamo opažaj E

BAYES-OVO PRAVILO - PRIMER

Pretpostavite sledeće:

- jednog jutra ste se probudili sa povišenom temperaturom
- prethodnog dana ste čuli da je u gradu počela da se širi virusna infekcija, ali da je verovatnoća zaraze mala, svega 2.5%
- takođe ste čuli da je u 50% slučajeva virusna infekcija praćena povišenom temperaturom
- u vašem slučaju, povišena temperatura se javlja svega par puta u godini, tako da je verovatnoća da imate povišenu temp. 6.5%

Pitanje: kolika je verovatnoća da, pošto imate povišenu temp., da imate i virusnu infekciju?

BAYES-OVO PRAVILO - PRIMER

| Teorija | Primer |
|---|---|
| Hipoteza (H) | Imate virusnu infekciju |
| $P(H)$ | 0.025 |
| Opažaj (evidence - E) | Imate povišenu temperaturu |
| $P(E)$ | 0.065 |
| (uslovna) verovatnoća opažaja E ukoliko važi hipoteza H: $P(E H)$ | Verovatnoća da je virusna infekcija praćena povišenom temperaturom 0.50 |
| (uslovna) verovatnoća hipoteze H ukoliko imamo opažaj E: $P(H E)$ | Verovatnoća da pošto imate povišenu temp., da imate i virusnu infekciju ? |

$$P(H|E) = P(E|H) * P(H) / P(E)$$

$$P(H|E) = 0.50 * 0.025 / 0.065 = 0.19$$

NAIVE BAYES U KLASIFIKACIJI TEKSTA

NB je jedan od najčešće korišćenih algoritama za klasifikaciju teksta

Zadatak klasifikacije teksta: odrediti kojoj klasi (c) iz datog skupa klasa (C), dati tekst pripada

Na primer:

- tematska klasifikacija novinskih članaka
- klasifikacija tweet poruka prema iskazanom stavu (poz./neg.)

FORMIRANJE VEKTORA ATRIBUTA

Tekst koji je predmet klasifikacije se tretira kao prost skup reči (tzv. bag-of-words)

- Reči iz teksta su osnova za kreiranje atributa (features) sa kojima će NB algoritam raditi
- Postoji više načina da se reči iz teksta upotrebe za definisanje vektora atributa (feature vector) za klasifikaciju

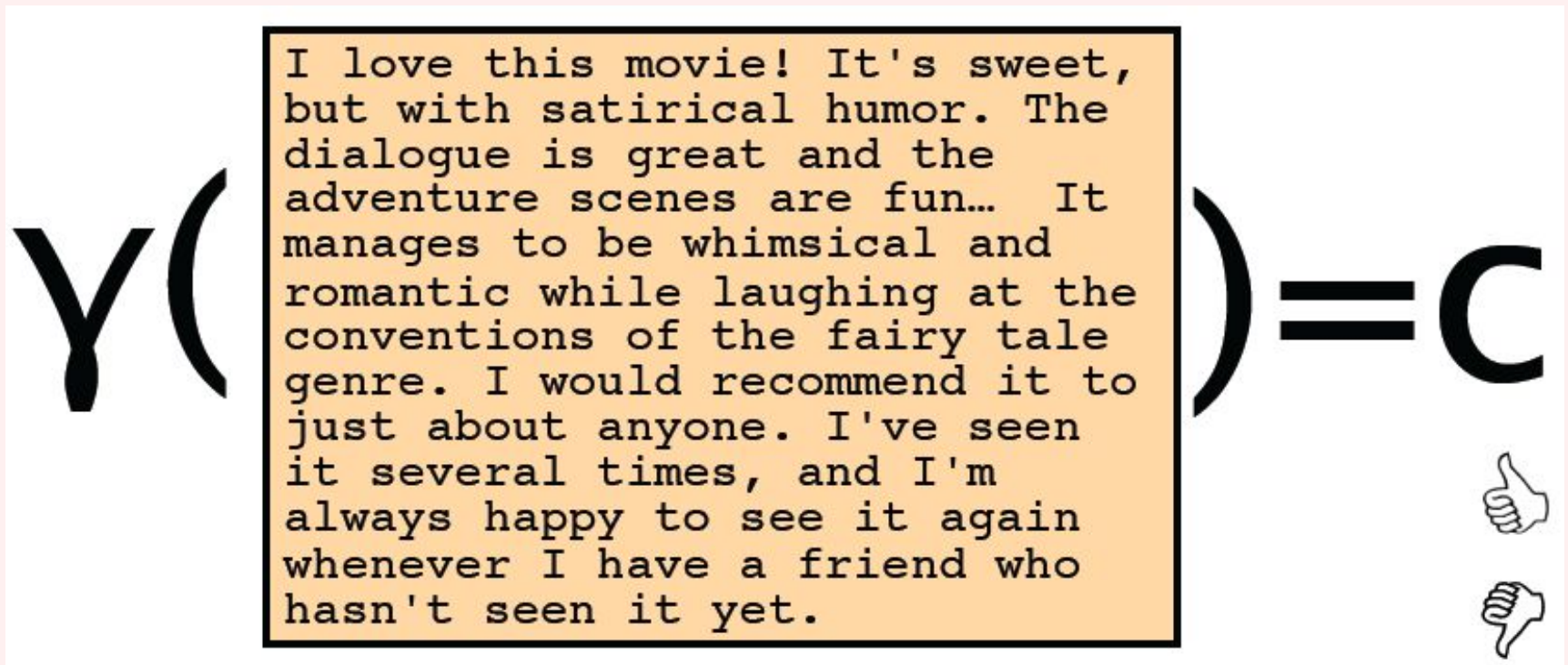
FORMIRANJE VEKTORA ATRIBUTA

Pristup koji se često primenjuje:

- Estrahovati reči iz dokumenata koji čine skup za trening \mathbf{D} , i formirati rečnik \mathbf{R} koji sadrži jedinstvena pojavljivanja reči iz skupa \mathbf{D}
- Rečnik \mathbf{R} se može kreirati na osnovu svih reči iz skupa \mathbf{D} ili samo na osnovu onih reči koje mogu biti značajne za dati zadatak klasifikacije
- Zatim se svaki document \mathbf{d} iz skupa \mathbf{D} predstavlja kao vektor učestanosti pojavljivanja reči iz rečnika \mathbf{R}
- Drugim rečima, reči iz rečnika su atributi koji se koriste u klasifikaciji, a njihove vrednosti su učestanosti pojavljivanja u pojedinačnim dokumentima

FORMIRANJE VEKTORA ATRIBUTA - PRIMER

Klasifikacija komentara (*reviews*) filmova; dokumenti su ovde pojedinačni komentari (*reviews*); svaki komentar je označen kao pozitivan ili negativan



FORMIRANJE VEKTORA ATRIBUTA - PRIMER

Ilustracija predstavljanja dokumenta (*review*) preko formiranog rečnika, odnosno skupa atributa (x_i), i vrednosti atributa za dati dokument (*review*)

| | | | |
|----|--------------------------|---------------------|-----|
| Y(| atribut (x_i) | vrednost (TF_i) |)=C |
| | $x_1 = \text{love}$ | $TF_1 = 1$ | |
| | $x_2 = \text{sweet}$ | $TF_2 = 1$ | |
| | $x_3 = \text{satirical}$ | $TF_3 = 1$ | |
| | ... | | |
| | $x_{11} = \text{happy}$ | $TF_{11} = 2$ | |
| | $x_{12} = \text{laugh}$ | $TF_{12} = 2$ | |
| | ... | | |



TF= Term Frequency

NB U KLASIFIKACIJI TEKSTA

Ako je ***c*** klasa, a ***d*** dokument, verovatnoća da je upravo ***c*** klasa dokumenta ***d*** biće:

$$P(c|d) = P(d|c) * P(c) / P(d) \quad (1)$$

Za dati skup klasa ***C*** i dokument ***d***, želimo da pronađemo onu klasu ***c*** iz skupa ***C*** koja ima najveću uslovnu verovatnoću za dokument ***d***, što daje sledeću funkciju:

$$f = \operatorname{argmax}_{c \text{ iz } C} P(c|d) \quad (2)$$

Primenom Bayes-ovog pravila, dobijamo:

$$f = \operatorname{argmax}_{c \text{ iz } C} P(d|c) * P(c) \quad (3)$$

NB U KLASIFIKACIJI TEKSTA

$$f = \operatorname{argmax}_{c \text{ iz } C} P(d|c) * P(c) \quad (3)$$

Potrebno je odrediti verovatnoće $P(c)$ i $P(d|c)$

$P(c)$ se može *proceniti* relativno jednostavno: brojanjem pojavljivanja klase c u skupu dokumenata za trening D

$P(d|c)$ - verovatnoća da u klasi c zateknemo dokument d – nije tako jednostavno odrediti i tu uvodimo pretpostavke koje NB algoritam čine “naivnim”

NB U KLASIFIKACIJI TEKSTA

Kako odrediti $P(d|c)$?

- dokument ***d*** predstavljamo kao skup atributa (x_1, x_2, \dots, x_n)
- umesto $P(d|c)$ imaćemo $P(x_1, x_2, x_3, \dots, x_n|c)$
- da bi izračunali $P(x_1, x_2, x_3, \dots, x_n|c)$ uvodimo 2 naivne pretpostavke:
 - dokument ***d*** posmatramo kao prost skup reči (bag-of-words); tj. pozicija i redosled reči u tekstu se smatraju nevažnim
 - pojavljivanje određene reči u datoj klasi ***c*** je nezavisno od pojavljivanja neke druge reči u toj klasi

NB U KLASIFIKACIJI TEKSTA

Uvedene pretpostavke

- dovode do značajnog gubitka informacija koje iz podataka možemo da izvučemo, ali,
- omogućuju značajno jednostavnije računanje $P(x_1, x_2, \dots, x_n | c)$, a time i ceo problem klasifikacije

NB U KLASIFIKACIJI TEKSTA

Na osnovu uvedenih pretpostavki, $P(x_1, x_2, \dots, x_n | c)$ možemo da predstavimo kao proizvod individualnih uslovnih verovatnoća

$$P(x_1, x_2, \dots, x_n | c) = P(x_1 | c) * P(x_2 | c) * \dots * P(x_n | c)$$

Time dolazimo do opšte jednačine NB algoritma:

$$f = \operatorname{argmax}_{c \text{ iz } C} P(c) * \prod_{i=1, n} P(x_i | c)$$

NB U KLASIFIKACIJI TEKSTA

Procena verovatnoća se vrši na osnovu skupa za trening, i zasniva na sledećim jednačinama:

$P(c) = \text{br. dok. klase } \mathbf{c} / \text{ukupan br. dok. u skupu za trening}$

$P(x_i | c) = \text{br. pojavljivanja reči } \mathbf{r}_i \text{ u dok. klase } \mathbf{c} \text{ (TF)} /$
 $\text{ukupan br. reči iz rečnika } \mathbf{R} \text{ u dok. klase } \mathbf{c}$

OSOBI NE NB ALGORITMA

- Veoma brz i efikasan
- Najčešće daje dobre rezultate
 - često se pokazuje kao bolji ili bar podjednako dobar kao drugi, sofisticiraniji modeli
- Nije memorijski zahtevan
- Ima vrlo mali afinitet ka preteranom podudaranju sa podacima za trening (overfitting)
- Pogodan kada imamo malu količinu podataka za trening

OSOBI NE NB ALGORITMA


- “Otporan” na nevažne attribute
 - atributi koji su podjednako distribuirani kroz skup podataka za trening, pa nemaju veći uticaj na izbor klase
- Namenjen primarno za rad sa nominalnim atributima; u slučaju numeričkih atributa:
 - koristiti raspodelu verovatnoća atributa (tipično Normalna raspodela) za procenu verovatnoće svake od vrednosti atributa
 - uraditi diskretizaciju vrednosti atributa

2.2. STABLA ODLUČIVANJA

PRIMER: KLASIFIKACIJA IGRAČA BEJZBOLA

Potrebno je klasifikovati igrače bejzbola na one koji su jako dobro plaćeni i one koji to nisu (WellPaid), na osnovu

- broja ostvarenih poena u prethodnoj godini (Hits) i
- broja godina koje je igrač proveo u glavnoj ligi (Years)

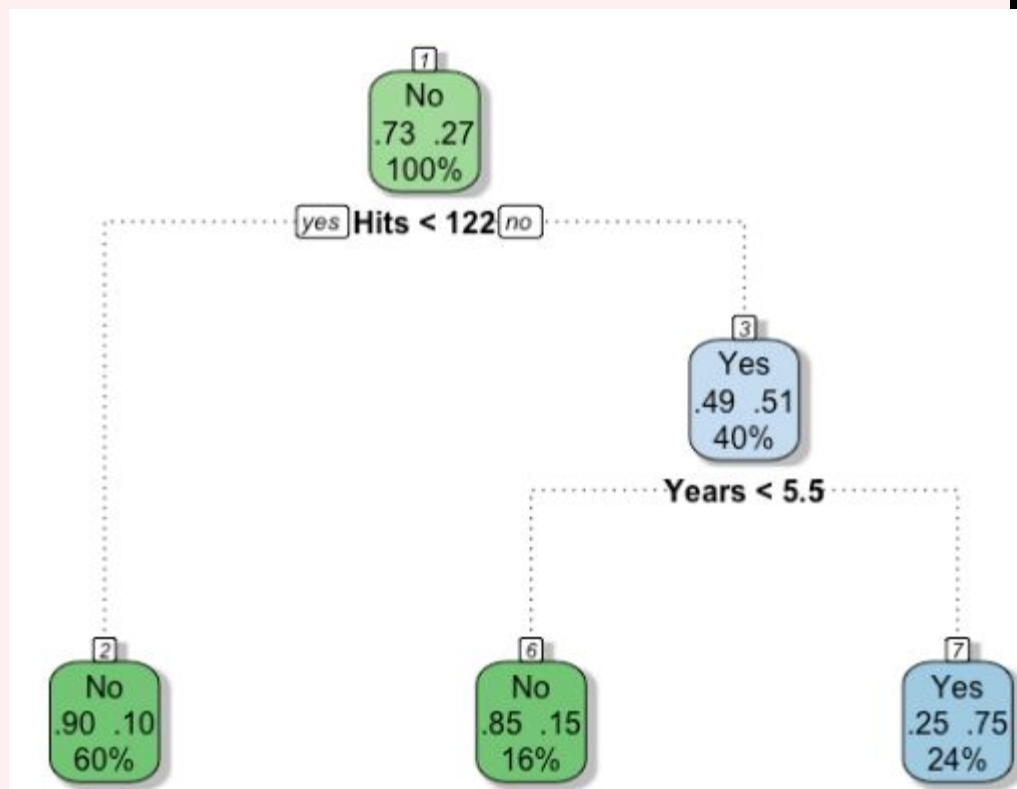
```
Console ~/R Studio Projects/Intelligent Systems Fall 2015/   
> str(hitters.subset)  
'data.frame': 263 obs. of 3 variables:  
 $ Hits : int 81 130 141 87 169 37 73 81 92 159 ...  
 $ Years : int 14 3 11 2 11 2 3 2 13 10 ...  
 $ WellPaid: Factor w/ 2 levels "No","Yes": 1 1 1 1 2 1 1 1 2 1 ...  
> head(hitters.subset)  
      Hits Years WellPaid  
-Alan Ashby      81      14      No  
-Alvin Davis    130       3      No  
-Andre Dawson   141      11      No  
-Andres Galarraga 87       2      No  
-Alfredo Griffin 169      11      Yes  
-Al Newman      37       2      No  
> |
```

PRIMER: KLASIFIKACIJA IGRAČA BEJZBOLA

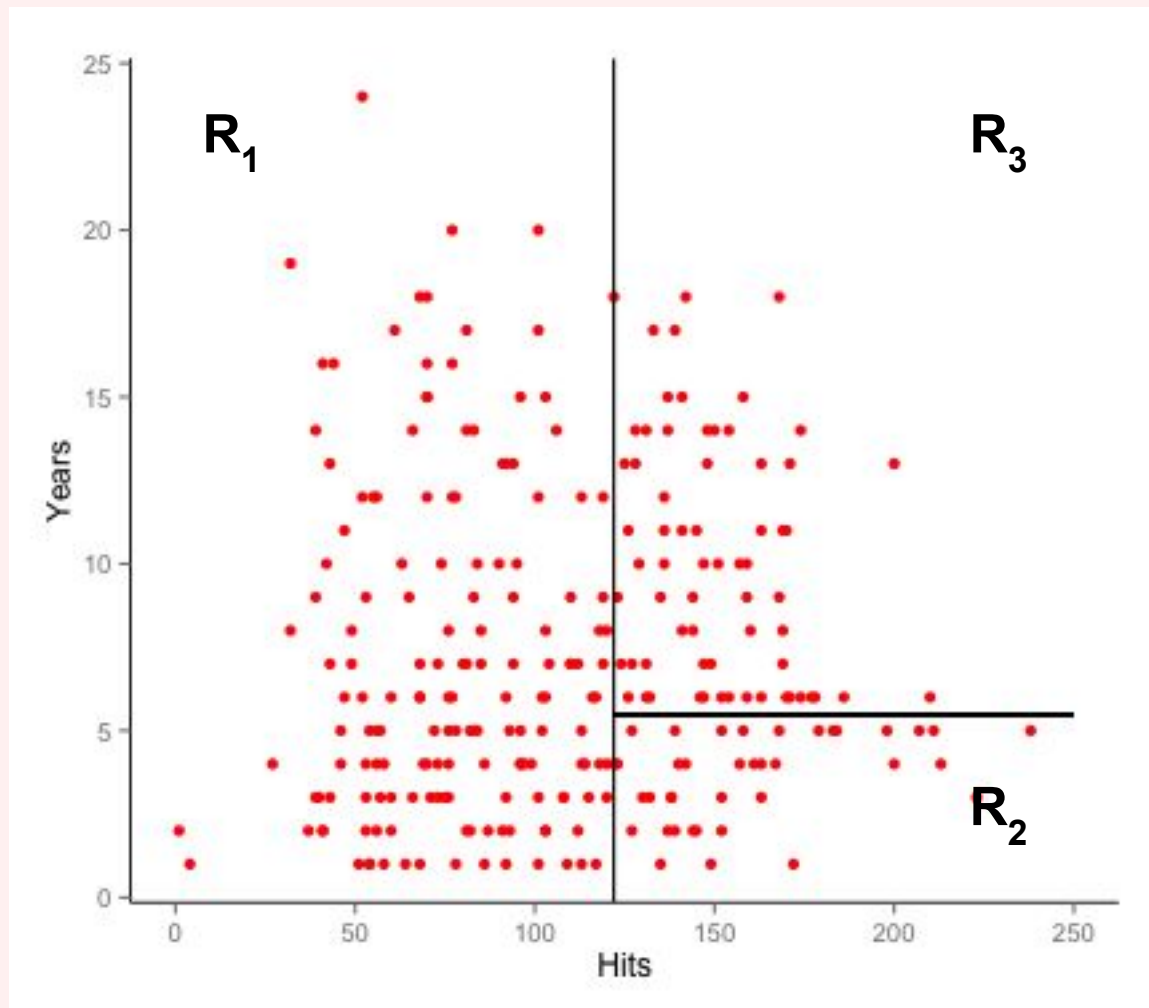
Stablo odlučivanja ukazuje da su dobro plaćeni oni igrači koji su ostvarili bar 122 pogotka u prethodnoj godini i koji bar 5.5 godina igraju u glavnoj ligi

Verovatnoća da je igrač sa opisanim karakteristikama dobro plaćen je 0.75

Ti igrači čine 24% svih igrača za koje su nam raspoloživi podaci (skup za trening)



DRUGI NAČIN ZA VIZUELIZACIJU STABLA ODLUČIVANJA...



OSNOVNA IDEJA KLASIFIKACIONIH STABALA

Podeliti prostora atributa kojima su objekti opisani u više različitih i međusobno nepreklopljenih regiona R_1, R_2, \dots, R_n

- prostor atributa je p -dimenzionalni prostor koga čine moguće vrednosti p atributa (x_1, x_2, \dots, x_p) kojima su dati objekti opisani

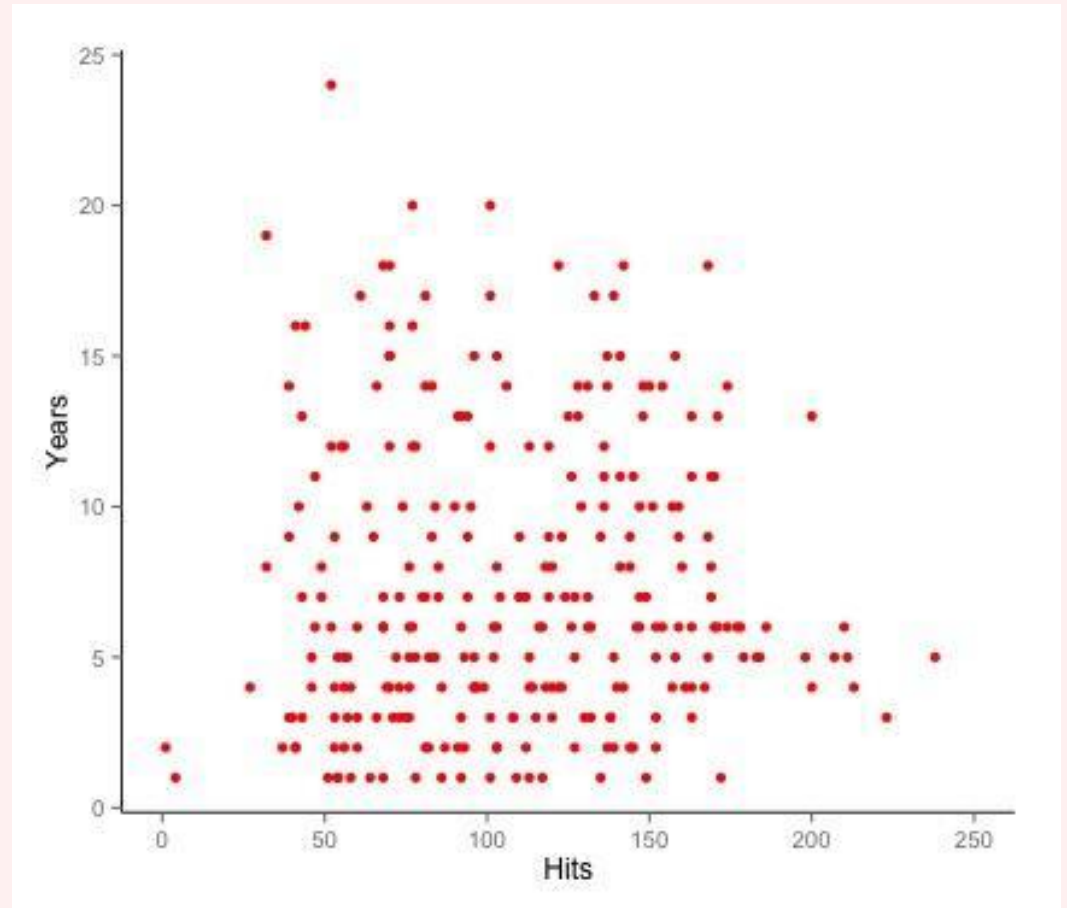
Za novi objekat X , određuje se pripadnost jednom od regiona $R_1 \dots R_n$ na osnovu vrednosti atributa (x_1, x_2, \dots, x_p) kojima je X opisan

Klasa novog objekta će biti ona klasa koja dominira (*majority class*) u regionu R_j u koji je X svrstan

PODELA PROSTORA ATRIBUTA

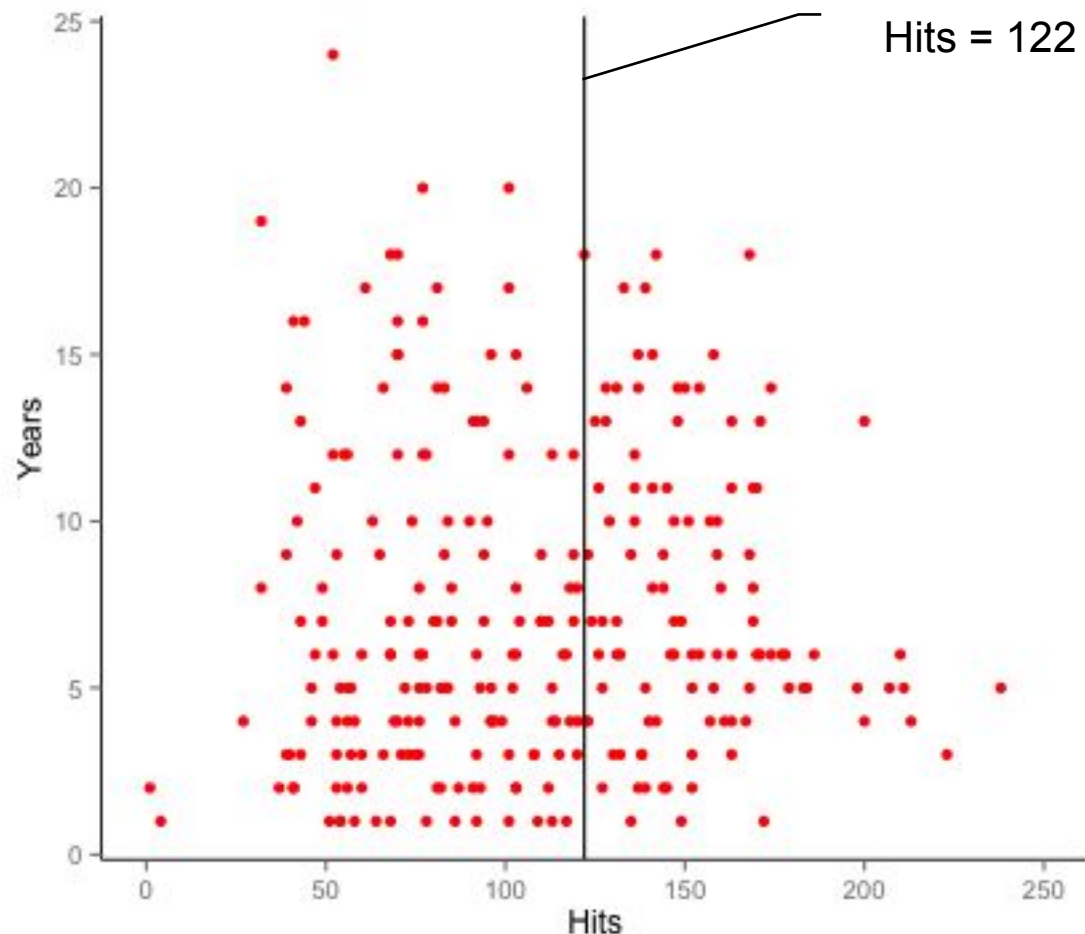
Podela prostora atributa na regione R_j je iterativni proces koji se sastoji od:

- izbora atributa x_i koji će biti osnova za podelu
- izbora vrednosti atributa x_i koja će poslužiti kao 'granična' vrednost



PODELA PROSTORA ATRIBUTA

Za prvu podelu, u datom primeru, izabran je atribut Hits, i vrednost 122

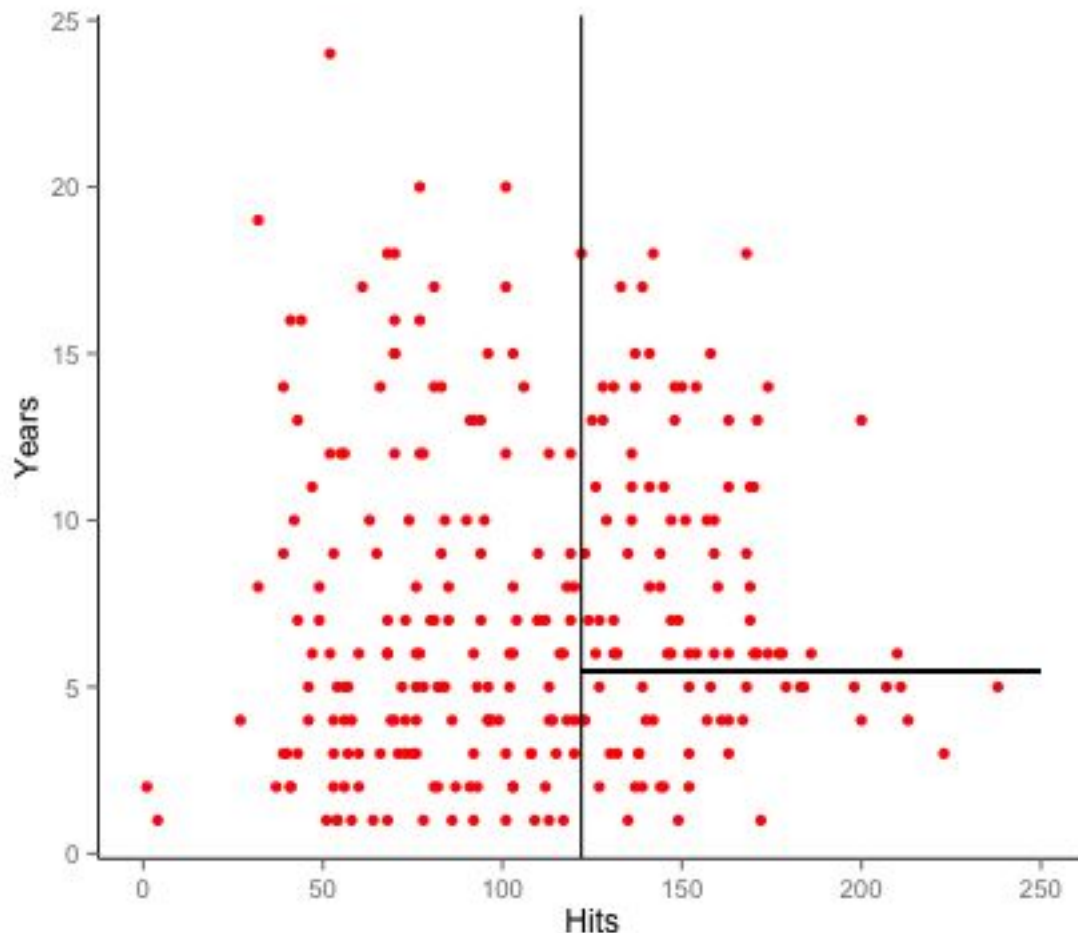


PODELA PROSTORA ATRIBUTA

Prva podela: Hits = 122

Ukoliko je Hits > 122,
sledeća podela je na
atributu Years:

Years = 5.5



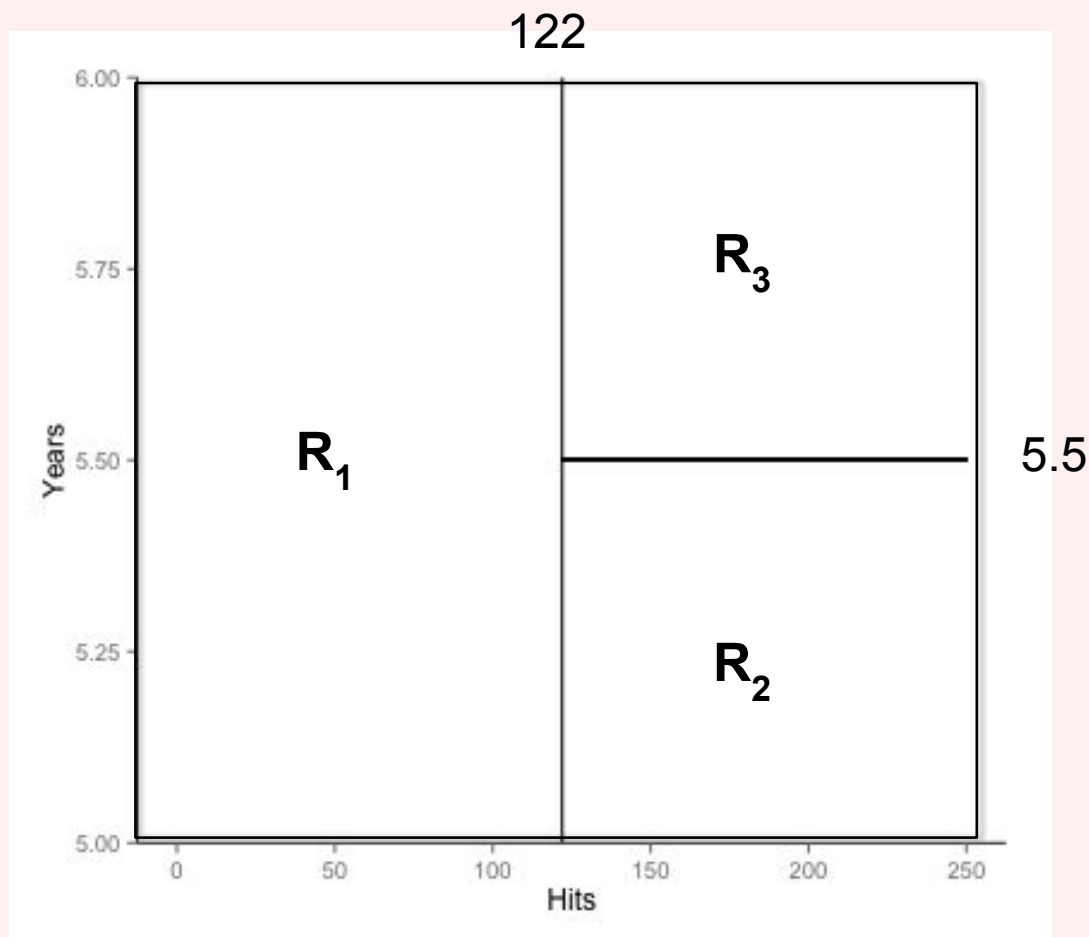
PODELA PROSTORA ATRIBUTA

Prva podela:

Hits = 122

Ako je Hits > 122,
sledeća podela:

Years = 5.5



PODELA PROSTORA ATRIBUTA

Pitanja koja se prirodno nameću:

- Kako i gde izvršiti podelu?
 - drugim rečima, kako kreiramo regione R_1, R_2, \dots, R_n ?
- Kako odrediti klasu instanci u svakom od regiona R_1, \dots, R_n ?

KAKO ODREDITI KLASU INSTANCI U REGIONIMA $R_1 \dots R_k$?

Jednostavno, koristeći princip većinske klase (*majority class*):
svakom regionu R_j , pridružiti klasu kojoj pripada većina instanci iz skupa za trening koja je svrstana u region R_j

U datom primeru, u regionu R1, 90% instanci čine igrači koji nisu visoko plaćeni => svaki novi igrač koji bude svrstan u region R1 biće klasifikovan kao igrač koji nije vrhunski plaćen



KAKO I GDE IZVRŠITI PODELU?

Cilj je pronaći regione R_1, R_2, \dots, R_n tako da se minimizuje greška pri klasifikaciji - *Classification Error Rate* (CER)

CER predstavlja proporciju instanci (iz skupa za trening) u datom regionu koje ne pripadaju dominantnoj klasi tog regiona

$$CER = 1 - \max_k \hat{p}_{ik}$$

\hat{p}_{ik} je proporcija (trening) instanci u regionu i koje pripadaju klasi k

KAKO I GDE IZVRŠITI PODELU?

Pristup koji se primenjuje da bi se identifikovali regioni koji minimizuju grešku pri klasifikaciji zasniva se na ***rekurzivnoj, binarnoj podeli*** (*recursive binary splitting*) prostora atributa

Osnovne karakteristike ovog pristupa:

- *top-down* pristup
- *greedy* pristup

REKURZIVNA, BINARNA PODELA PROSTORA ATRIBUTA

Top-down pristup

- kreće od vrha stabla, gde sve (trening) instance pripadaju jednoj (zajedničkoj) regiji, a zatim sukcesivno deli prostor atributa na regione

Greedy pristup

- pri svakom koraku, najbolja podela se određuje na osnovu stanja u tom koraku, odnosno, ne uzima se u obzir šta će biti u narednim koracima, tj koja bi to podela mogla dovesti do boljih rezultata u nekom narednom koraku

REKURZIVNA, BINARNA PODELA

Algoritam razmatra svaki atribut x_j ($j=1,p$) i svaku tačku podele s_j za taj atribut, i

bira onu kombinaciju koja će podeliti prostor atributa u dva regiona $\{X|x_j > s_j\}$ i $\{X|x_j < s_j\}$ tako da se minimizuje greška klasifikacije

KAKO I GDE IZVRŠITI PODELU?

Osim greške pri klasifikaciji (*Classification Error Rate*), kao kriterijumi za podelu prostora atributa, često se koriste i:

Gini index

Cross-entropy

GINI INDEX

Osim greške pri klasifikaciji (*Classification Error Rate*), kao kriterijumi za podelu prostora atributa, često se koriste i:

Gini index

Cross-entropy

CROSS-ENTROPY

Definiše se na sledeći način:

$$D = - \sum_{k=1}^K \hat{p}_{ik} \log \hat{p}_{ik}$$

Kao i Gini indeks, cross-entropy predstavlja meru 'čistoće' čvora (što je vrednost manja, to je čvor 'čistiji')

OREZIVANJE STABLA (*TREE PRUNING*)

- Velika klasifikaciona stabla, tj. stabla sa velikim brojem terminalnih čvorova (listova), imaju tendenciju over-fitting-a (tj. prevelikog uklapanja sa trening podacima)
- Ovaj problem se može rešiti 'orezivanjem' stabla, odnosno odsecanjem nekih terminalnih čvorova

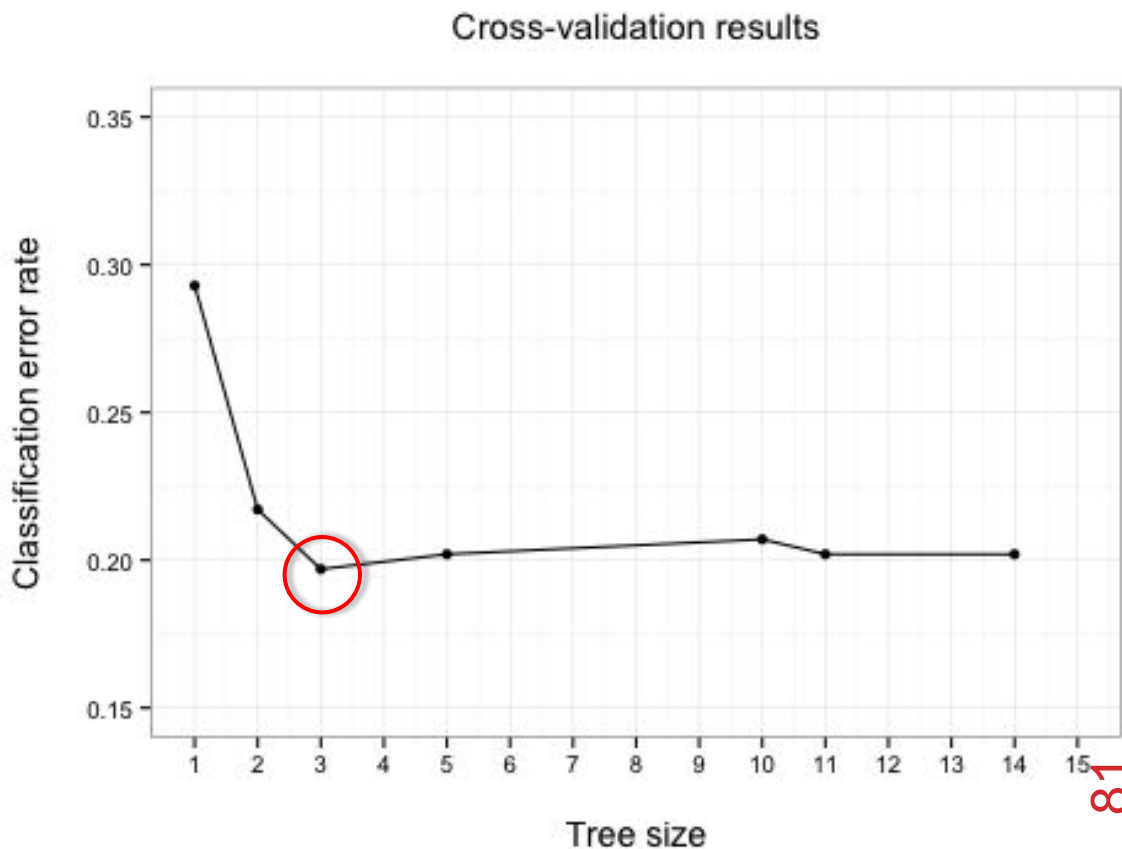
OREZIVANJE STABLA (*TREE PRUNING*)

- Kako ćemo znati na koji način i u kojoj meri treba da 'orežemo' stablo?

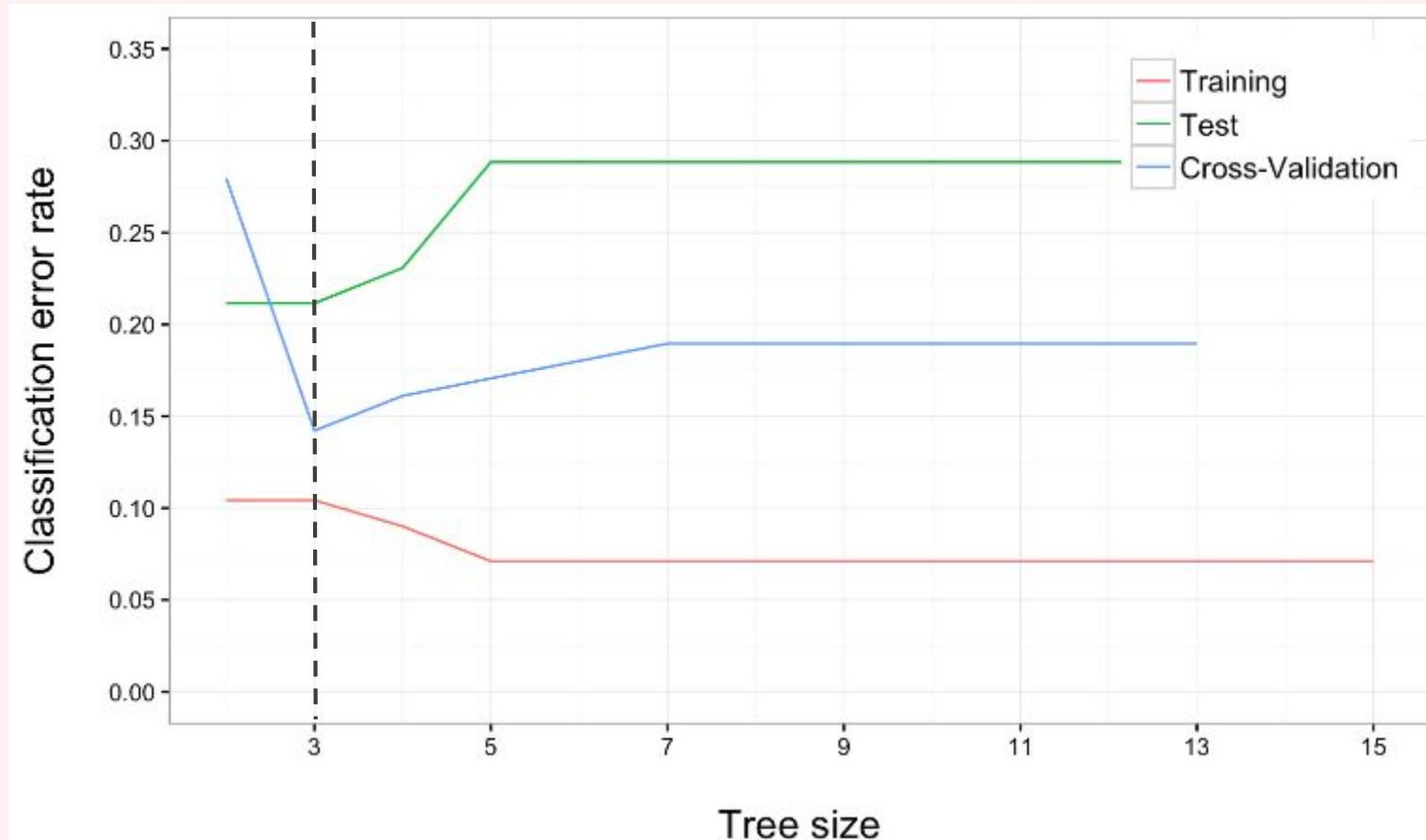
Preporuka je primenom kros validacije (*cross validation*) utvrditi grešku pri klasifikaciji za podstabla različite veličine (tj broja terminalnih čvorova) i izabrati podstablo koje daje najmanju grešku

OREZIVANJE STABLA KROZ KROS VALIDACIJU

U primeru klasifikacije igrača bejzbola, kros validacija pokazuje da se najmanja greška klasifikacije postiže u slučaju stabla veličine 3 (tj. stabla sa 3 terminalna čvora)



OREZIVANJE STABLA KROZ KROS VALIDACIJU



Grafikon potvrđuje da veličina stabla utvrđena kros validacijom ($n=3$), vodi minimalnoj grešci na test setu

PREDNOSTI I NEDOSTACI STABALA ODLUČIVANJA

Prednosti:

- Mogu se grafički predstaviti i jednostavno interpretirati
- Mogu se primeniti kako na klasifikacione, tako i regresione probleme
- Mogu se primeniti i u slučaju da atributi imaju nedostajuće vrednosti

Nedostaci:

- Daju slabije rezultate (manje tačne predikcije) nego drugi pristupi nadgledanog m. učenja

MERE USPEŠNOSTI KLASIFIKATORA

Neke od najčešće korišćenih metrika:

- Matrica zabune (Confusion Matrix)
- Tačnost (Accuracy)
- Preciznost (Precision) i Odziv (Recall)
- F mera (F measure)
- Površina ispod ROC krive (Area Under the Curve - AUC)

MATRICA ZABUNE (CONFUSION MATRIX)

Služi kao osnova za računanje mera performansi (uspešnosti) algoritama klasifikacije

U slučaju binarne klasifikacije izgleda ovako:

| | | Predicted Class | |
|--------------|-----|-----------------|----|
| | | Yes | No |
| Actual Class | Yes | TP | FN |
| | No | FP | TN |

TP = True Positive

FP = False Positive

TN = True Negative

FN = False Negative

MATRICA ZABUNE - PRIMER

Primer: klasifikacija studenata da li će položiti ispit (Yes) ili ne (No)

Značenje termina u matrici:

TP: broj studenata za koje je predviđeno da će položiti ispit i koji su stvarno položili

TN: broj studenata za koje je predviđeno da neće položiti ispit i koji ga stvarno nisu položili

FP: broj studenata za koje je predviđeno da će položiti ispit, a nisu ga položili (pogrešno su klasifikovani kao da će položiti)

FN: broj studenata za koje je predviđeno da neće položiti ispit, a položili su ga (pogrešno su klasifikovani kao da neće položiti)

TAČNOST (ACCURACY)

Tačnost (Accuracy) predstavlja procenat slučajeva (instanci) koji su uspešno (korektno) klasifikovani

$$\text{Accuracy} = (TP + TN) / N$$

gde je:

- TP – True Positive; TN – True Negative
- N – ukupan broj uzoraka (instanci) u skupu podataka

| | | Predicted Class | |
|--------------|-----|-----------------|----|
| | | Yes | No |
| Actual Class | Yes | TP | FN |
| | No | FP | TN |

TAČNOST (ACCURACY)

U slučaju vrlo neravnomerne raspodele instanci između klasa (tzv. skewed classes), ova mera je nepouzdana

Npr. u slučaju klasifikacije poruka na spam vs. non-spam, možemo imati skup za trening sa 0.5% spam poruka

Ako primenimo “klasifikator” koji svaku poruku svrstava u non-spam klasu, dobijamo tačnost od 99.5%

Očigledno je da ova metrika nije pouzdana i da su u slučaju skewed classes potrebne druge metrike

PRECIZNOST (PRECISION) I ODZIV (RECALL)

Precision = $TP / \text{no. predicted positive} = TP / (TP + FP)$

Npr. od svih poruka koje su *označene kao spam* poruke, koji procenat čine poruke koje su stvarno spam

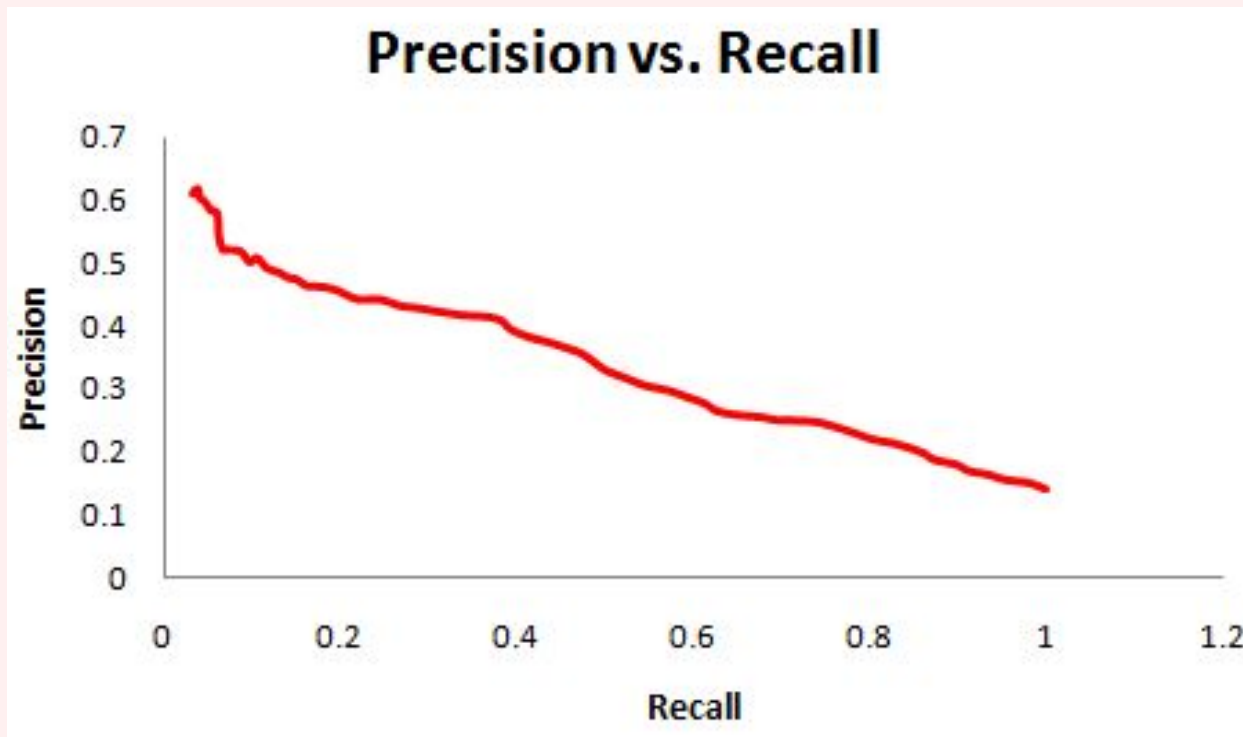
Recall = $TP / \text{no. actual positive} = TP / (TP + FN)$

Npr. od svih poruka koje su *stvarno spam* poruke, koji procenat poruka je detektovan/klasifikovan kao spam

| | | Predicted Class | |
|--------------|-----|-----------------|----|
| | | Yes | No |
| Actual Class | Yes | TP | FN |
| | No | FP | TN |

PRECIZNOST I ODZIV

U praksi je nužno praviti kompromis između ove dve mere: ako želimo da povećamo Odziv, smanjićemo Preciznost, i obrnuto.



Izvor:

<http://groups.csail.mit.edu/cb/struct2net/webserver/images/prec-v-recall-v2.png>

F MERA (F MEASURE)

F mera kombinuje Preciznost i Odziv i omogućuje jednostavnije poređenje dva ili više algoritama

$$F = (1 + \beta^2) * \text{Precision} * \text{Recall} / (\beta^2 * \text{Precision} + \text{Recall})$$

Parametar β kontroliše koliko više značaja će se pridavati Odzivu u odnosu na Preciznost

U praksi se najčešće koristi tzv. F1 mera („balansirana“ F mera) koja daje podjednak značaj i Preciznosti i Odzivu:

$$F1 = 2 * \text{Precision} * \text{Recall} / (\text{Precision} + \text{Recall})$$

POVRŠINA ISPOD ROC KRIVE

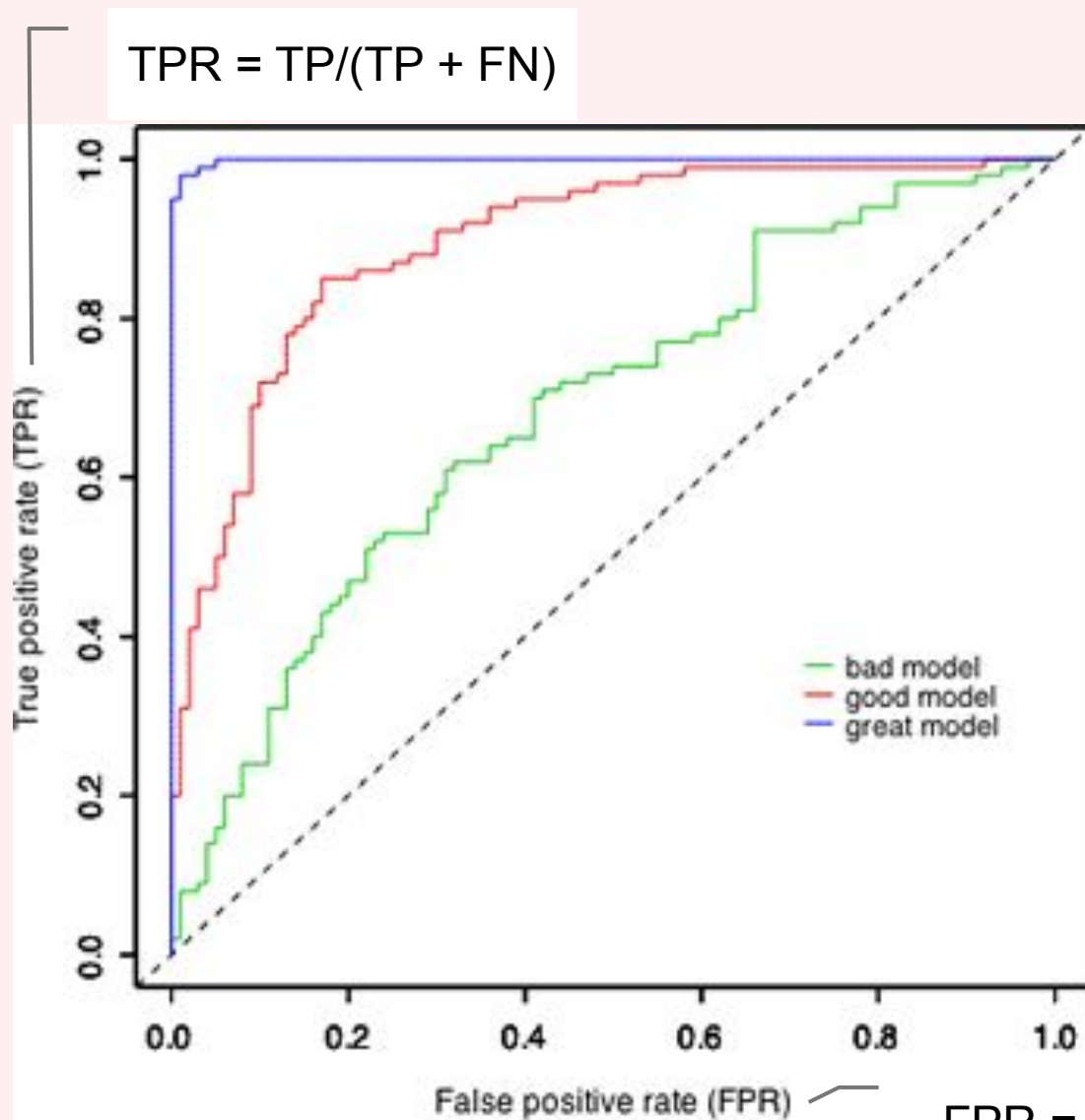
Površina ispod ROC* krive – Area Under the Curve (AUC):

- meri diskriminacionu moć klasifikatora tj. sposobnost da razlikuje instance koje pripadaju različitim klasama
- primenjuje se za merenje performansi binarnih klasifikatora
- vrednost za AUC se kreće u intervalu 0-1
- za metodu slučajnog izbora važi da je $AUC = 0.5$; što je AUC vrednost klasifikatora > 0.5 , to je klasifikator bolji
 - 0.7–0.8 se smatra prihvatljivim; 0.8–0.9 jako dobrim; sve > 0.9 je odlično

*ROC = Receiver Operating Characteristic;

http://en.wikipedia.org/wiki/Receiver_operating_characteristic

POVRŠINA ISPOD ROC KRIVE



$FPR = FP / (FP + TN)$

3.

KLASTERIZACIJA

PREGLED TEMA

- Šta je klasterizacija?
- Koje su oblasti/primeri primene?
- Klasterizacija primenom K-Means algoritma
 - Upoznavanje sa algoritmom kroz primer
 - K-Means algoritam
 - Potencijalni problemi pri primeni algoritma

ŠTA JE KLASSTERIZACIJA?

Klasterizacija je jedan od oblika nenadgledanog m. učenja

- ono što je raspoloživo od podataka su podaci o instancama koje je potrebno na neki način grupisati;
- ne posedujemo podatke o poželjnoj / ispravnoj grupi za ulazne instance

ŠTA JE KLASSTERIZACIJA?

Klasterizacija je zadatak grupisanja instanci, tako da za svaku instancu važi da je *sličnija* instancama iz svoje grupe (klastera), nego instancama iz drugih grupa (klastera)

Sličnost instanci se određuje primenom neke od mera za računanje

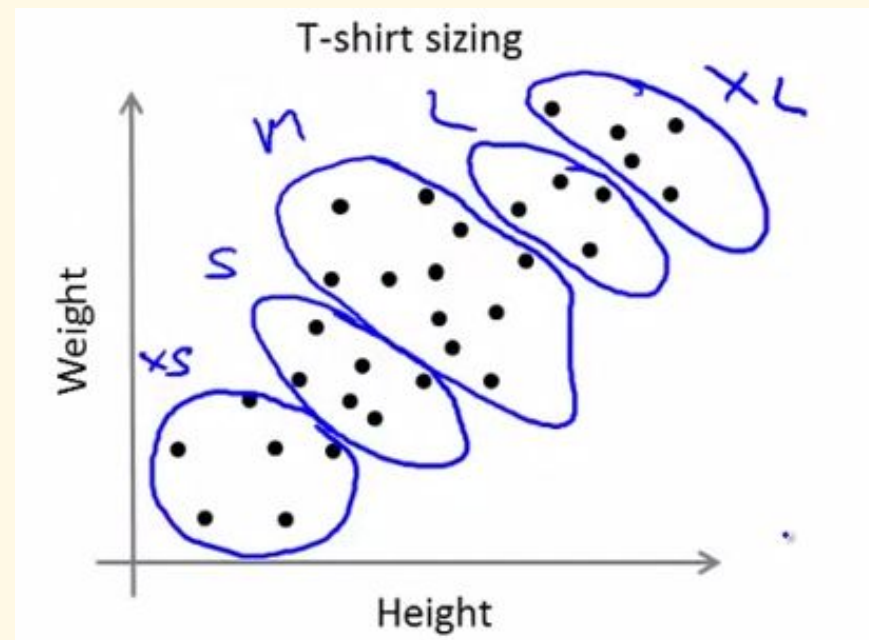
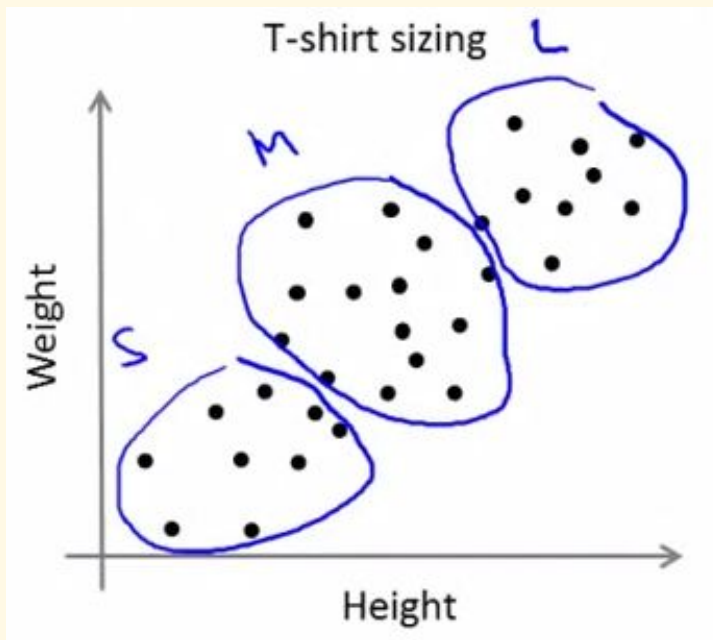
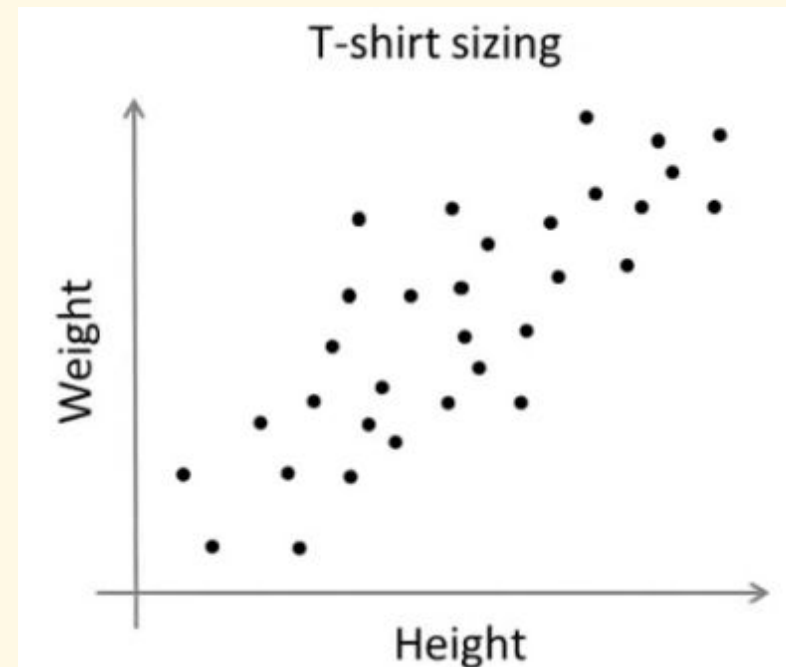
- sličnosti (npr. Kosinusna sličnost) ili
- udaljenosti dva objekta (npr. Euklidska udaljenost)

ŠTA JE KLASTERIZACIJA?

Za razliku od klasifikacije, ovde nemamo “tačno” rešenje

- ocena uspešnosti algoritma je dosta teža nego kod klasifikacije
- pogodnost rešenja zavisi od domena i slučaja primene – jedno isto rešenje može biti različito ocenjeno u različitim slučajevima primene
- zahteva angažovanje domenskih eksperata koji će evaluirati rešenje

Primer različitih dobrih
rešenja za isti polazni skup
podataka



OBLASTI PRIMENE

- Segmentacija tržišta
- Uočavanje grupa u društvenim mrežama
- Identifikacija paterna u ponašanju korisnika nekog Web sajta
- Grupisanje objekata (npr., slika/dokumenata) prema zajedničkim karakteristikama
- ...

3.1. K-MEANS ALGORITAM

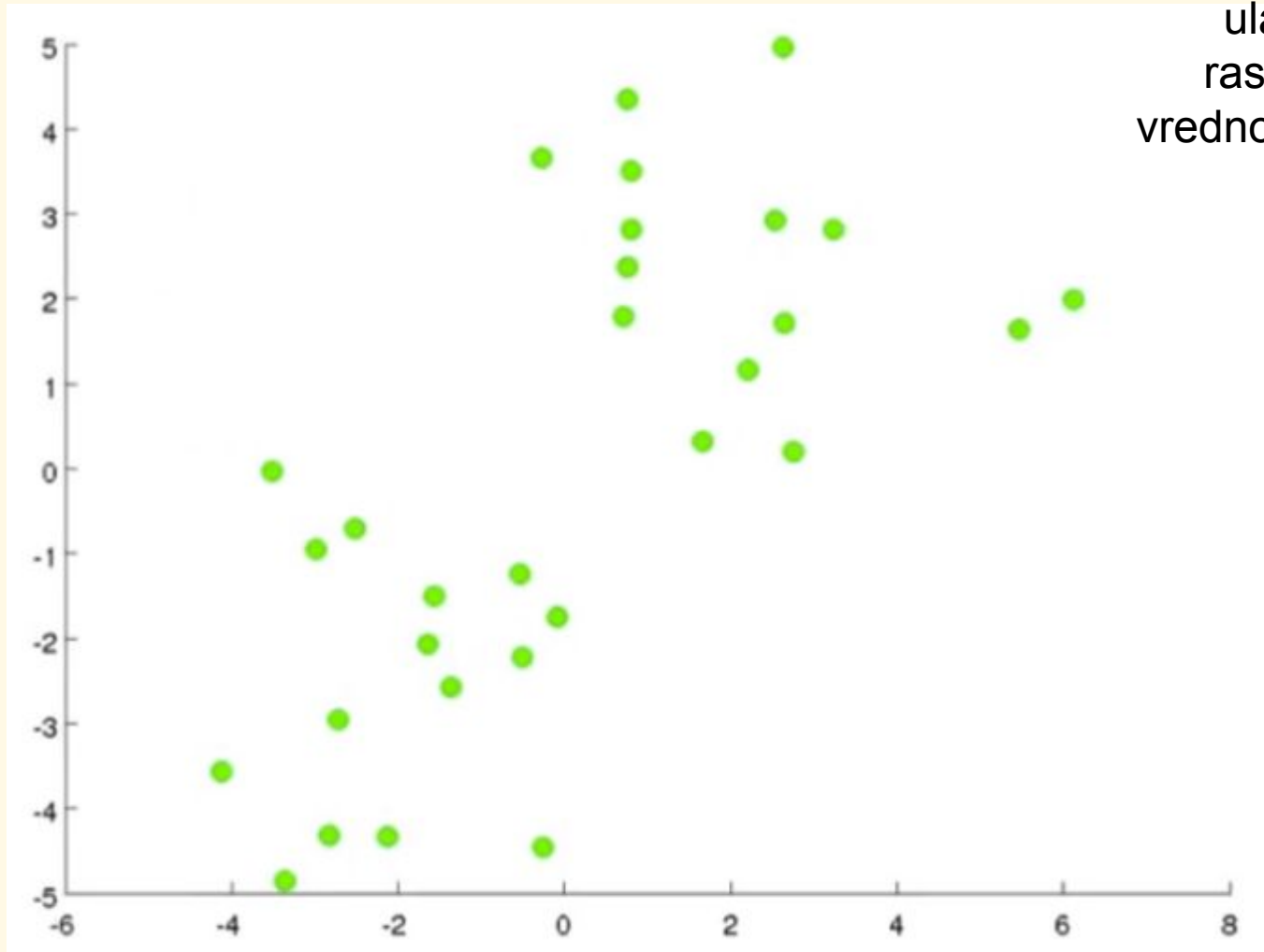
K-MEANS

Jedan od najpoznatijih i najjednostavnijih algoritama klasterizacije

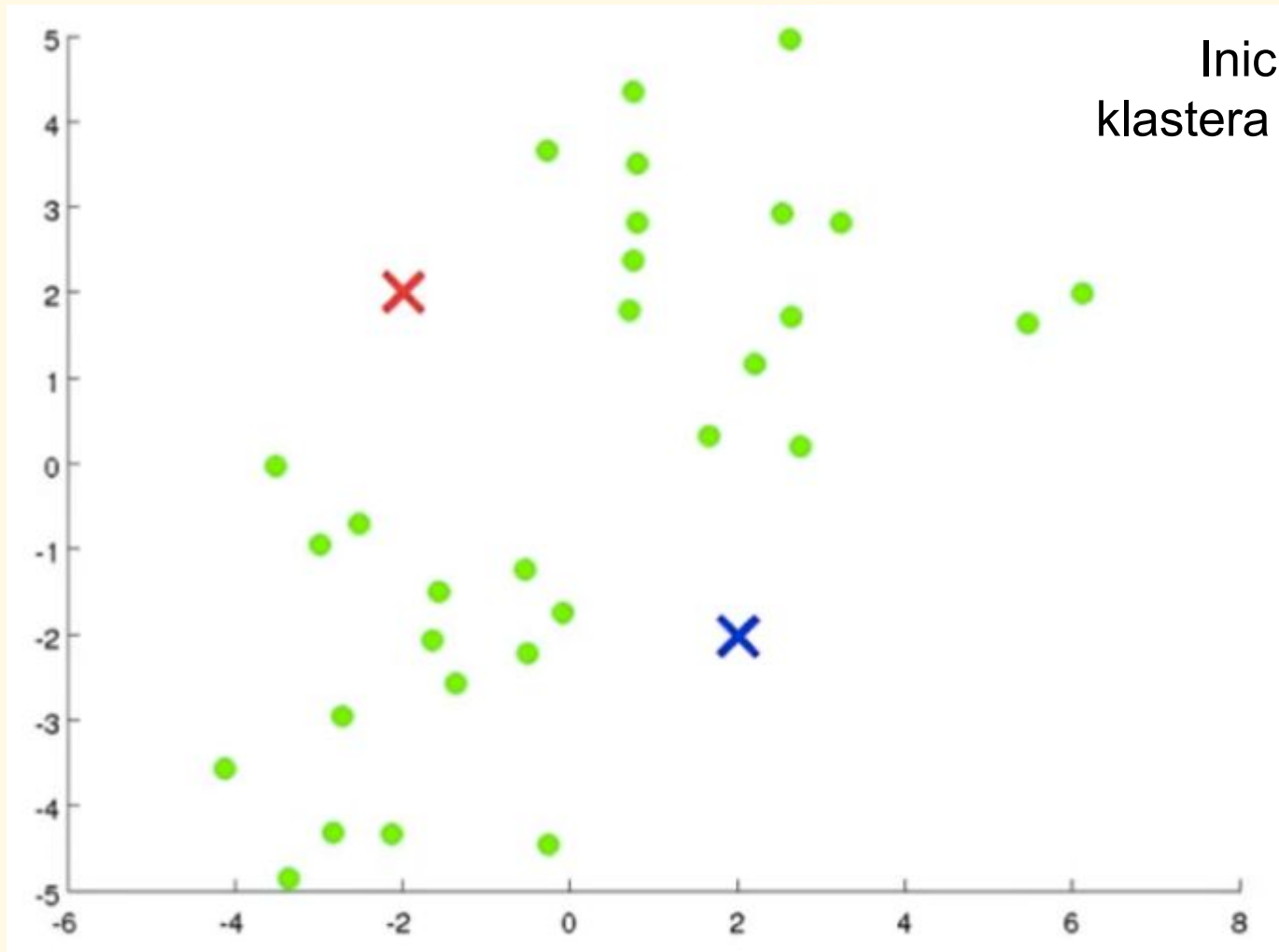
Najlakše ga je razumeti na primeru, pa ćemo prvo razmotriti jedan primer

K-MEANS ALGORITAM — PRIMER

Pretpostavimo da su ovo
ulazni podaci kojima
raspoložemo, opisani
vrednostima dva atributa

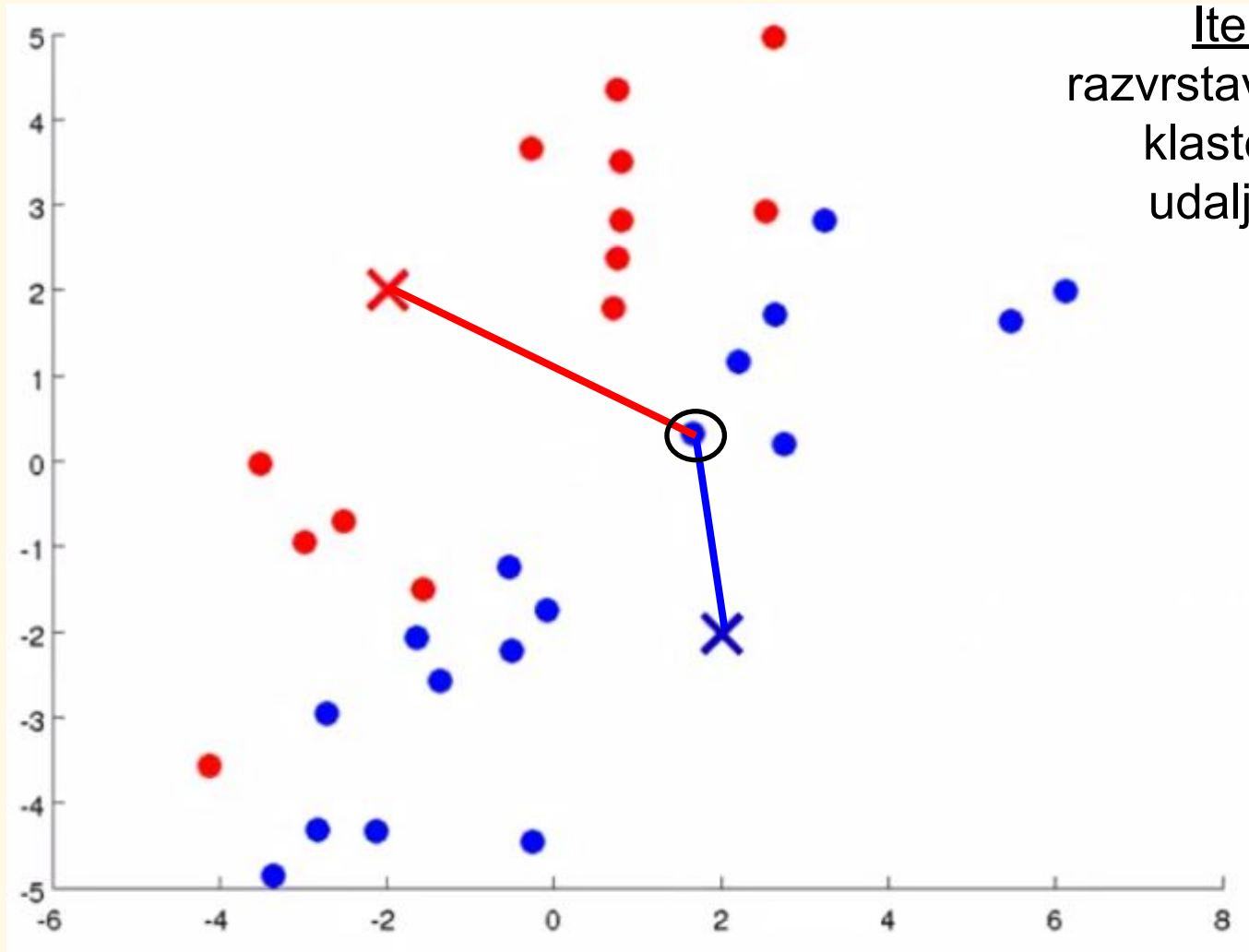


K-MEANS: PRIMER



Inicijalizacija:
Inicijalni izbor težišta
klastera ($K = 2$) metodom
slučajnog izbora

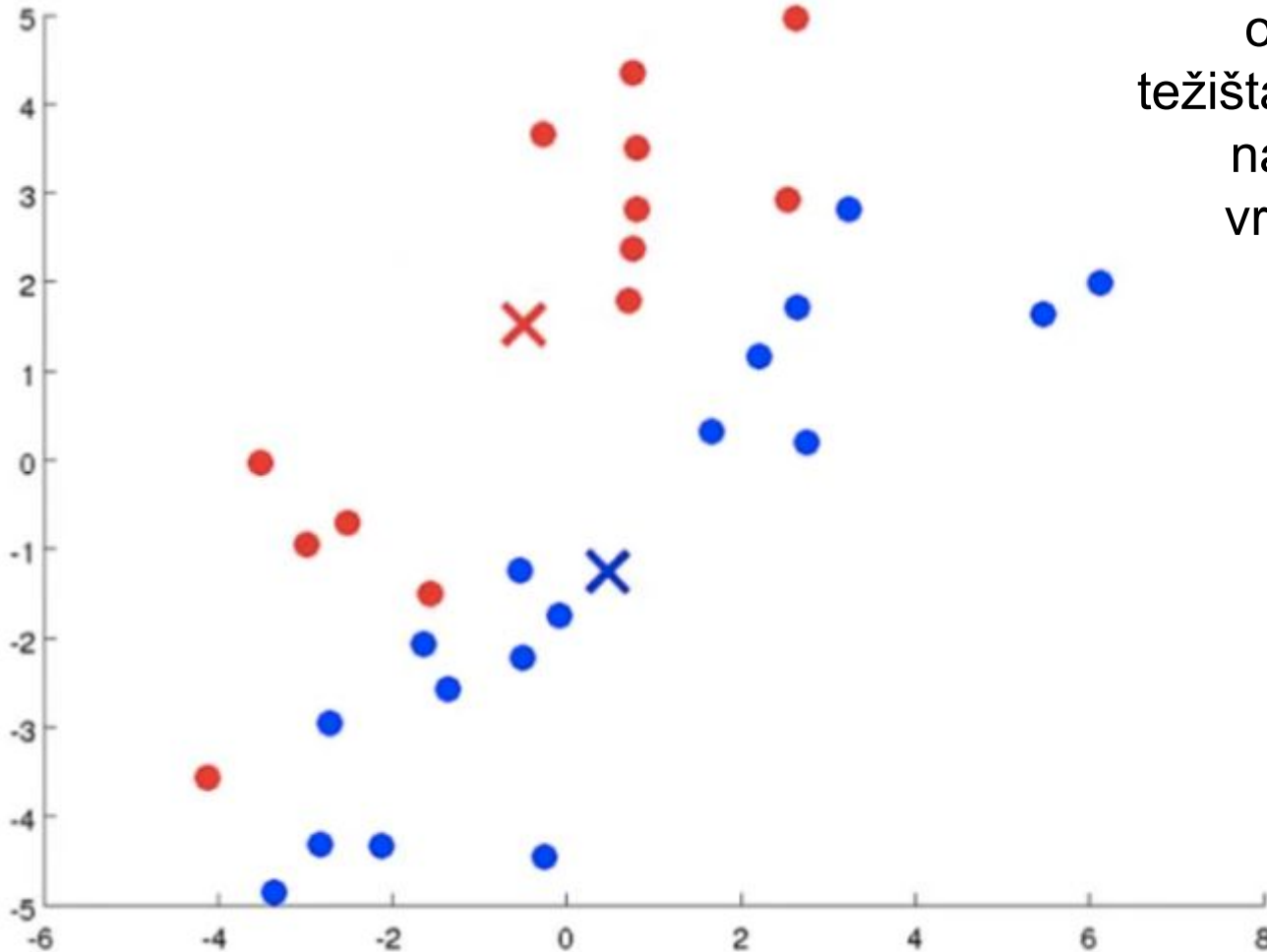
K-MEANS: PRIMER



Iteracija 1, korak 1:
razvrstavanje instanci po
klasterima na osnovu
udaljenosti od težišta
klastera

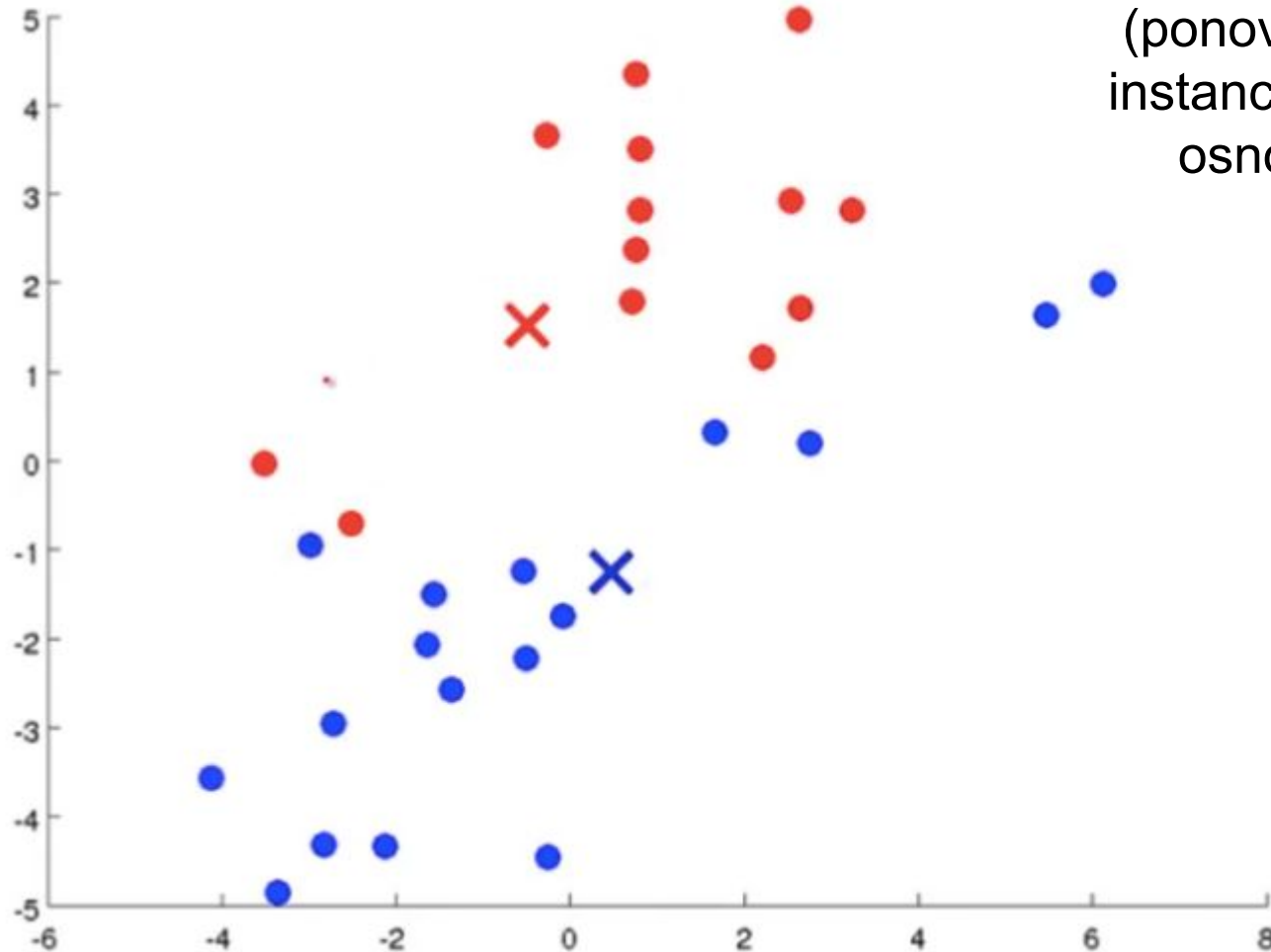
K-MEANS: PRIMER

Iteracija 1, korak 2:
određivanje novog
težišta za svaki klaster,
na osnovu proseka
vrednosti instanci u
datom klasteru

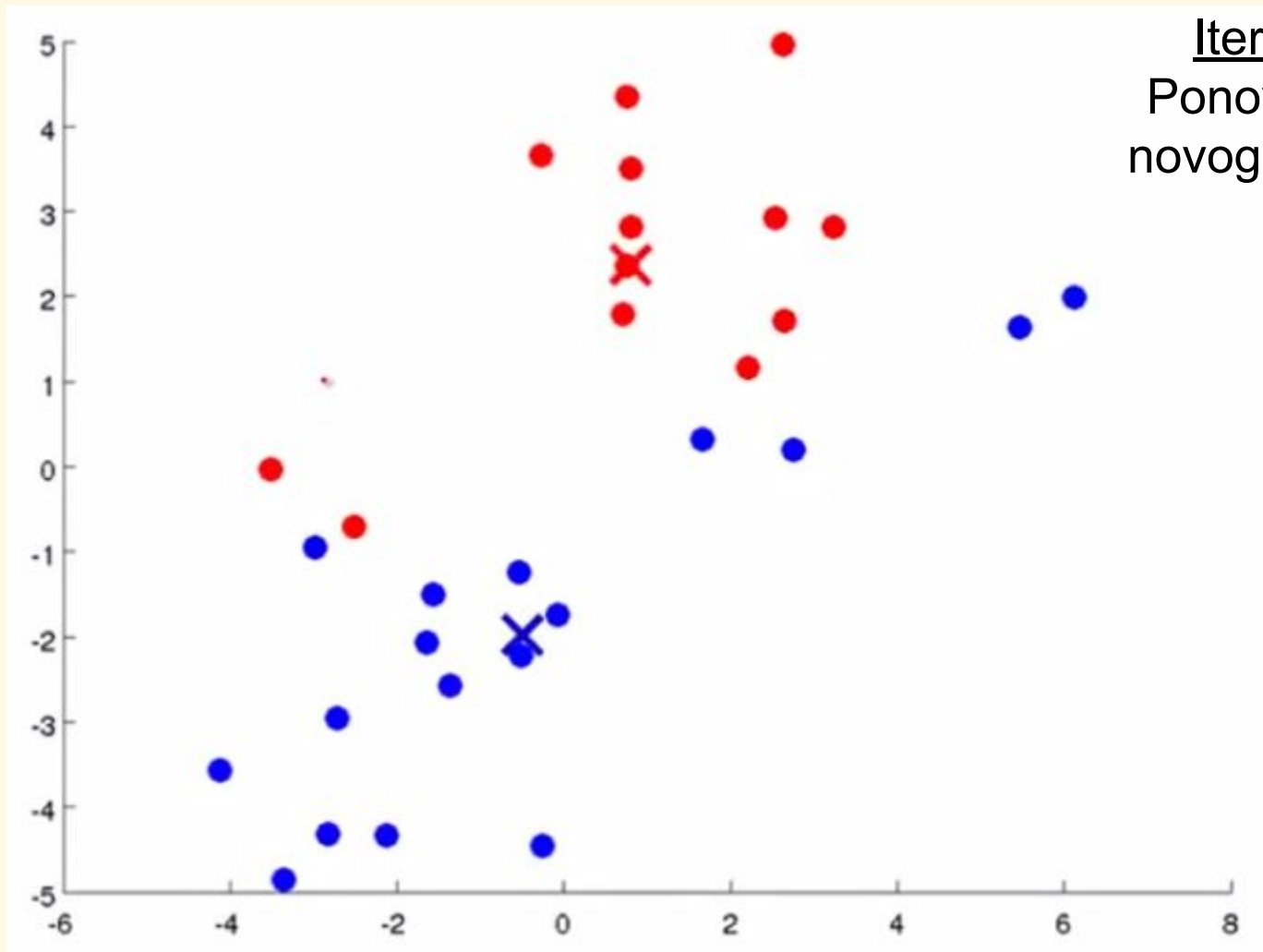


K-MEANS: PRIMER

Iteracija 2, korak 1:
(ponovno) razvrstavanje
instanci po klasterima na
osnovu udaljenosti od
težišta klastera

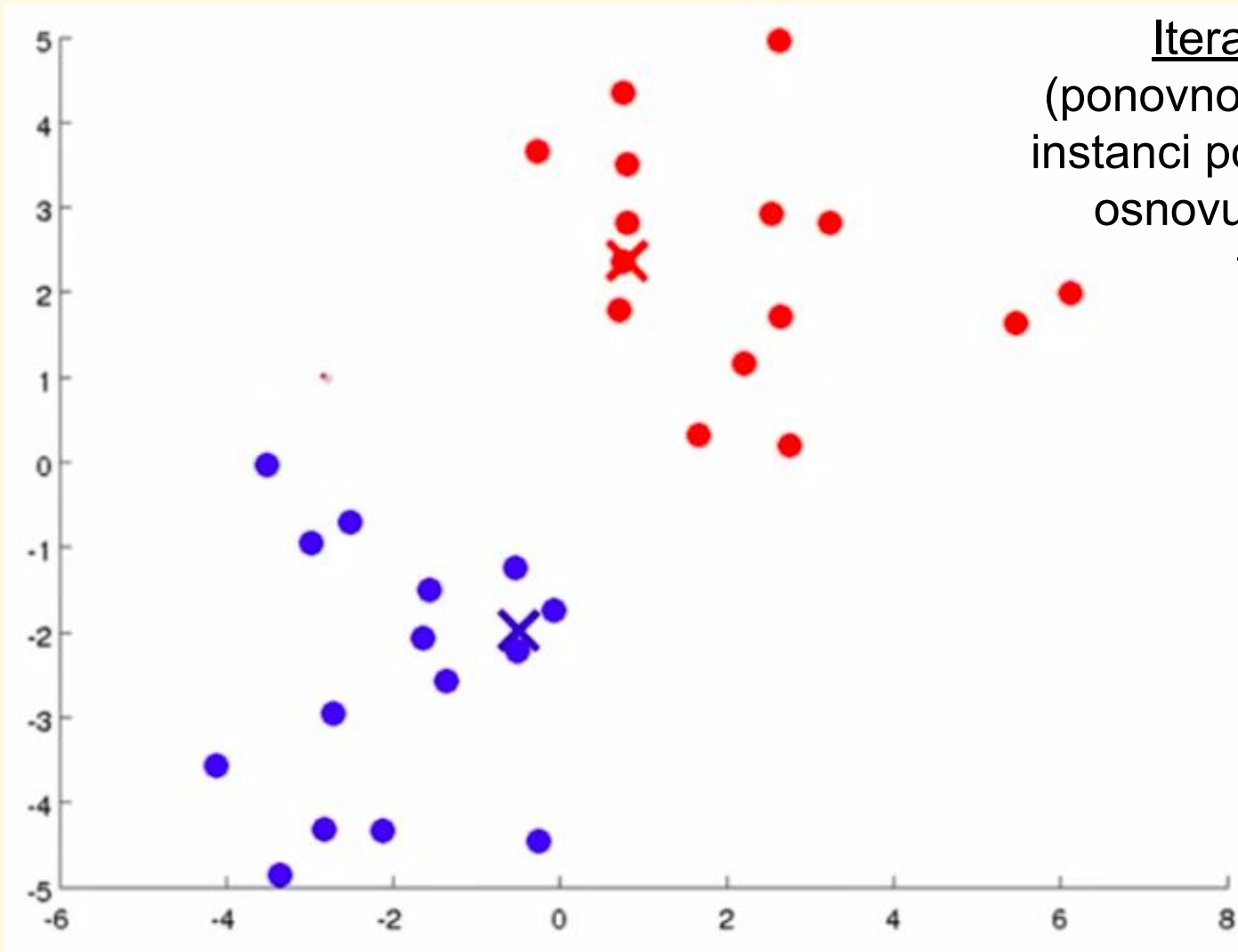


K-MEANS: PRIMER



Iteracija 2, korak 2:
Ponovno određivanje
novog težišta za svaki
klaster

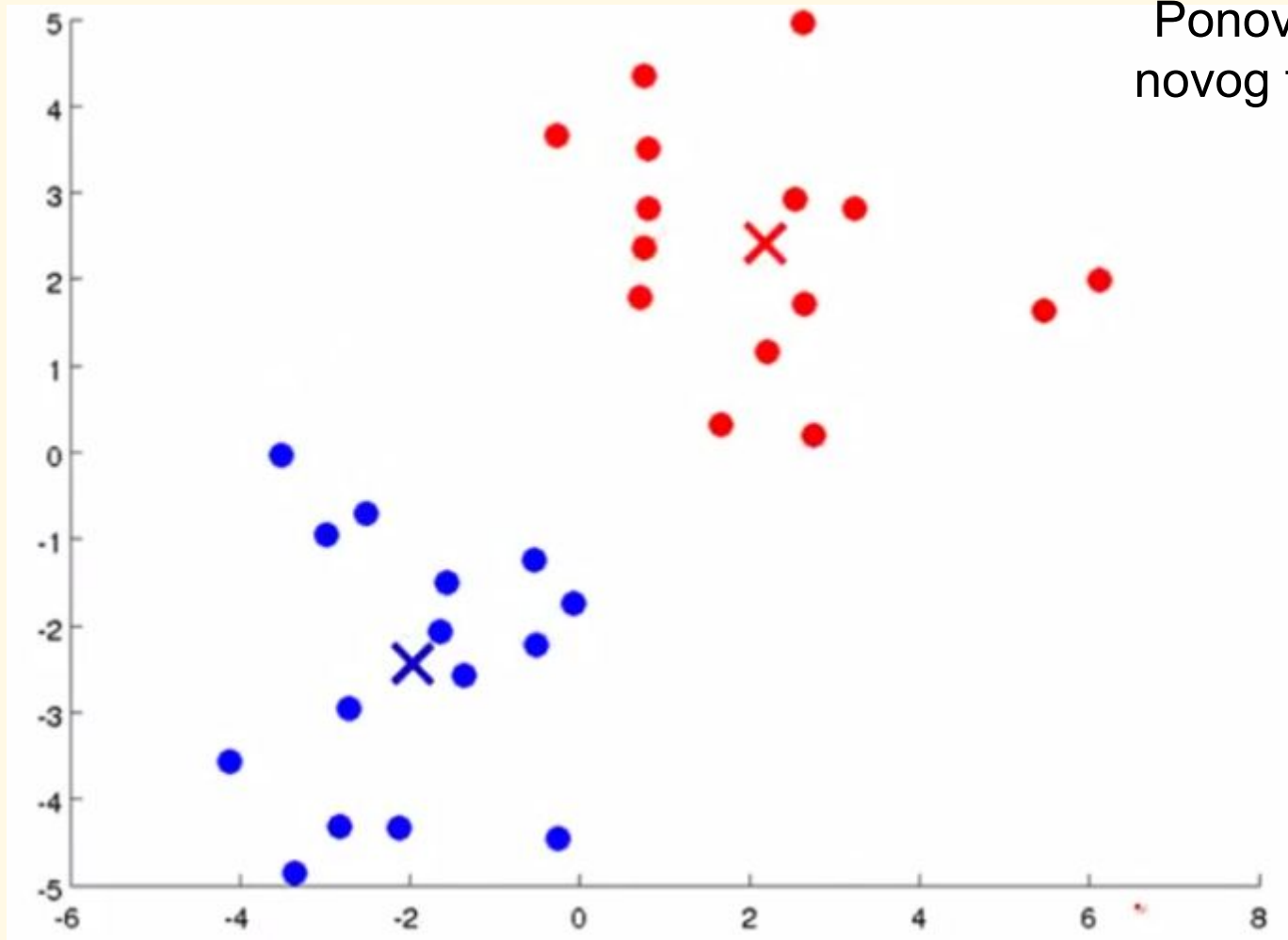
K-MEANS: PRIMER



Iteracija 3, korak 1:
(ponovno) razvrstavanje
instanci po klasterima na
osnovu udaljenosti od
težišta klastera

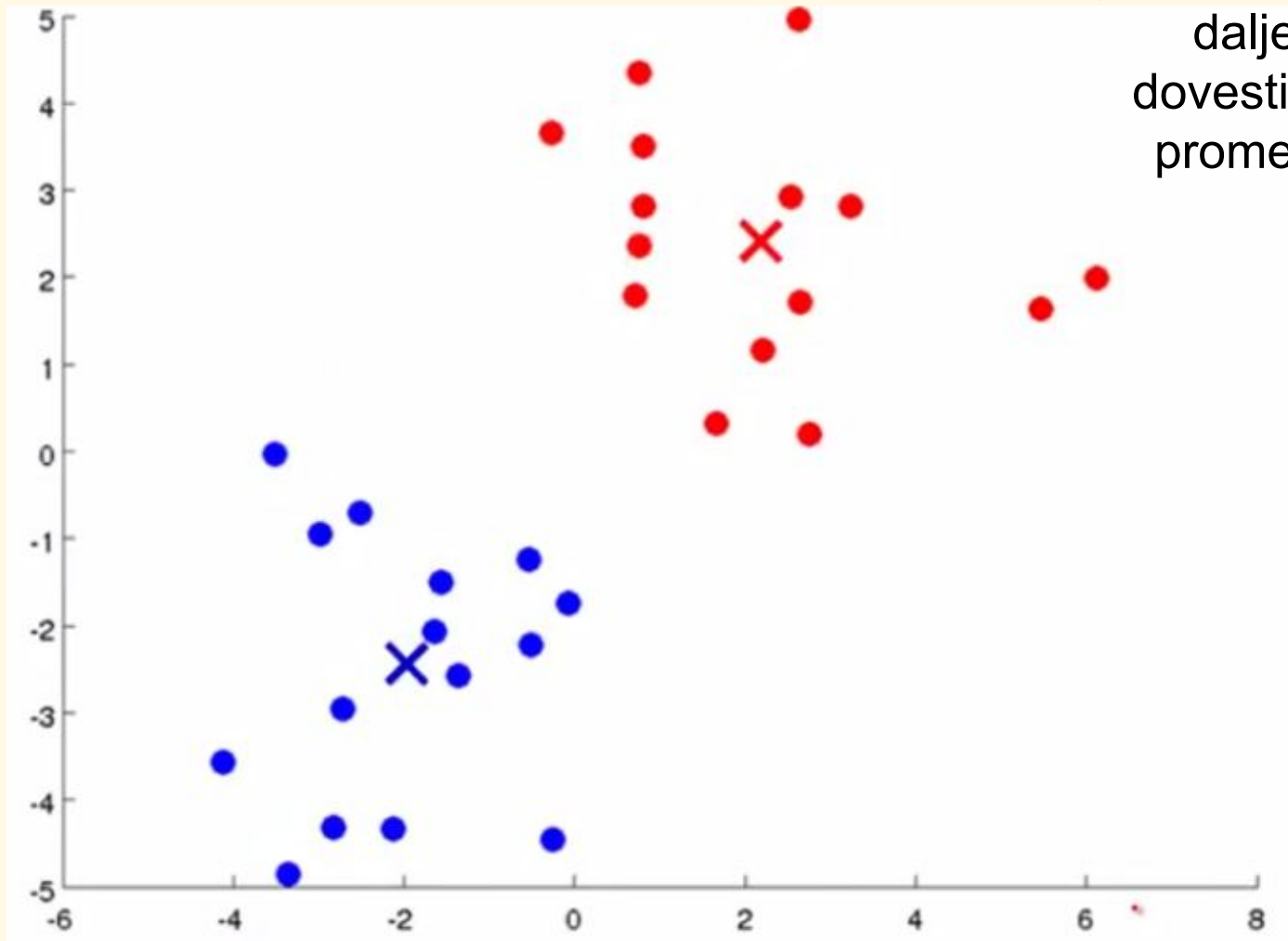
K-MEANS: PRIMER

Iteracija 3, korak 2:
Ponovno određivanje
novog težišta za svaki
klaster



K-MEANS: PRIMER

Algoritam već konvergira:
dalje iteracije neće
dovesti do značajnijih
promena i proces se
zaustavlja



K-MEANS: ALGORITAM

Ulaz:

- K - broj klastera
- (neobebežen) skup za trening sa m instanci; svaka instanca u skupu je vektor opisan sa n atributa (x_1, x_2, \dots, x_n)
- max - max broj iteracija (opcionni parametar)

K-MEANS: ALGORITAM

Koraci:

- 1) Inicijalni izbor težišta klastera, slučajnim izborom
 - težišta se biraju iz skupa instanci za trening, tj. K instanci za trening se nasumično izabere i proglasi za težišta
- 2) Ponoviti dok algoritam ne konvergira ili broj iteracija $\leq \max$:
 - 1) *Grupisanje po klasterima*: za svaku instancu iz skupa za trening, $i = 1, m$, identifikovati najbliže težište i dodeliti instancu klasteru kome to težište pripada
 - 2) *Pomeranje težišta*: za svaki klaster izračunati novo težište uzimajući prosek tačaka (instanci) koje su dodeljene tom klasteru

K-MEANS ALGORITAM: FUNKCIJA KOŠTANJA

Smisao K-means algoritma je *minimizacija funkcije koštanja J* (cost function):

$$J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K) = \frac{1}{m} \sum_{i=1}^m \|x^{(i)} - \mu_{c^{(i)}}\|^2$$

$x^{(i)}$ – i -ta instanca u skupu podataka za trening, $i=1, m$

$c^{(i)}$ – indeks klastera u koji je instanca $x^{(i)}$ trenutno raspoređena

μ_j – težište klastera j , $j=1, K$

$\mu_{c^{(i)}}$ – težište klastera u koji je instanca $x^{(i)}$ trenutno raspoređena

Ova funkcija se zove i funkcija distorzije (distortion function)

K-MEANS ALGORITAM: FUNKCIJA KOŠTANJA

$$\min_{\substack{c^{(1)}, \dots, c^{(m)}, \\ \mu_1, \dots, \mu_K}} J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$$

Minimizacija funkcije koštanja J kroz K-means algoritam:

- faza *Grupisanja po klasterima* minimizuje J po parametrima $c^{(1)}, \dots, c^{(m)}$, držeći μ_1, \dots, μ_K fiksnim
- faza *Pomeranja težišta* minimizuje J po parametrima μ_1, \dots, μ_K , držeći $c^{(1)}, \dots, c^{(m)}$ fiksnim

K-MEANS: EVALUACIJA

Kriterijumi za procenu “kvaliteta” kreiranih klastera:

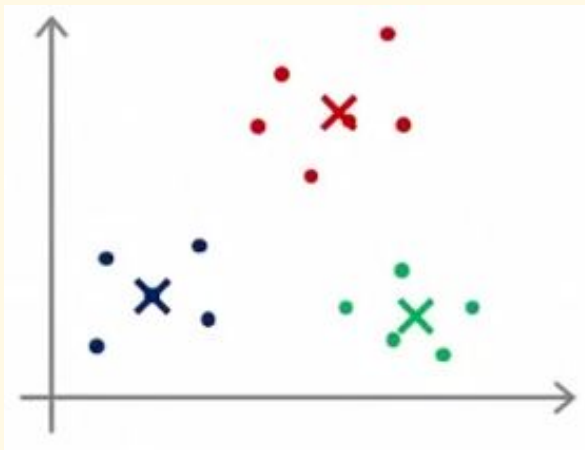
- Međusobna udaljenost težišta
 - što su težišta dalje jedno od drugog, to je stepen preklapanja klastera manji, i njihov kvalitet viši
- St. devijacija pojedinačnih instanci u odnosu na težište
 - što je st. devijacija manja, to su instance tešnje grupisane oko težišta i klasteri se smatraju boljim
- Suma kvadrata greške unutar klastera (within cluster sum of squared errors)
 - daje kvantitativnu meru za procenu kvaliteta kreiranih klastera
 - obično se koristi za procenu broja klastera

K-MEANS:

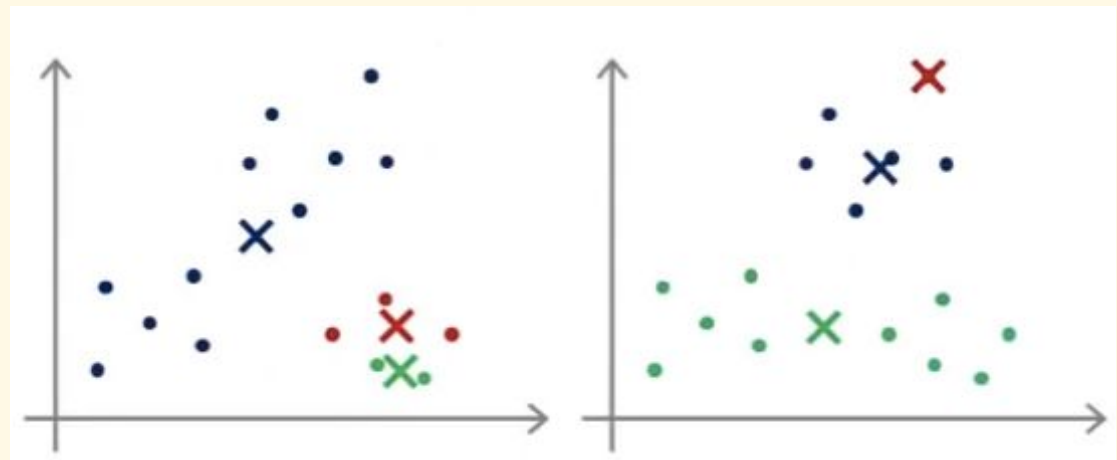
PROBLEM INICIJALNOG IZBORA TEŽIŠTA

Zavisno od inicijalnog izbora težišta:

- K-means algoritam može konvergirati brže ili sporije
- Takođe, može “upasti” u tzv. lokalni minimum i dati loše rešenje
 - reč je o lokalnom min. funkcije koštanja



Dobra inicijalizacija



Inicijalizacija koja vodi u lokalne minimume

K-MEANS:

VIŠESTRUKA NASUMIČNA INICIJALIZACIJA

Omogućuje da se izbegnu situacije koje K-means dovode u lokalni minimum

Sastoji se u sledećem:

```
for i = 1 to n { //n obično uzima vrednosti 50 - 1000
    Nasumično odabrati inicijalni skup težišta;
    Izvršiti K-Means algoritam;
    Izračunati funkciju koštanja (cost function)
}
Izabrati instancu algoritma koja daje najmanju vrednost
za f. koštanja
```

Ovaj pristup daje dobre rezultate ukoliko je broj klastera relativno mali (2 - 10); za veći broj klastera ne bi ga trebalo koristiti

K-MEANS: KAKO ODREDITI K?

Kako odrediti broj klastera K?

- U slučaju da posedujemo znanje o fenomenu/pojavi koju podaci opisuju
 - Pretpostaviti broj klastera (K) na osnovu domenskog znanja
 - Testirati model sa K-1, K, K+1 klastera i uporediti grešku*
- Ukoliko ne posedujemo znanje o fenomenu/pojavi
 - Krenuti od malog broja klastera i u više iteracija testirati model uvek sa jednim klasterom više
 - U svakoj od iteracija, uporediti grešku* tekućeg i prethodnog modela i kad smanjenje greške postaje zanemarljivo, prekinuti postupak

*Na primer, within cluster sum of squared errors

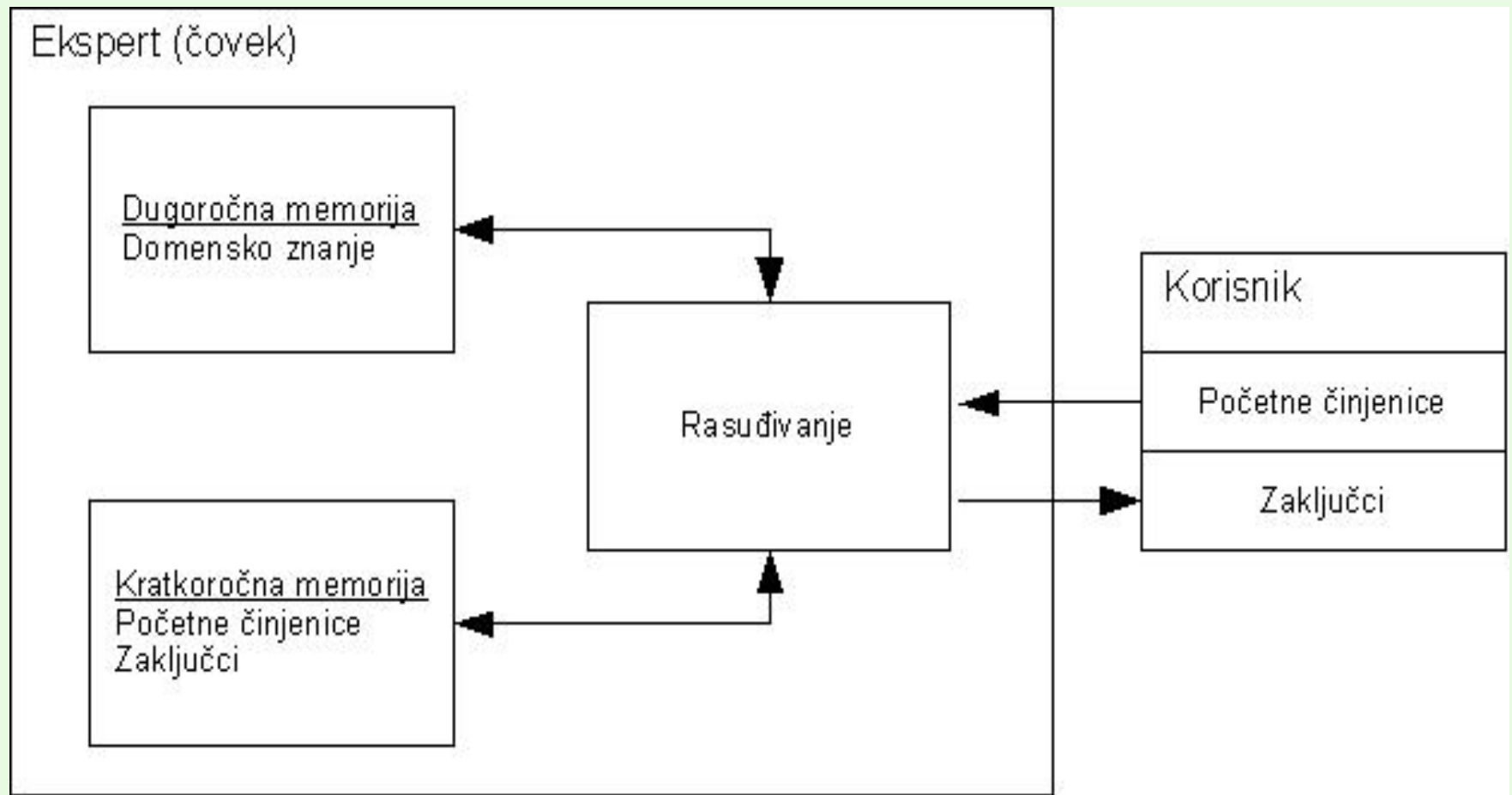
4.

EKSPERTNI SISTEMI

□ EKSPERTNI SISTEMI - OSNOVE

- Ekspertni sistem (ES) je računarski program kojim se emulira rešavanje problema na način na koji to čini ekspert (čovek)
- Da bi neki program mogao da se nazove ES, on mora da:
 - sadrži *ekspertsko znanje* iz neke oblasti
 - omogućava *automatizovano rezonovanje*

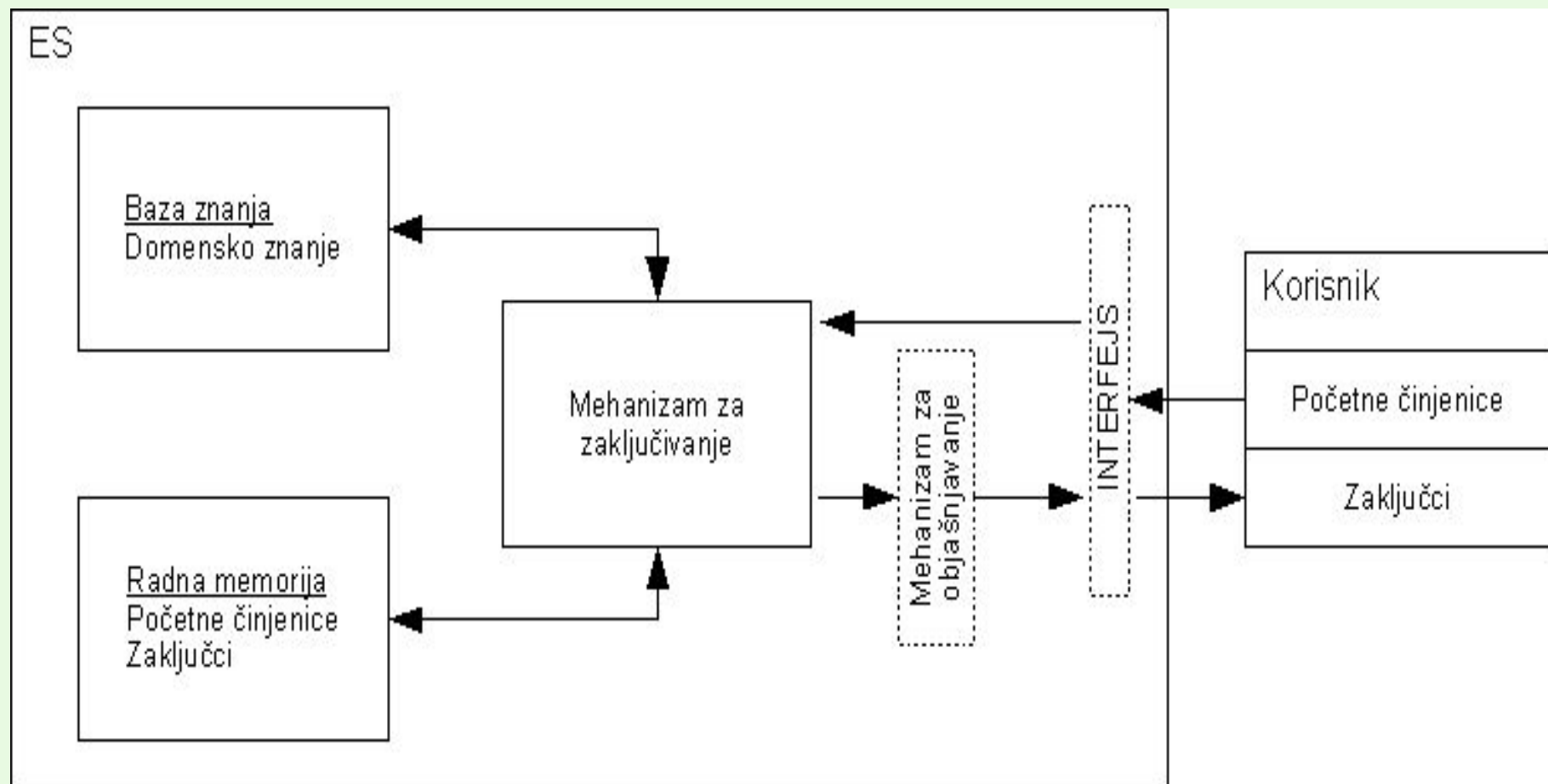
MODEL REZONOVANJA ČOVEKA



□ MODEL REZONOVANJA ČOVEKA

- Dugoročna memorija sadrži domensko znanje (domen = oblast):
“Ako je napolju oblačno, verovatno će padati kiša”
- Kratkoročna memorija sadrži činjenice
“Napolju je oblačno”
- Rasuđivanje - spajanje sadržaja iz obe vrste memorije i izvođenje zaključaka
“Padaće kiša”

ARHITEKTURA EKSPERTNOG SISTEMA



▣ BAZA ZNANJA

- Sadrži domensko znanje
- Domensko znanje mora da bude formalizovano (da bi računar mogao da ga koristi)
- Najčešće se koristi tehnika pravila za predstavljanje znanja u okviru ES

IF

Napolju je oblačno

THEN

Padaće kiša

▣ BAZA ZNANJA

- Pravila se sastoje iz IF i THEN dela i povezuju uslov (premisu) sa zaključkom:

IF

Auto neće da “upali” (premisu)

THEN

Kvar može da bude u
električnom sistemu (zaključak)

- Premisa može da bude i složena
 - više jednostavnih premisa povezanih logičkim operatorima AND, OR i NOT

▣ BAZA ZNANJA

- Osnovna i najvažnija karakteristika pravila je da mogu da se “ulančavaju”
- Ulančavanje pravila se postiže time što zaključak jednog pravila predstavlja premisu drugog

▣ BAZA ZNANJA

IF Auto neće da “upali” AND Napon na akumulatoru < 12V

THEN Akumulator je prazan

IF Akumulator je prazan

THEN Napuni akumulator

IF Auto neće da “upali” AND Napon na akumulatoru = 12V

THEN Anlaser je neispravan

IF Anlaser je neispravan

THEN Zameni anlaser

▣ RADNA MEMORIJA

- Sadrži činjenice i zaključke
- Zaključci nisu ništa drugo nego činjenice koje su nastale kao posledica rezonovanja
- I činjenice moraju da budu formalizovane
- Za predstavljanje činjenica koriste se:
 - Klase i objekti – nekada su to bili okviri (frames)
 - Obične promenljive

▣ RADNA MEMORIJA

- Okvir (frame)
 - forma za predstavljanje znanja o nekom objektu
 - analogija koncepta klasa u OO programskim jezicima
 - sadrži:
 - deklarativno znanje – opisno znanje o objektu
 - proceduralno znanje – šta objekti mogu da “rade”
 - slot – polje okvira, nosilac deklarativnog znanja

▣ RADNA MEMORIJA

- Okvir (frame)

— Primeri:

Covek.visina = 185

Vreme.temperatura = 17

Automobil.problem = “Neće da upali”

Automobil.naponNaAkumulatoru = 12.3

RADNA MEMORIJA

- Kada se pravila i okviri koriste zajedno za predstavljanje znanja, to izgleda ovako:

IF Auto.problem = “neće da upali” AND

Auto.napon_na_akumulatoru < 12

THEN

Auto.uzrok_problema = “Prazan akumulator”

IF

Auto.uzrok_problema = “Prazan akumulator”

THEN

Auto.resenje = “Napuni akumulator”

▣ MEHANIZAM ZA ZAKLJUČIVANJE

- Kombinuje znanje iz baze znanja i činjenice iz radne memorije i stvara nove zaključke
- Omogućava automatizovano rezonovanje

□ MEHANIZAM ZA ZAKLJUČIVANJE

- Izbor tehnike zaključivanja zavisi od korišćene tehnike za predstavljanje znanja
- Najpopularnije tehnike za zaključivanje:
 - **Ulančavanje unapred** (Forward chaining)
 - **Ulančavanje unazad** (Backward chaining)
- Ove dve tehnike mogu da se koriste isključivo u kombinaciji sa pravilima

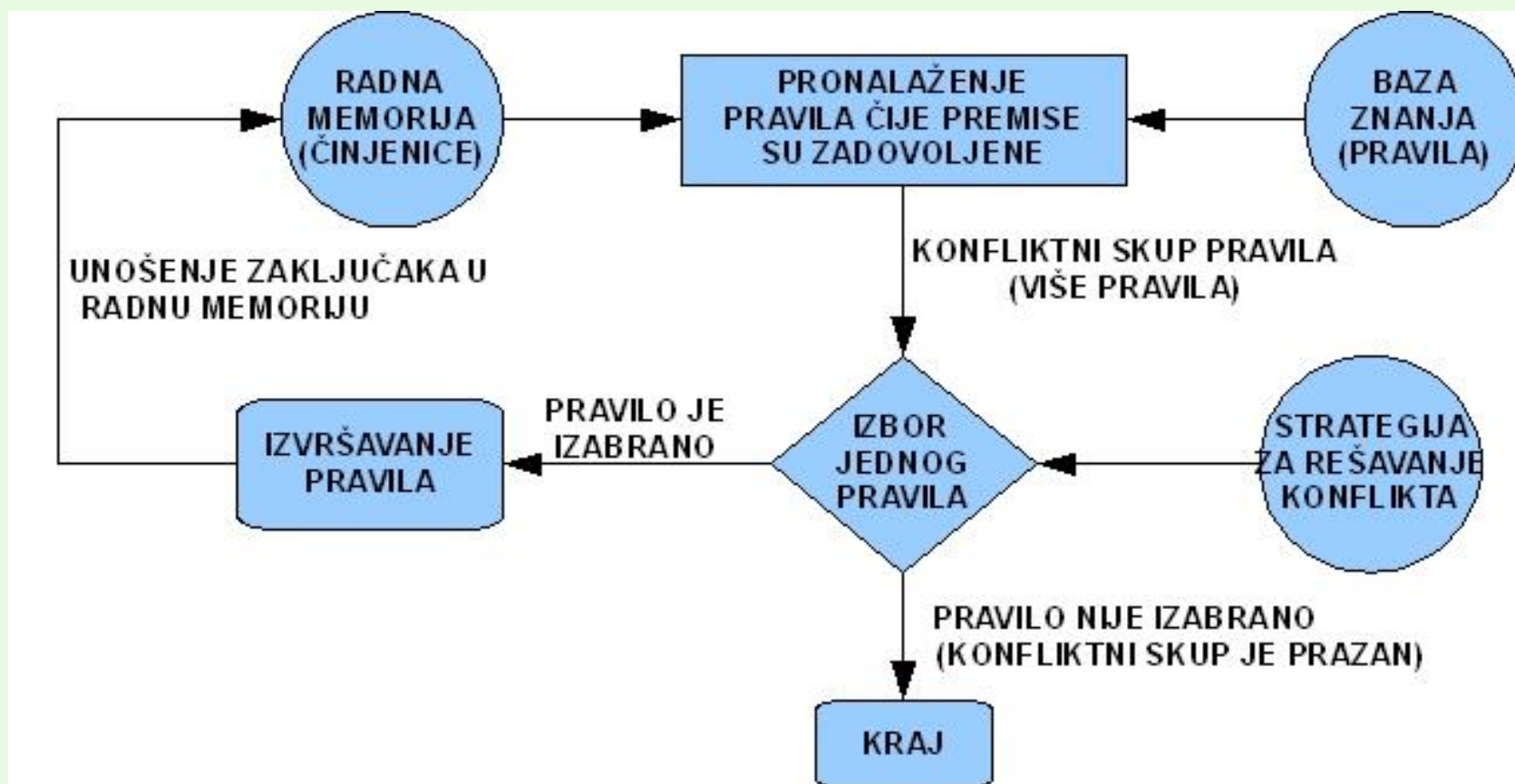
▣ MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred

- Zaključivanje “vođeno podacima” (data-driven)
- Na osnovu ulaznih podataka se pokušava zaključiti što više o problemu
- Poznat i kao “prepoznaj-razreši-izvrši” ciklus (recognize-resolve-act)

MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred - algoritam



□ MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – algoritam

Korak 1 – Pronaći sva pravila čije premise su zadovoljene (ova pravila čine konfliktni skup).

Korak 2 – Iz konfliktnog skupa izabrati samo jedno pravilo (korišćenjem strategije za rešavanje konflikta). Ako je konfliktni skup prazan, to je kraj.

Korak 3 – Izvršiti izabrano pravilo (uneti zaključke tog pravila kao činjenice u radnu memoriju) i ići na korak 1.

□ MEHANIZAM ZA ZAKLJUČIVANJE

- Ulančavanje unapred – strategije za rešavanje konflikta
 - izbor prvog pravila
 - izbor pravila sa najvišim prioritetom
 - izbor najspecifičnijeg pravila
(sa najsloženijom premisom)
 - izbor pravila koje se odnosi na najskorije dodate činjenice
 - svako pravilo može samo jednom da se izvrši
- Najčešće se koristi više strategija odjednom

□ MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer

— neka baza znanja sadrži sledeća pravila:

IF Auto neće da “upali” AND Napon na akumulatoru < 12V
THEN Akumulator je prazan

IF Akumulator je prazan
THEN Napuni akumulator

IF Auto neće da “upali” AND Napon na akumulatoru = 12V
THEN Anlaser je neispravan

IF Anlaser je neispravan
THEN Zameni anlaser

□ MECHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer

— radna memorija sadrži sledeće početne činjenice:

Auto neće da “upali”

Napon na akumulatoru = 11V

MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred - primer (početak)

RADNA MEMORIJA

- 1) Auto neće da "upali"
- 2) Napon na akumulatoru = 11V

MEHANIZAM ZA ZAKLJUČIVANJE

Konfliktni skup pravila

Izvrši pravilo

BAZA ZNANJA

PRAVILO 1

IF
Auto neće da "upali" AND
Napon na akumulatoru $< 12V$
THEN
Akumulator je prazan

PRAVILO 2

IF
Akumulator je prazan
THEN
Napuni akumulator

PRAVILO 3

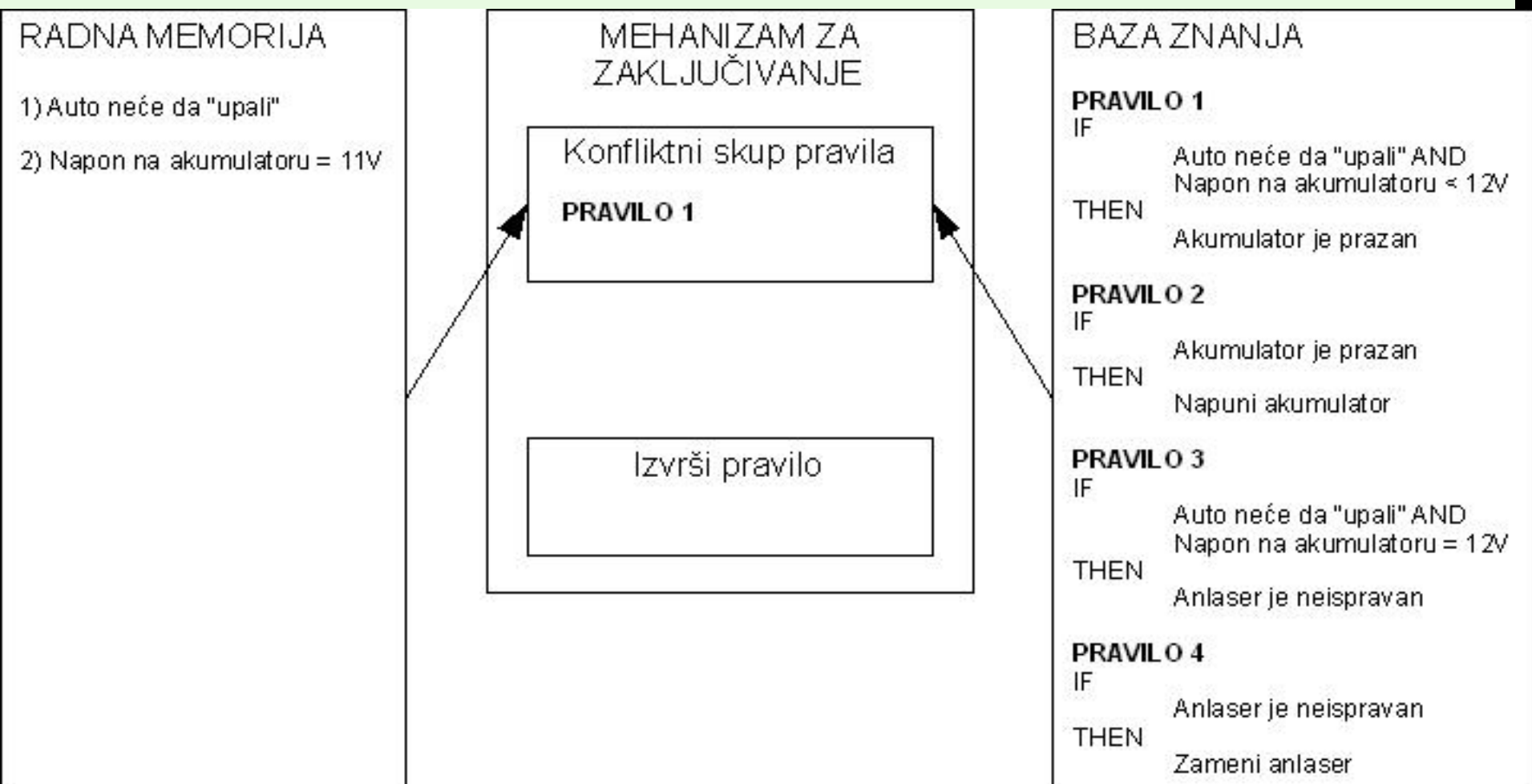
IF
Auto neće da "upali" AND
Napon na akumulatoru = 12V
THEN
Anlaser je neispravan

PRAVILO 4

IF
Anlaser je neispravan
THEN
Zameni anlaser

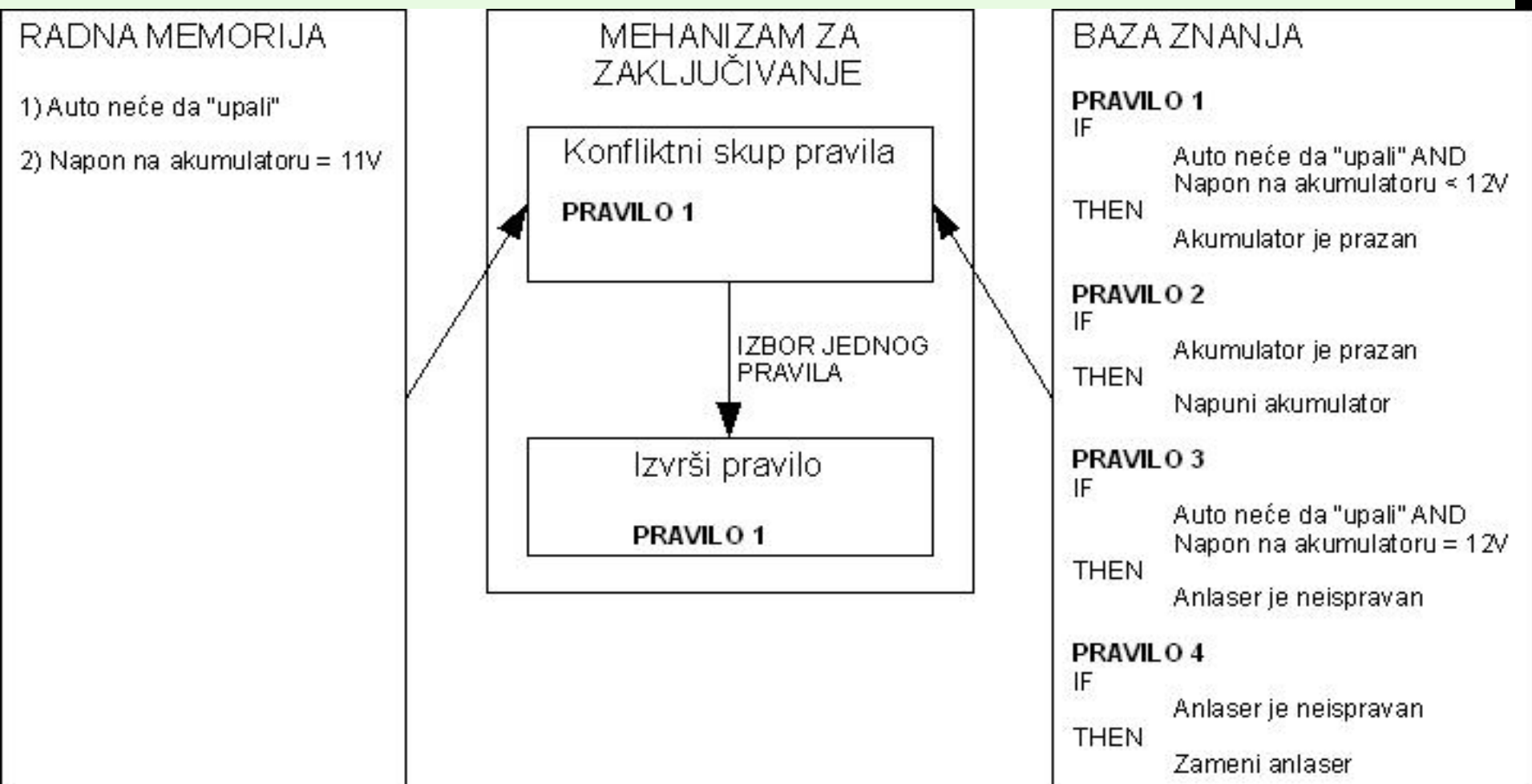
MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer (ciklus 1 korak 1)



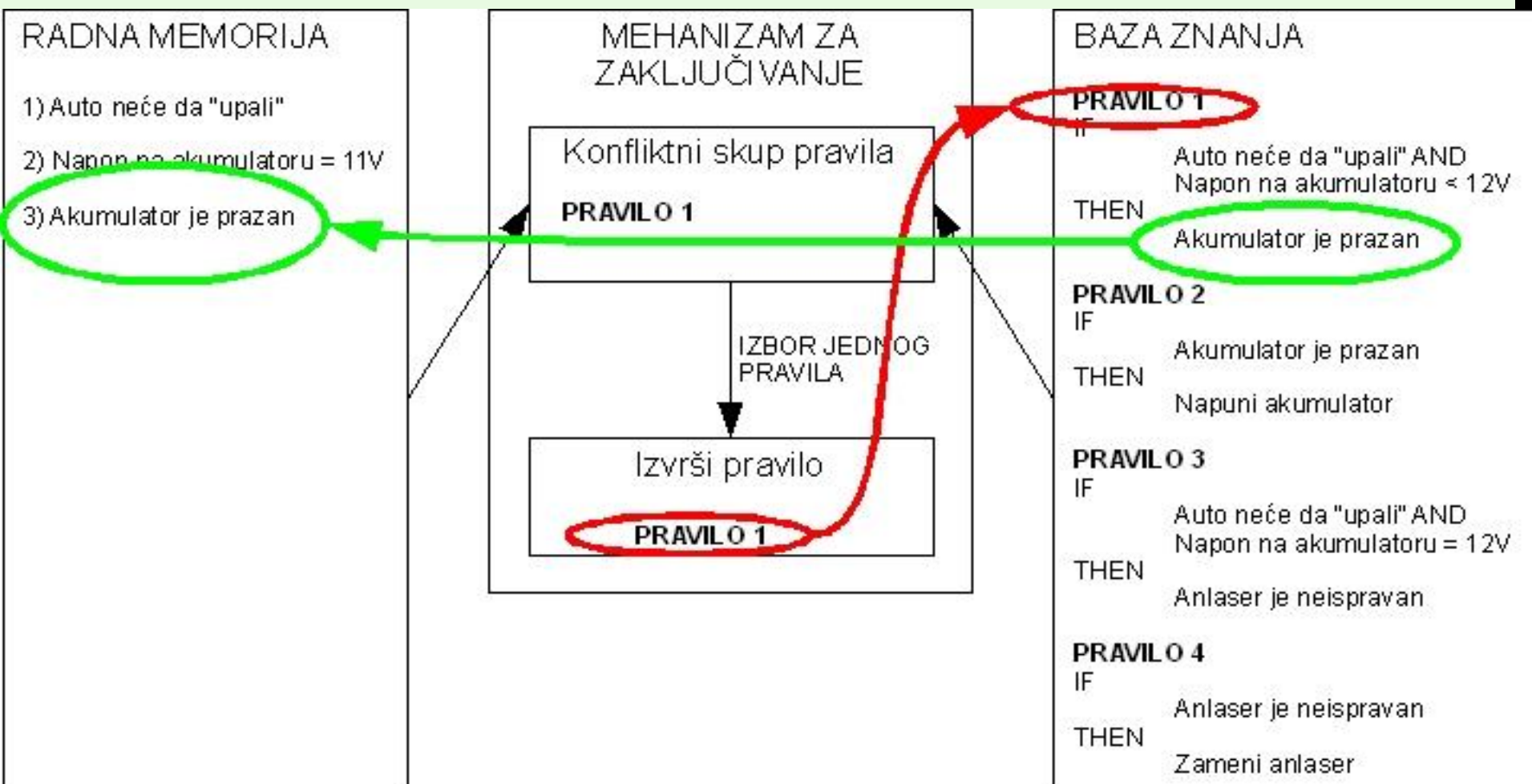
MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer (ciklus 1 korak 2)



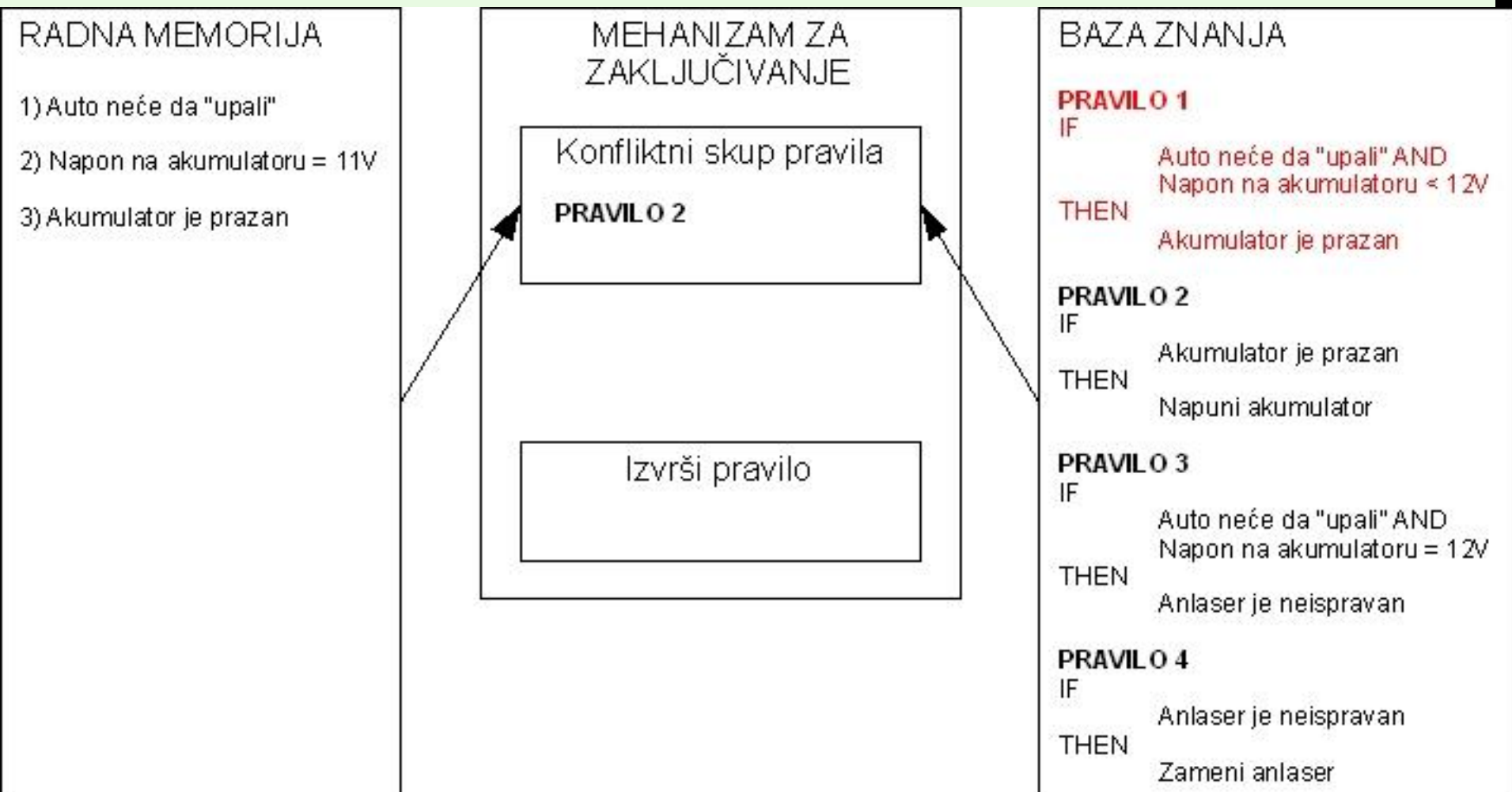
MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer (ciklus 1 korak 3)



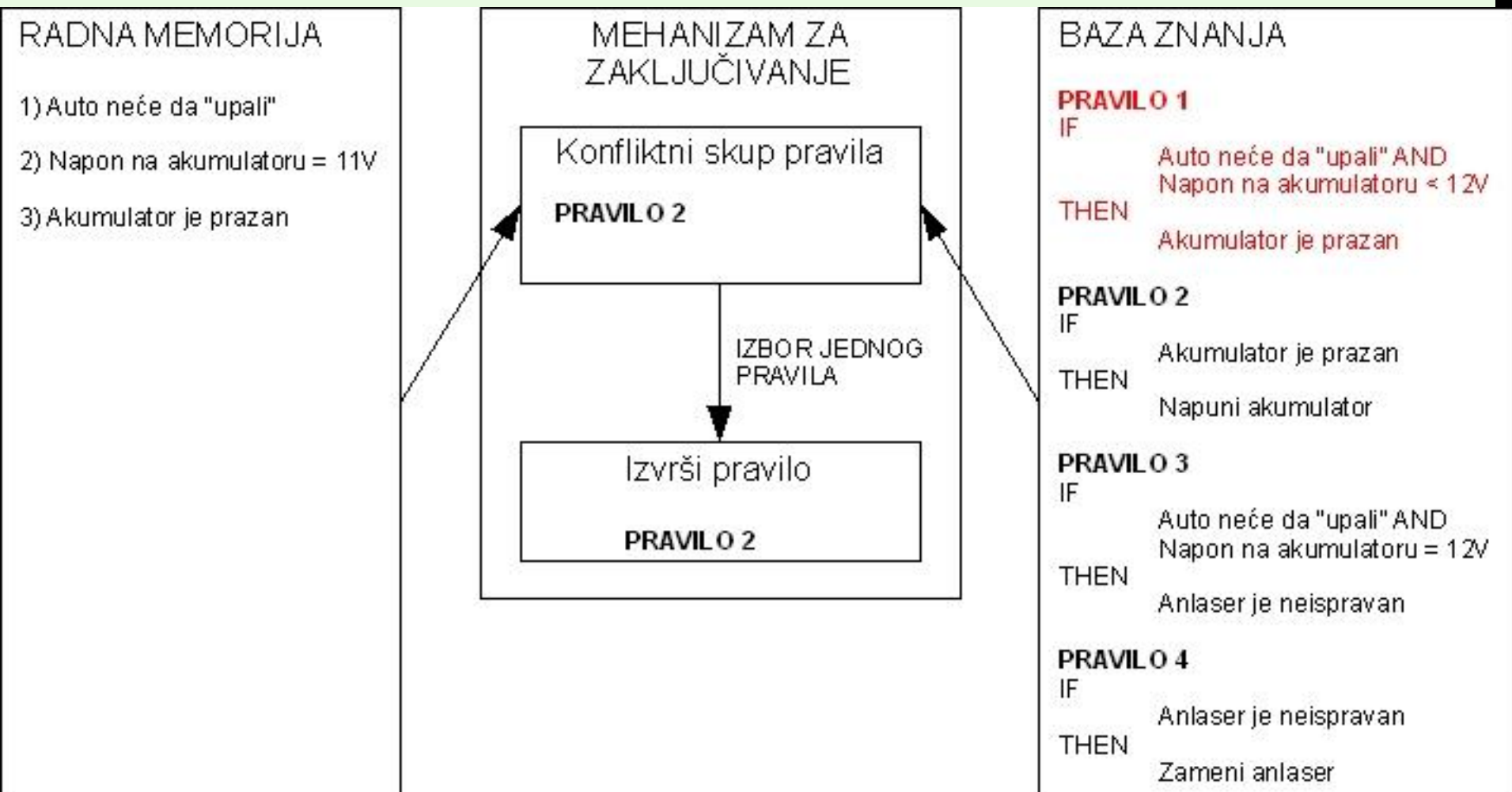
MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer (ciklus 2 korak 1)



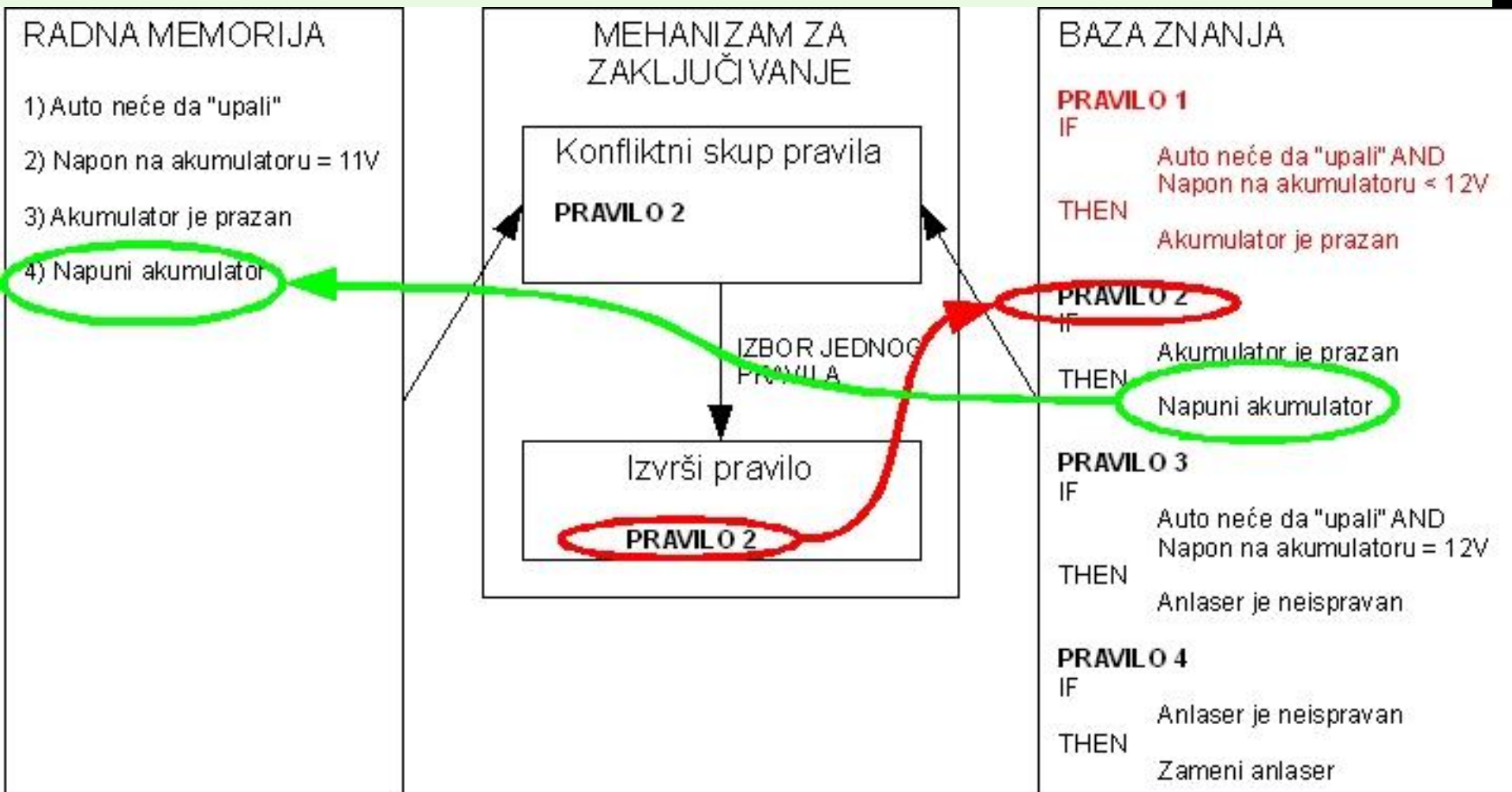
MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer (ciklus 2 korak 2)



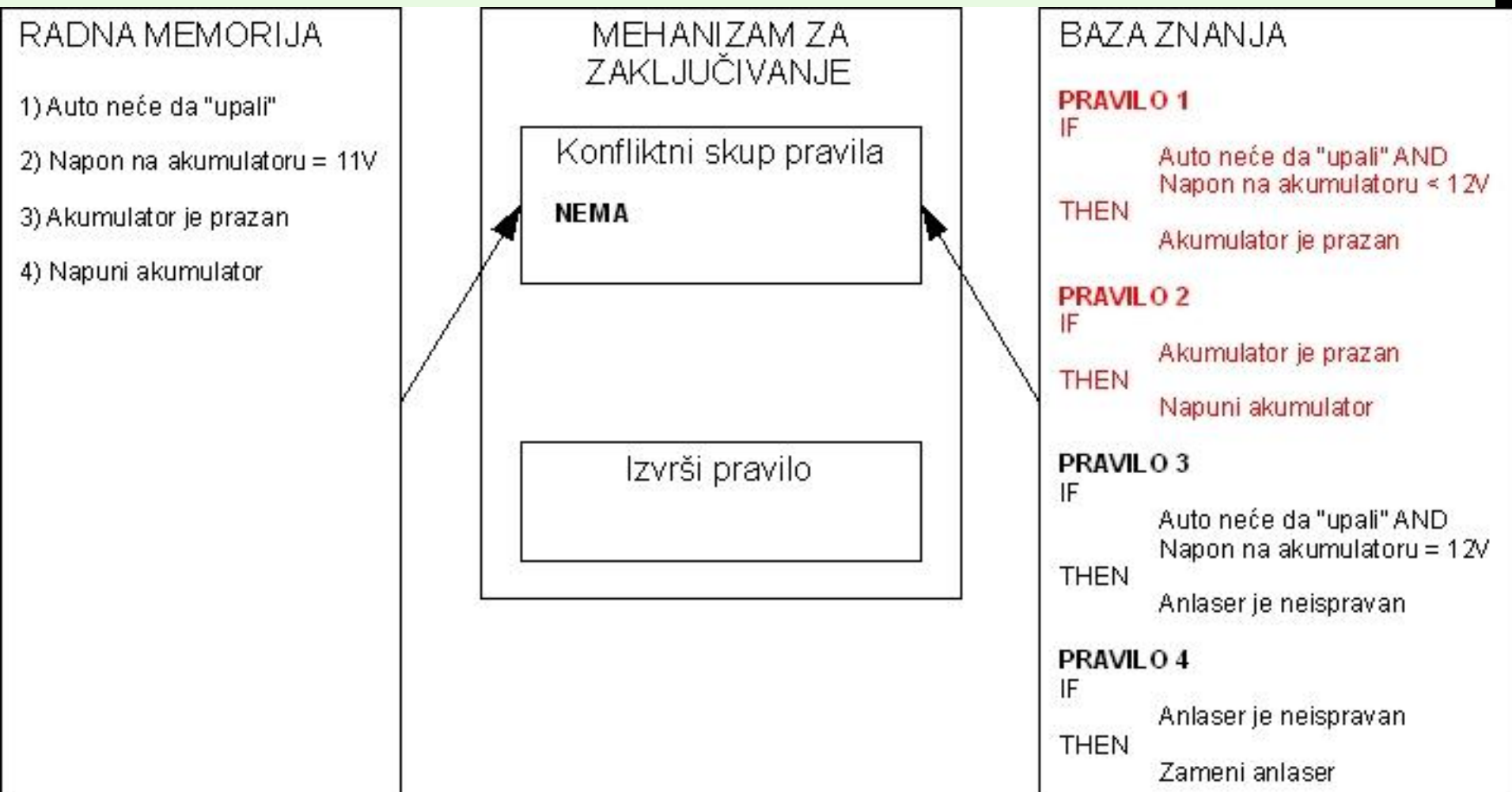
MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer (ciklus 2 korak 3)



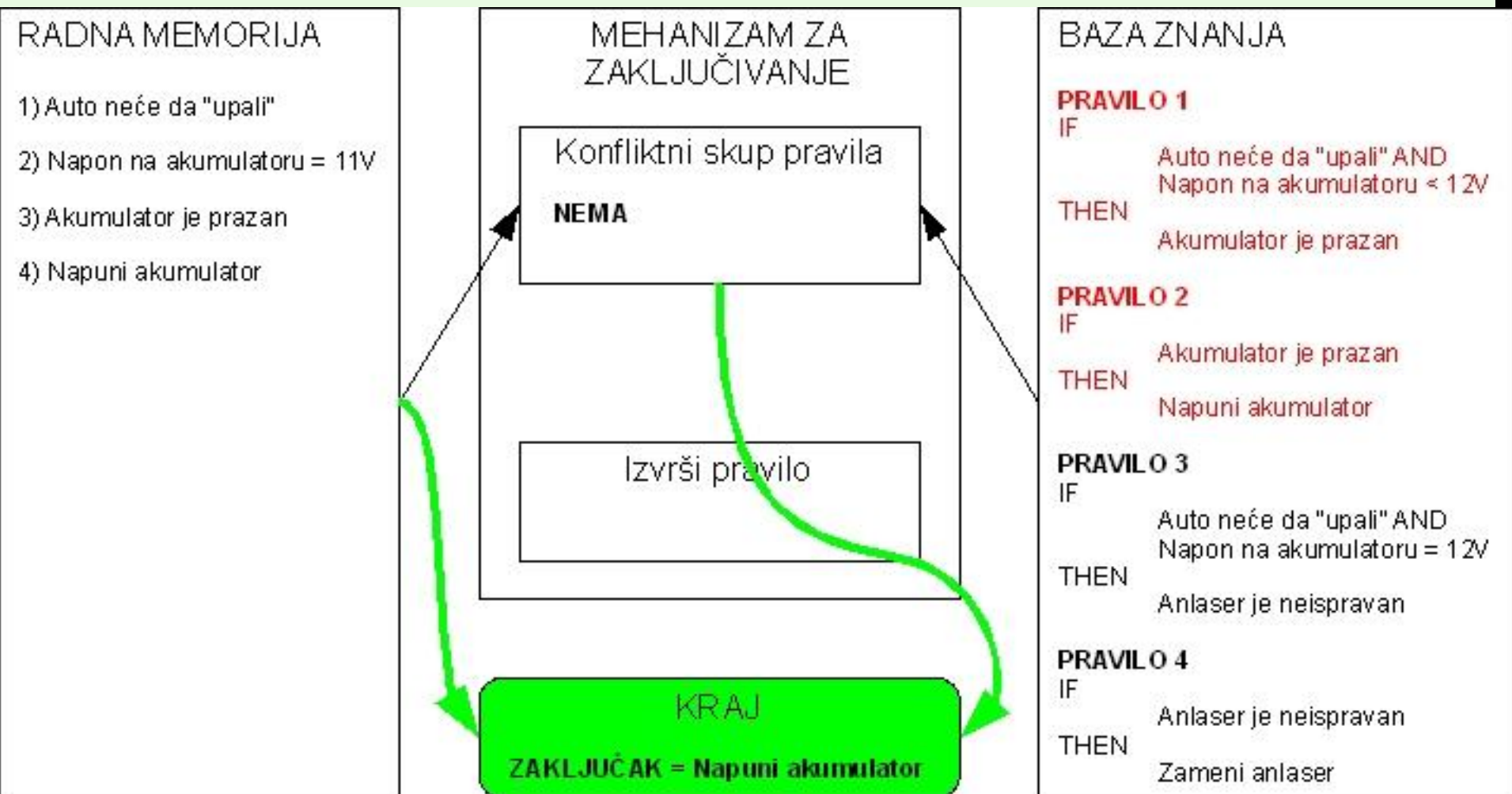
MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer (ciklus 3 korak 1)



MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unapred – primer (kraj)



□ MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad

- Agenda ciljeva – hijerarhijska struktura ciljeva koje je potrebno dokazati da bi se dokazao osnovni cilj
- Ne posmatraju se svi podaci već samo oni koji mogu da pomognu dokazivanju cilja

□ MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad - primer

- Isti skup pravila i ulaznih činjenica kao u prethodnom primeru
- Osnovni cilj koji je potrebno dokazati:

“Napuni akumulator”

MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad – primer (početak)

RADNA MEMORIJA

- 1) Auto neće da "upali"
- 2) Napon na akumulatoru = 11V

MEHANIZAM ZA ZAKLJUČIVANJE

Agenda ciljeva

- Napuni akumulator

BAZA ZNANJA

PRAVILO 1

IF
Auto neće da "upali" AND
Napon na akumulatoru < 12V
THEN
Akumulator je prazan

PRAVILO 2

IF
Akumulator je prazan
THEN
Napuni akumulator

PRAVILO 3

IF
Auto neće da "upali" AND
Napon na akumulatoru = 12V
THEN
Anlaser je neispravan

PRAVILO 4

IF
Anlaser je neispravan
THEN
Zameni anlaser

MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad – primer (ciklus 1 korak 1)



MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad – primer (ciklus 1 korak 2)

RADNA MEMORIJA

- 1) Auto neće da "upali"
- 2) Napon na akumulatoru = 11V

MEHANIZAM ZA ZAKLJUČIVANJE

Agenda ciljeva

- Napuni akumulator (nedokazan)
- Akumulator je prazan

BAZA ZNANJA

PRAVILO 1

IF
Auto neće da "upali" AND
Napon na akumulatoru $< 12V$
THEN
Akumulator je prazan

PRAVILO 2

IF
Akumulator je prazan
THEN
Napuni akumulator

PRAVILO 3

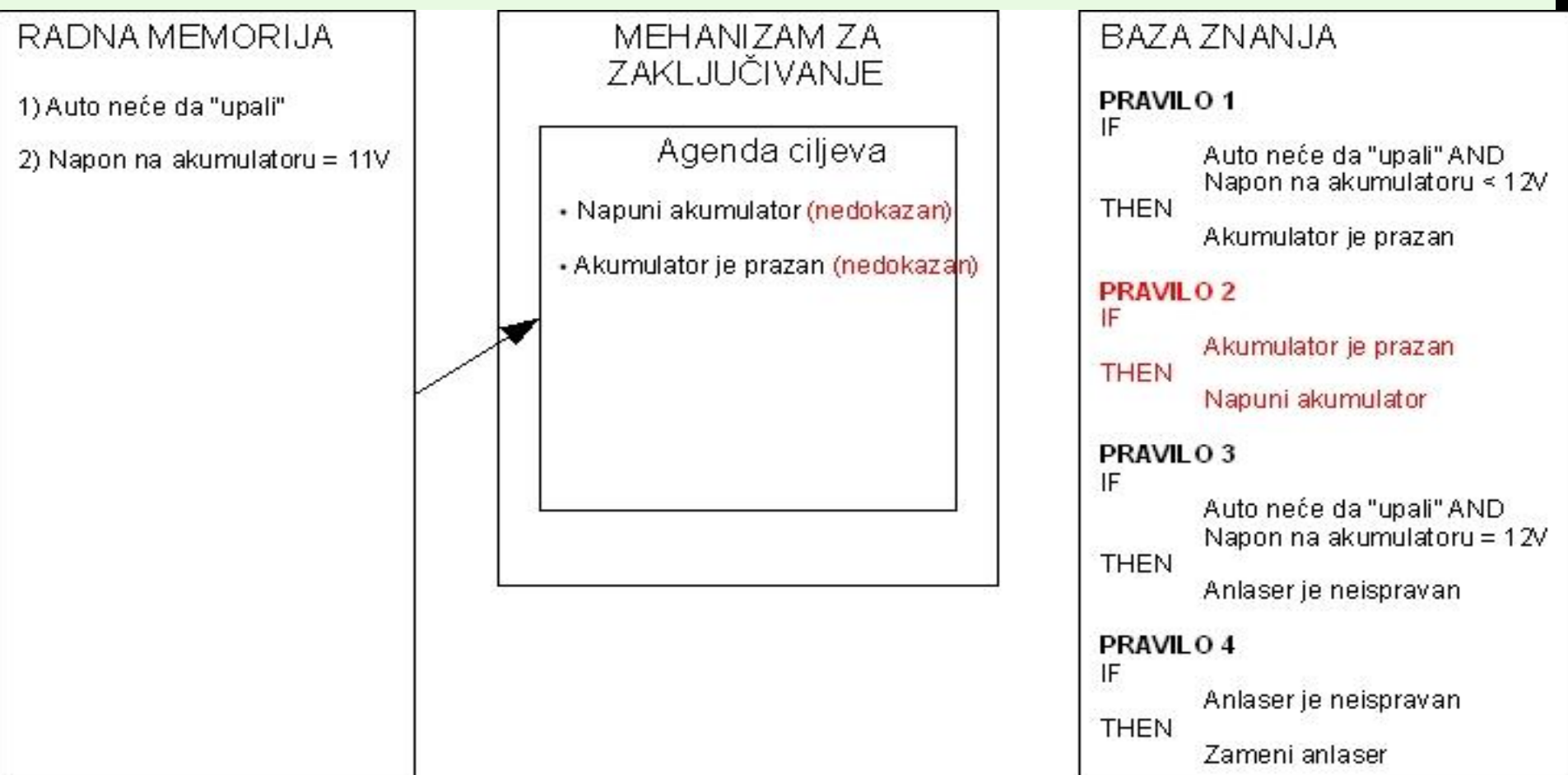
IF
Auto neće da "upali" AND
Napon na akumulatoru = 12V
THEN
Anlaser je неисправan

PRAVILO 4

IF
Anlaser je неисправan
THEN
Zameni anlaser

MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad – primer (ciklus 2 korak 1)



MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad – primer (ciklus 2 korak 2)

RADNA MEMORIJA

- 1) Auto neće da "upali"
- 2) Napon na akumulatoru = 11V

MEHANIZAM ZA ZAKLJUČIVANJE

Agenda ciljeva

- Napuni akumulator (nedokazan)
- Akumulator je prazan (nedokazan)
- Auto neće da upali
- Napon na akumulatoru < 12V

BAZA ZNANJA

PRAVILO 1

IF
Auto neće da "upali" AND
Napon na akumulatoru < 12V
THEN
Akumulator je prazan

PRAVILO 2

IF
Akumulator je prazan
THEN
Napuni akumulator

PRAVILO 3

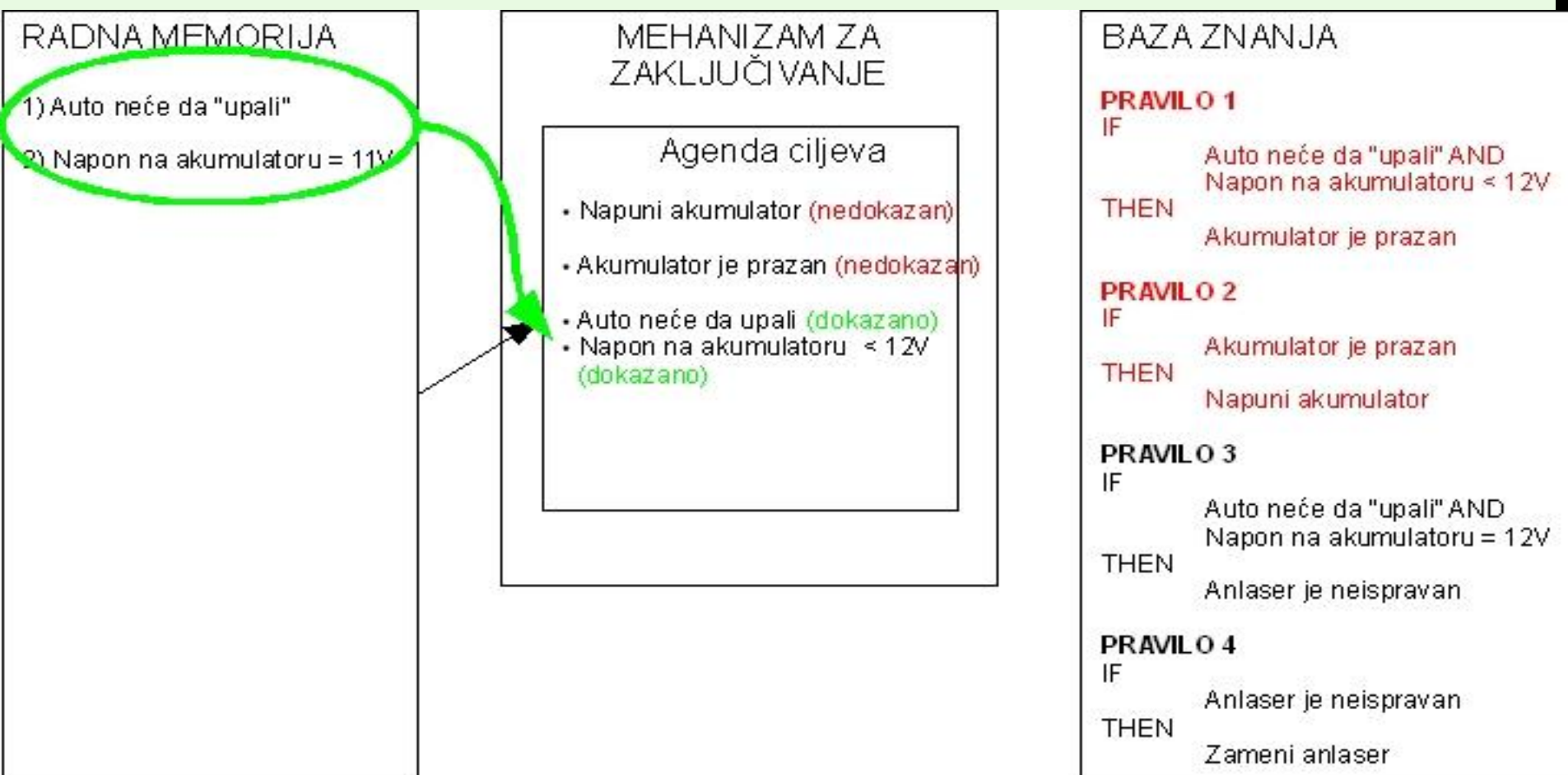
IF
Auto neće da "upali" AND
Napon na akumulatoru = 12V
THEN
Anlaser je неисправan

PRAVILO 4

IF
Anlaser je неисправan
THEN
Zameni anlaser

MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad – primer (ciklus 3 korak 1)



MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad – primer (ciklus 3 korak 1A)

RADNA MEMORIJA

- 1) Auto neće da "upali"
- 2) Napon na akumulatoru = 11V

MEHANIZAM ZA ZAKLJUČIVANJE

Agenda ciljeva

- Napuni akumulator (nedokazan)
- Akumulator je prazan (dokazano)
- Auto neće da upali (dokazano)
- Napon na akumulatoru $< 12V$ (dokazano)

BAZA ZNANJA

PRAVILO 1

IF
Auto neće da "upali" AND
Napon na akumulatoru $< 12V$
THEN
Akumulator je prazan

PRAVILO 2

IF
Akumulator je prazan
THEN
Napuni akumulator

PRAVILO 3

IF
Auto neće da "upali" AND
Napon na akumulatoru = 12V
THEN
Anlaser je неисправan

PRAVILO 4

IF
Anlaser je неисправan
THEN
Zameni anlaser

MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad – primer (ciklus 3 korak 1B)

RADNA MEMORIJA

- 1) Auto neće da "upali"
- 2) Napon na akumulatoru = 11V

MEHANIZAM ZA ZAKLJUČIVANJE

Agenda ciljeva

- Napuni akumulator (dokazano)
- Akumulator je prazan (dokazano)
- Auto neće da upali (dokazano)
- Napon na akumulatoru $< 12V$ (dokazano)

BAZA ZNANJA

PRAVILO 1

IF
Auto neće da "upali" AND
Napon na akumulatoru $< 12V$
THEN
Akumulator je prazan

PRAVILO 2

IF
Akumulator je prazan
THEN
Napuni akumulator

PRAVILO 3

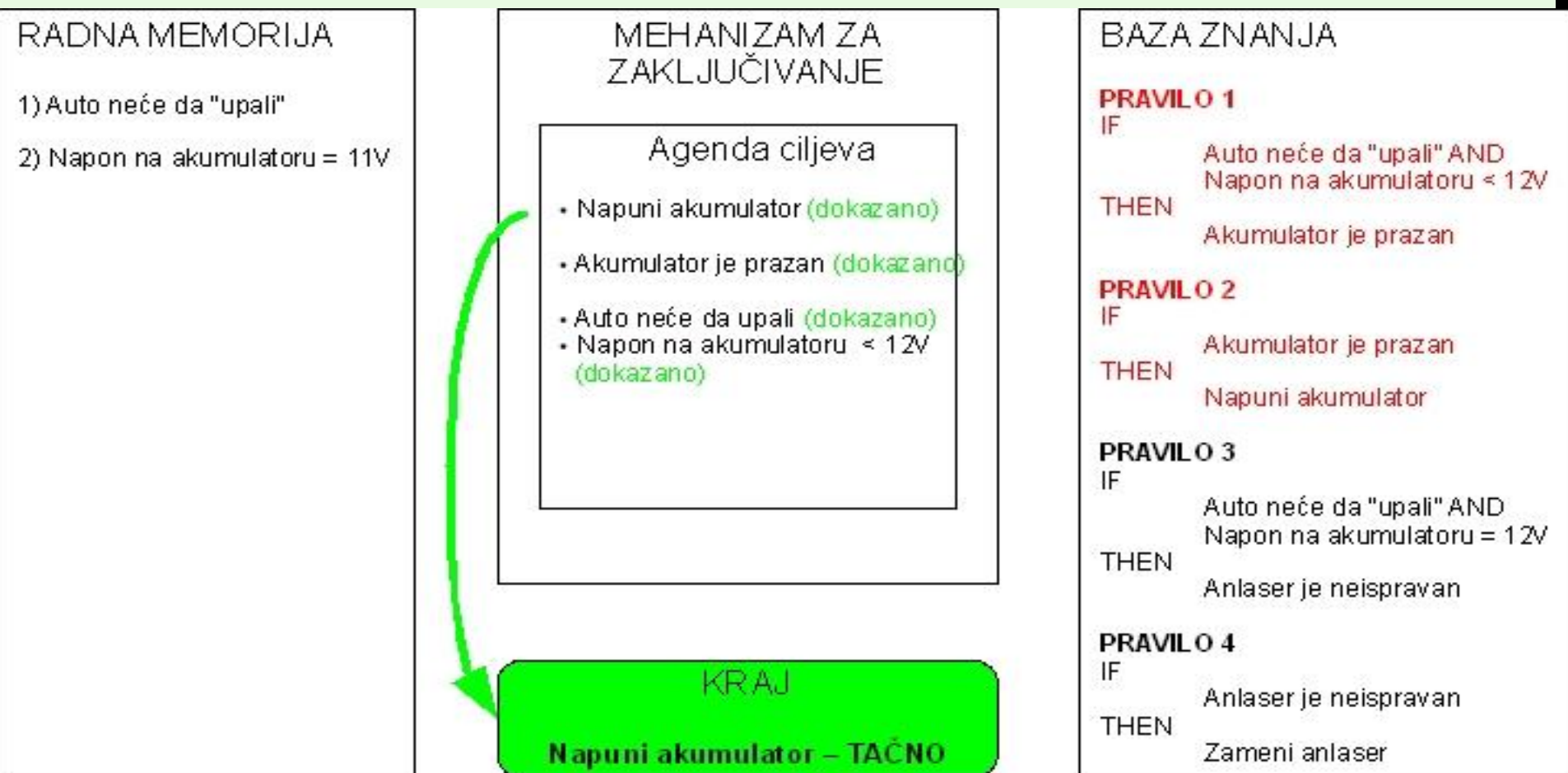
IF
Auto neće da "upali" AND
Napon na akumulatoru = 12V
THEN
Anlaser je неисправan

PRAVILO 4

IF
Anlaser je неисправan
THEN
Zameni anlaser

MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad – primer (kraj)



MEHANIZAM ZA ZAKLJUČIVANJE

Ulančavanje unazad - algoritam



□ MEHANIZAM ZA OBJAŠNJAVANJE

- Formira tri vrste objašnjenja o zaključivanju ES
 - ZAŠTO – objašnjenje o tome zašto ES postavlja određeno pitanje
 - KAKO – objašnjenje o tome kako je ES stigao do rešenja
 - STRATEGIJA – koju je strategiju izabrao ES da bi stigao do rešenja (meta-pravila, heuristike)

MEHANIZAM ZA OBJAŠNJAVANJE

- Dve vrste korisnika objašnjenja ES:
 - Oni koji prave ES (programeri, inženjeri znanja)
 - Oni koji samo koriste ES (krajnji korisnici, eksperti)
- Prvi koriste objašnjenje da bi testirali ili debug-ovali ES
- Drugi koriste objašnjenje da bi se uverili u istinitost zaključaka i stekli uvid u proces zaključivanja

MECHANIZAM ZA OBJAŠNJAVANJE

- Objasnjenje mora da bude prilagođeno korisniku
 - iskustvu
 - nivou znanja
 - rečniku
- Tehnike za formiranje objašnjenja (najčešće korišćene)
 - trag pravila
 - učeaurani tekst
 - prikaz pravila u pseudo-kodu ili na način razumljiv korisniku

□ MEHANIZAM ZA OBJAŠNJAVANJE

- Objašnjenja za one koji prave ES se često definišu u formi liste izvršenih pravila (trag pravila, “rule trace”):
 - Lista naziva izvršenih pravila u redosledu izvršavanja
 - Lista trenutnih činjenica koje su dovele do izvršenja svakog pravila (trenutno stanje radne memorije)
 - Primer:
Pravilo 1 → Pravilo 4 → Pravilo 2 → KRAJ

□ MEHANIZAM ZA OBJAŠNJAVANJE

- Objašnjenja za krajnje korisnike se često definišu u formi teksta predefinisane forme (učaurenog teksta, “templates”, “canned text”):
 - Unapred utvrđene rečenice koje mogu da imaju i dinamičke delove, npr. vrednosti promenljivih
 - Primer:

Ako je napon na akumulatoru <X> što je manje od optimalnih 12V, akumulator je prazan.

□ MEHANIZAM ZA OBJAŠNJAVANJE

Objašnjenje “Zašto” - primer

ES: Da li auto hoće da upali?

Korisnik: NE

ES: Koliki je napon na akumulatoru?

Korisnik: ZAŠTO?

ES: Ako auto neće da upali, a napon na akumulatoru je manji od 12V onda je akumulator prazan i to je problem.

MEHANIZAM ZA OBJAŠNJAVANJE

Objašnjenje "Kako" - primer (početak)

RADNA MEMORIJA

- 1) Auto neće da "upali"
- 2) Napon na akumulatoru = 11V

BAZA ZNANJA

PRAVILO 1

IF

Auto neće da "upali" AND
Napon na akumulatoru $\leq 12V$

THEN

Akumulator je prazan

OBJAŠNJENJE (KAKO?): "Auto neće da upali. Napon na akumulatoru je $<X>$ volti što je manje od optimalnih 12V. Iz toga sledi da je akumulator prazan"

PRAVILO 2

IF

Akumulator je prazan

THEN

Napuni akumulator

OBJAŠNJENJE (KAKO?): "Sa obzirom na to da je akumulator prazan, jedino rešenje je da se akumulator napuni ili zameni"

MEHANIZAM ZA OBJAŠNJAVANJE

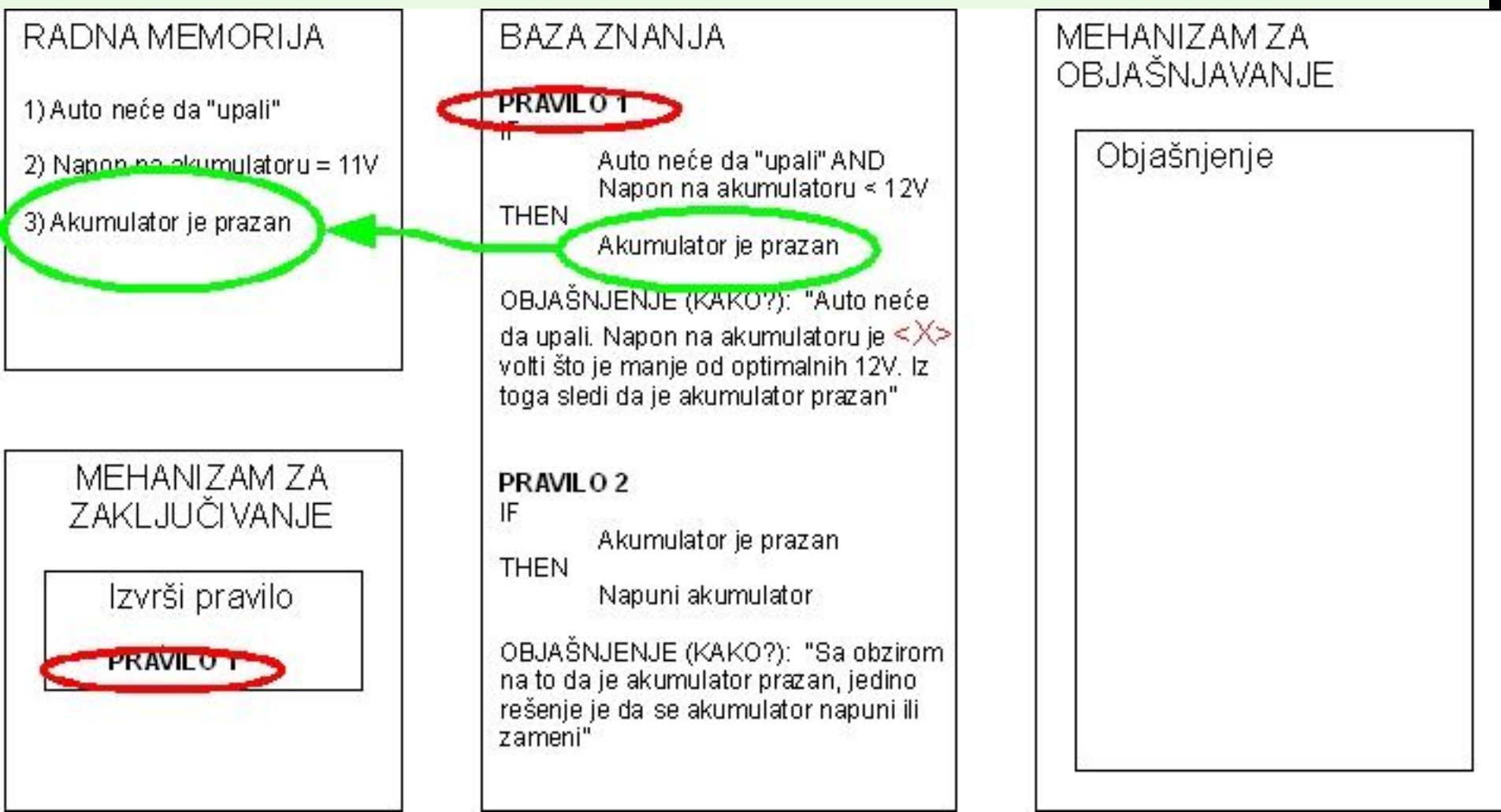
Objašnjenje

MEHANIZAM ZA ZAKLJUČIVANJE

Izvrši pravilo

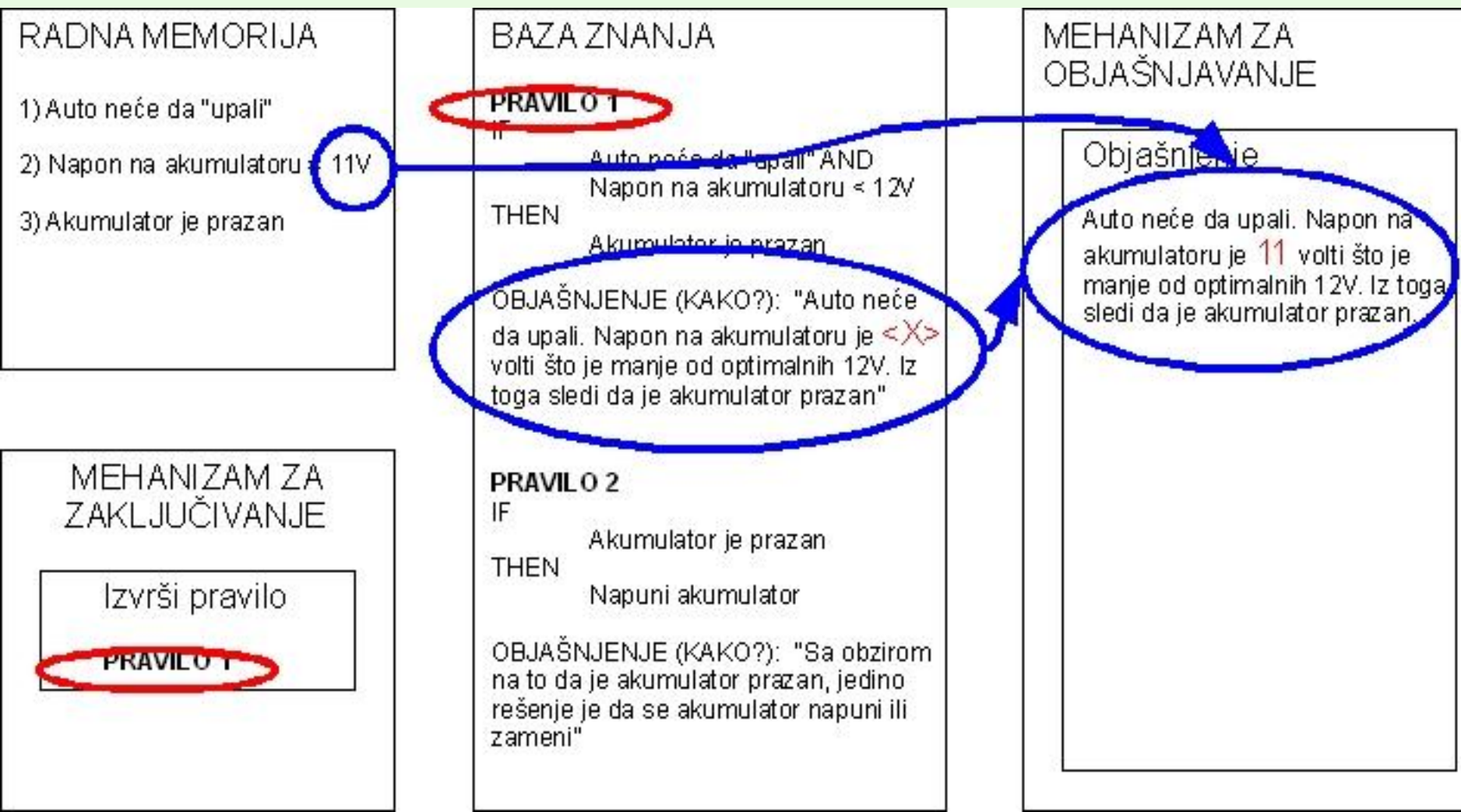
MEHANIZAM ZA OBJAŠNJAVANJE

Objašnjenje "Kako" - primer (korak 1)



MEHANIZAM ZA OBJAŠNJAVANJE

Objašnjenje "Kako" - primer (korak 1A)



MEHANIZAM ZA OBJAŠNJAVANJE

Objašnjenje "Kako" - primer (korak 2)

RADNA MEMORIJA

- 1) Auto neće da "upali"
- 2) Napon na akumulatoru = 11V
- 3) Akumulator je prazan
- 4) Napuni akumulator

BAZA ZNANJA

PRAVILO 1

IF

Auto neće da "upali" AND
Napon na akumulatoru $\leq 12V$

THEN

Akumulator je prazan

OBJAŠNJENJE (KAKO?): "Auto neće da upali. Napon na akumulatoru je $<X>$ volti što je manje od optimalnih 12V. Iz toga sledi da je akumulator prazan"

PRAVILO 2

IF

Akumulator je prazan

THEN

Napuni akumulator

OBJAŠNJENJE (KAKO?): "Sa obzirom na to da je akumulator prazan, jedino rešenje je da se akumulator napuni ili zameni"

MEHANIZAM ZA OBJAŠNJAVANJE

Objašnjenje

Auto neće da upali. Napon na akumulatoru je 11 volti što je manje od optimalnih 12V. Iz toga sledi da je akumulator prazan.

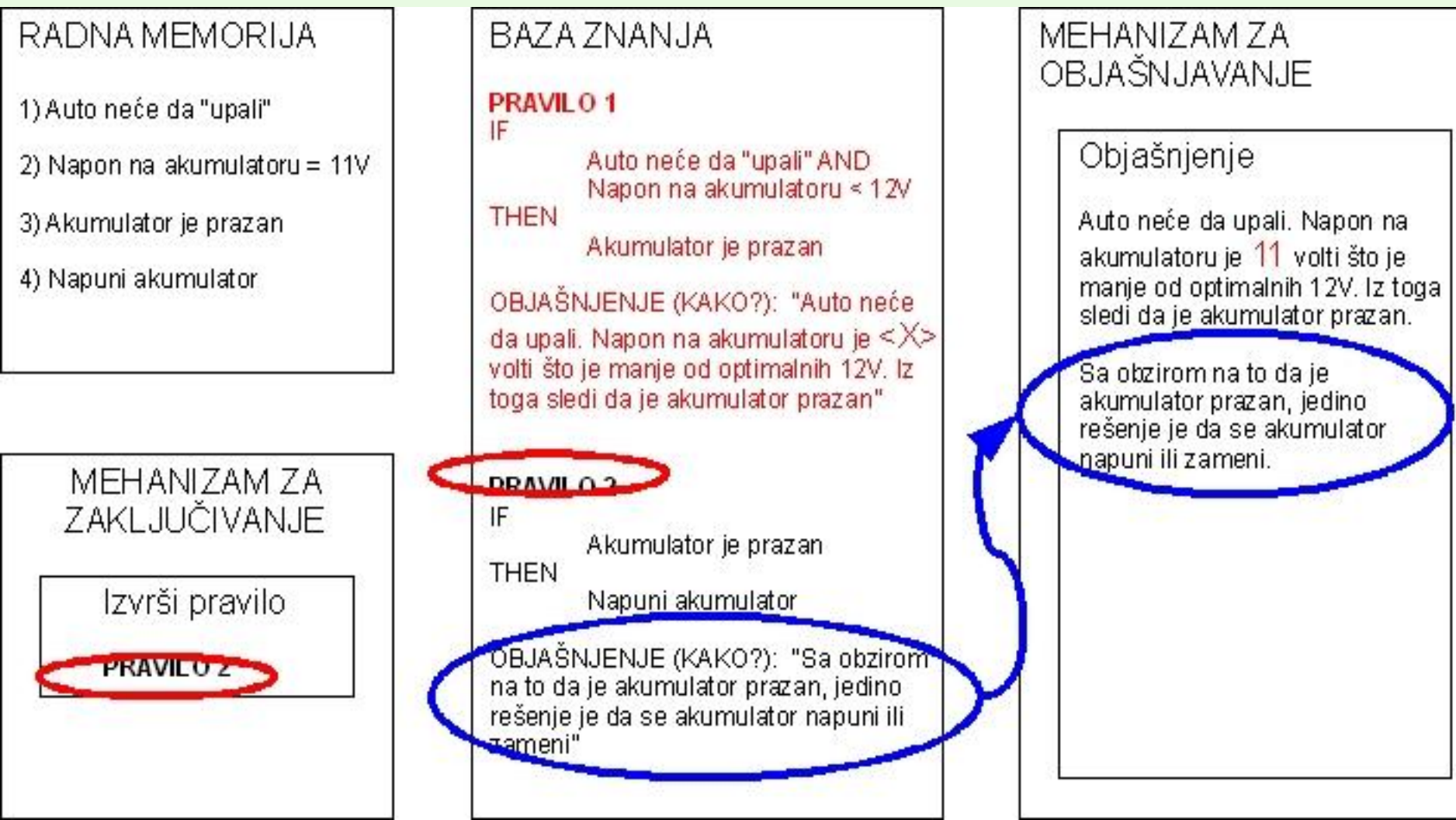
MEHANIZAM ZA ZAKLJUČIVANJE

Izvrši pravilo

PRAVILO 2

MEHANIZAM ZA OBJAŠNJAVANJE

Objašnjenje "Kako" - primer (korak 2A)



MEHANIZAM ZA OBJAŠNJAVANJE

Objašnjenje "Kako" - primer (kraj)

RADNA MEMORIJA

- 1) Auto neće da "upali"
- 2) Napon na akumulatoru = 11V
- 3) Akumulator je prazan
- 4) Napuni akumulator

MEHANIZAM ZA ZAKLJUČIVANJE

Izvrši pravilo

BAZA ZNANJA

PRAVILO 1

IF

Auto neće da "upali" AND
Napon na akumulatoru < 12V

THEN

Akumulator je prazan

OBJAŠNJENJE (KAKO?): "Auto neće da upali. Napon na akumulatoru je <X> volti što je manje od optimalnih 12V. Iz toga sledi da je akumulator prazan"

PRAVILO 2

IF

Akumulator je prazan

THEN

Napuni akumulator

OBJAŠNJENJE (KAKO?): "Sa obzirom na to da je akumulator prazan, jedino rešenje je da se akumulator napuni ili zameni"

MEHANIZAM ZA OBJAŠNJAVANJE

Objašnjenje

Auto neće da upali. Napon na akumulatoru je 11 volti što je manje od optimalnih 12V. Iz toga sledi da je akumulator prazan.

Sa obzirom na to da je akumulator prazan, jedino rešenje je da se akumulator napuni ili zameni.

□ MEHANIZAM ZA OBJAŠNJAVANJE

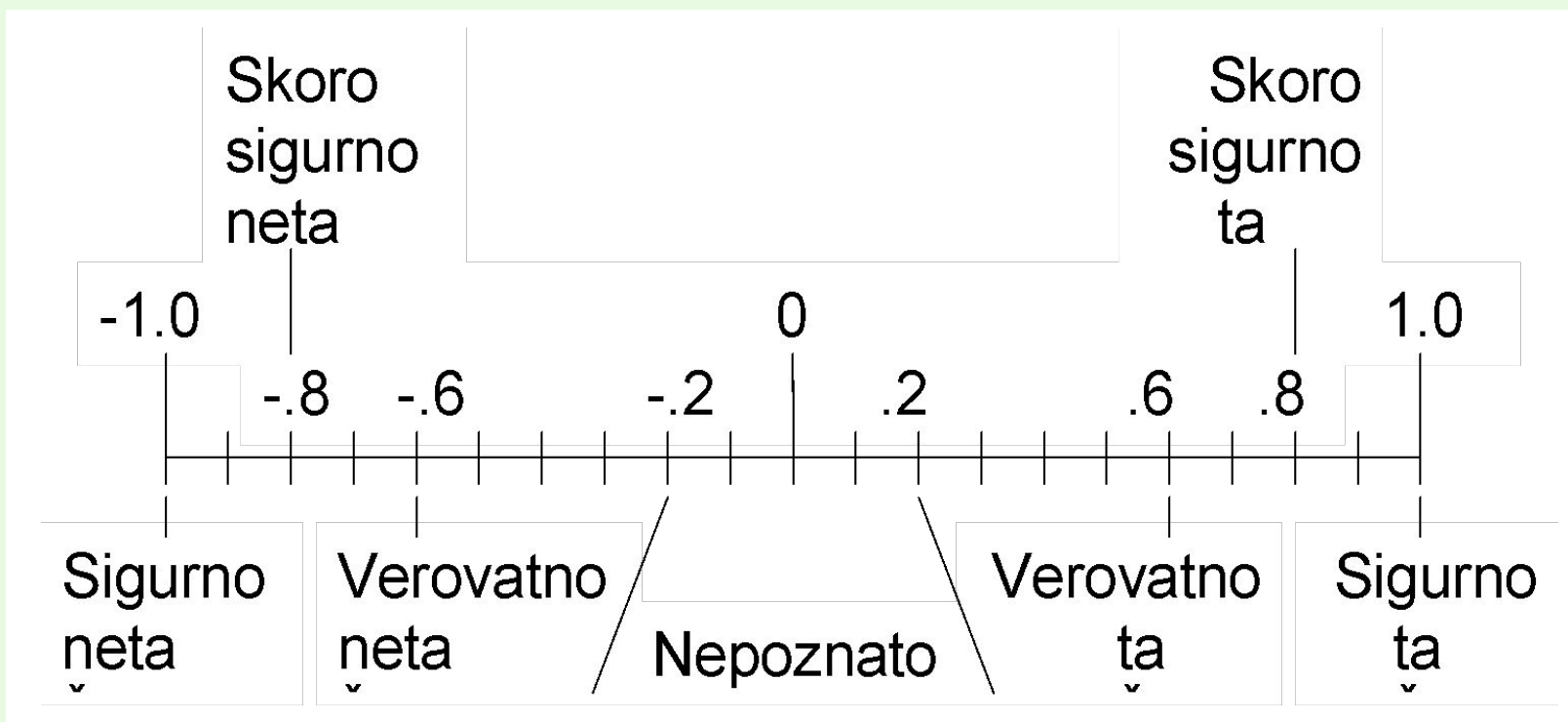
- Objašnjenje „STRATEGIJA“
 - Objašnjavaju se koraci (znanje, meta pravila, meta heuristike) koji su usmerili proces zaključivanja
 - Meta-pravila, zajedno sa grupama pravila omogućavaju optimizaciju procesa zaključivanja fokusiranjem na pravila koja najviše „obećavaju“.
 - Meta-pravila ne vode uvek tačnom rešenju.
- Primer meta-pravila
 - Ako auto neće ni da vergla, usmeriti rešavanje problema na probleme sa električnim sistemom auta

▣ PREDSTAVLJANJE ZNANJA

- Neizvesne činjenice
 - stepen poverenja u tačnost pojedinih činjenica
 - kolokvijalni izrazi kao "možda", "veoma", "donekle",...
 - faktor izvesnosti -
numerička vrednost stepena poverenja
 - ideja o faktorima izvesnosti
prvi put je primenjena u sistemu MYCIN

▣ PREDSTAVLJANJE ZNANJA

- Neizvesne činjenice



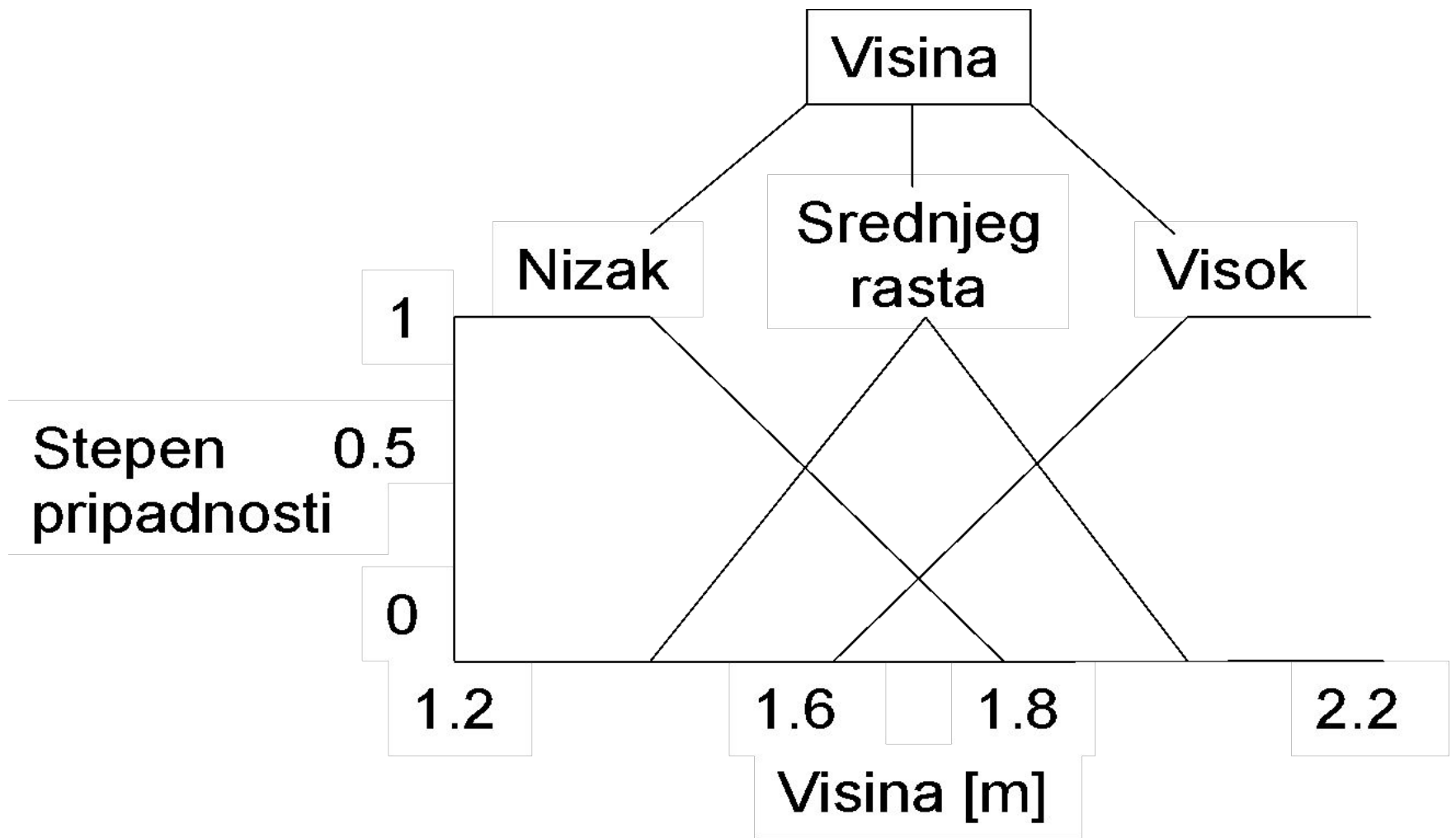
▣ PREDSTAVLJANJE ZNANJA

- Pravila sa faktorom izvesnosti
 - IF Starost < 25 AND
Položen-vozački-ispit = False
THEN Krivac-u-nesreći = Yes (CF = 0.2)

▣ PREDSTAVLJANJE ZNANJA

- Fuzzy činjenice
 - pogodne za predstavljanje određenih izraza iz prirodnog jezika
 - izrazi koji u sebi nose dvosmislenost, neodređenost, nepreciznost
 - fuzzy skupovi
 - kvantitativna analogija neodređenih izraza
 - stepen pripadnosti
 - nivo poverenja da vrednost neke veličine pripada nekom fuzzy skupu

PREDSTAVLJANJE ZNANJA



▣ PREDSTAVLJANJE ZNANJA

- Fuzzy pravila
 - sadrže fuzzy skupove i u IF-delu i u THEN-delu
 - vrše preslikavanje fuzzy skupova iz jednih u druge
 - primer:
Ako (IF) Temperatura je normalna
Onda (THEN) Brzina je srednja

□ BITNE KARAKTERISTIKE ES

- Ograničenost na rešive probleme
 - ako problem ne može da reši ekspert, najverovatnije neće moći da ga reši ni ES
 - ne treba koristiti ES za nove probleme
- Uzana oblast ekspertize
 - slabe performanse izvan te oblasti

□ BITNE KARAKTERISTIKE ES

- Neegzaktno rezonovanje
 - rezonovanje sa neizvesnim, dvosmislenim ili nedostupnim podacima
 - ekspertsko znanje je samo po sebi neegzaktno
- Heurističko rezonovanje
 - nepisana pravila (rules of thumb)
 - algoritmi vs. heuristike
- ES prave i greške

□ ZAŠTO RAZVIJATI ES ?

- Poređenje eksperta i ES

| <i>Faktor</i> | <i>Ekspert</i> | <i>ES</i> |
|----------------------|-----------------------|------------------------------|
| Raspoloživost | Radnim danom | Uvek |
| Geografski | Lokalno | Bilo gde |
| Sigurnost | Nezamenljiv | Zamenljiv |
| Nestalnost | Da | Ne |
| Performanse | Promenljive | Konzistentne |
| Brzina | Promenjliva | Konzistentna (obično i veća) |
| Cena | Visoka | Prihvatljiva |

ZAŠTO RAZVIJATI ES

- Razlozi za razvoj ES kao **zamene za eksperta**
 - Potreba za ekspertizom van radnog vremena i na drugom mestu
 - Potreba za ekspertizom u nepristupačnom okruženju
 - Automatizacija rutinskih poslova koji zahtevaju eksperta
 - Ekspert odlazi u penziju ili napušta kompaniju
 - Ekspert je skup

ZAŠTO RAZVIJATI ES

- Razlozi za razvoj ES kao pomoći za eksperta
 - Povećanje produktivnosti eksperta u rutinskim poslovima
 - Olakšavanje rada eksperta u rešavanju složenih zadataka
 - Omogućavanje ekspertu da se lakše priseti nekih stvari

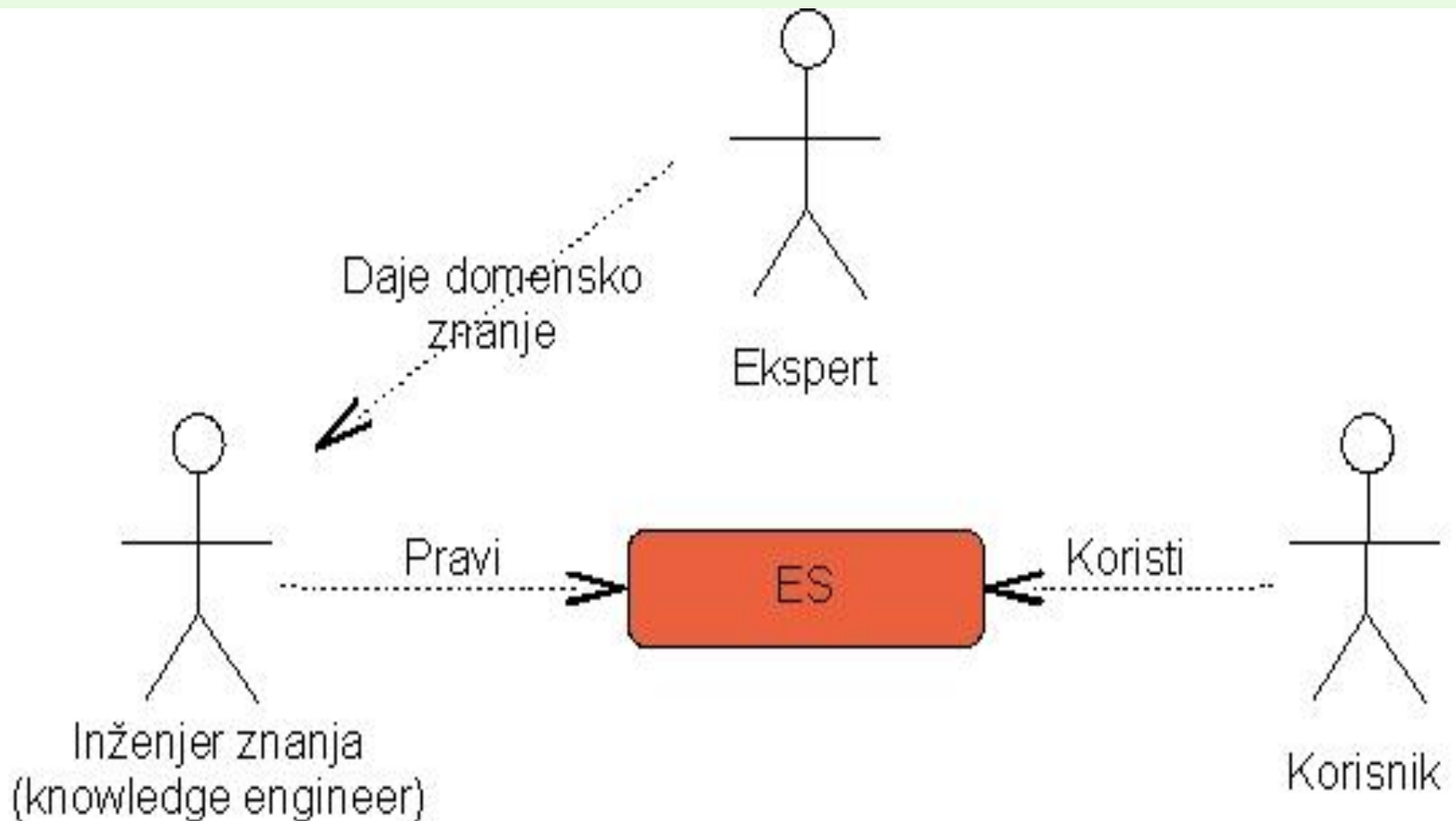
PRIMENE ES

- ES su veoma dobro izučena oblast
- Nova naučna dostignuća su najčešće novi vidovi primene ES (i tehnologija ES)
- Pojam “ekspertnog sistema” se skoro uopšte više ne koristi, ali su tehnologije ES široko rasprostranjene pod drugim imenom.
 - BRE (Business Rule Engine)
 - BRMS (Business Rule Management System)
 - RBS (Rule-Based System)
 - Negde i Recommender System

PRIMENE ES

- Oblasti gde se tehnologija ES koristi „ispod haube“
 - Auto industrija (dijagnoza kvara na vozilima)
 - Praćenje i nadzor vozila, objekata
 - Računarske mreže i zaštita
 - Baze podataka
 - Poslovna pravila (“Business Rules”)
 - Programiranje u okviru ograničenja („Constraint based programming“)

KLJUČNE ULOGE U RAZVOJU ES



KLJUČNE ULOGE U RAZVOJU ES

- Ekspert
 - “Pozajmljuje” tj. daje svoje znanje
 - Pomaže pri proveru (testiranju) znanja
 - Problemi ako je:
 - Nedostupan
 - Nekomunikativan
 - Sklon tome da ističe očigledno
 - Zaboravan – ne može da se seti svega

KLJUČNE ULOGE U RAZVOJU ES

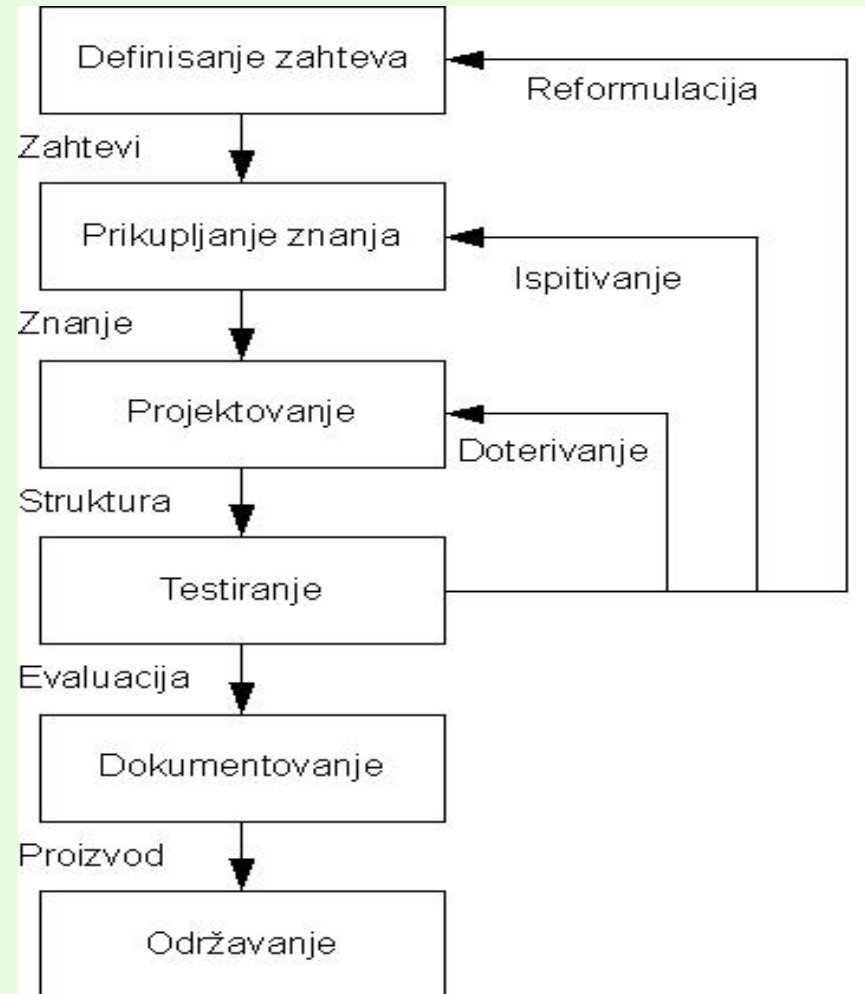
- Inženjer znanja
 - Vodi intervju sa ekspertom i iz njega “izvlači” znanje
 - Vrš izbor odgovarajućih tehnika za predstavljanje znanja
 - Vrš izbor odgovarajućih tehnika za zaključivanje
 - Vrš izbor razvojnog alata
 - Formalizuje, formuliše i “sređuje” ekspertovo znanje
 - Testira ES

KLJUČNE ULOGE U RAZVOJU ES

- Korisnik
 - Koristi gotov ES
 - Učestvuje u formiranju zahteva
 - Može da učestvuje u testiranju i pisanju dokumentacije za ES

PROCES RAZVOJA ES

- Ekspert
 - Prikupljanje znanja (ključna faza)
 - Testiranje
- Inženjer znanja
 - Sve faze
- Korisnik
 - Definisanje zahteva
 - Testiranje
 - Dokumentovanje

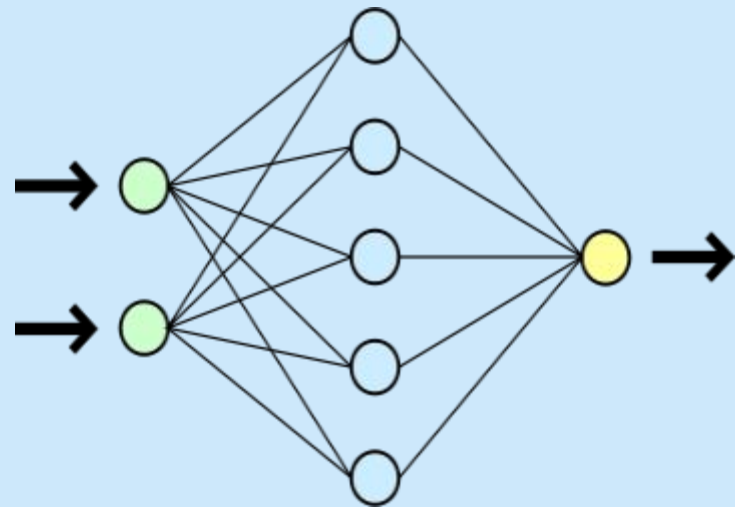
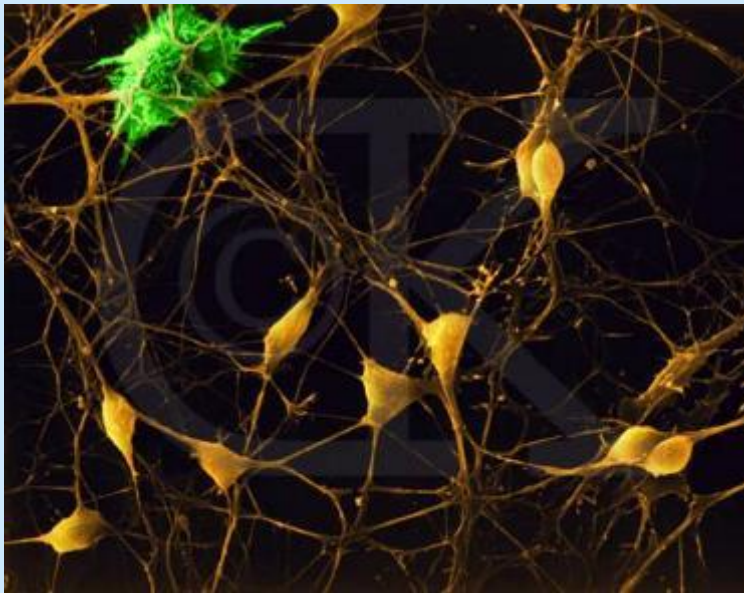


5.

VEŠTAČKE NEURONSKE MREŽE

ŠTA SU NEURONSKE MREŽE

- Jedna od tehnika mašinskog učenja
- Matematički modeli u obliku grafa inspirisani principima na kojima funkcioniše mozak



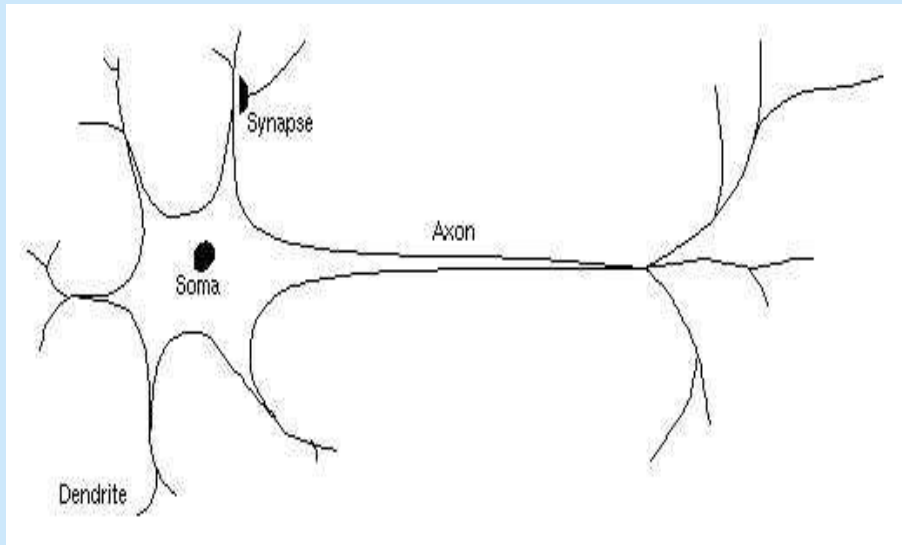
ŠTA JE VEŠTAČKA NEURONSKA MREŽA - DEFINICIJE

- **DARPA:** Neuronska mreža je sistem koji se sastoji od velikog broja međusobno **povezanih, jednostavnih elemenata procesiranja** koji rade paralelno.
Funkcija VNM je određena strukturom mreže, težinom veza, i obradom u elementima procesiranja.
- **Haykin:** Neuronska mreža je paralelni distribuirani procesor koji ima prirodnu sposobnost **čuvanja i korišćenja iskustvenog znanja**.
Sličnost sa mozgom se ogleda kroz dve osobine:
 - mreža stiče znanje kroz proces učenja
 - znanje se čuva u vezama između neurona (sinaptičkim težinama)

ZAŠTO NEURONSKE MREŽE?

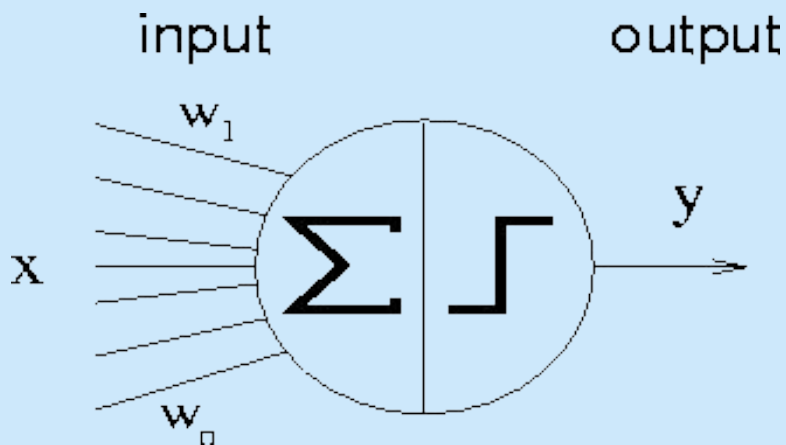
- Omogućavaju napredu obradu podataka bez prethodno definisanog modela ili algoritma već na osnovu podataka o ponašanju nekog Sistema
- Efikasno mogu da rešavaju veoma složene probleme koji bi inače bili teško rešivi nekim algoritamskim postupkom.
- Primenjive su na širok spektar problema (klasifikacija, regresija, klasterizacija, slike, tekst)

BIOLOŠKI I VEŠTAČKI NEURON



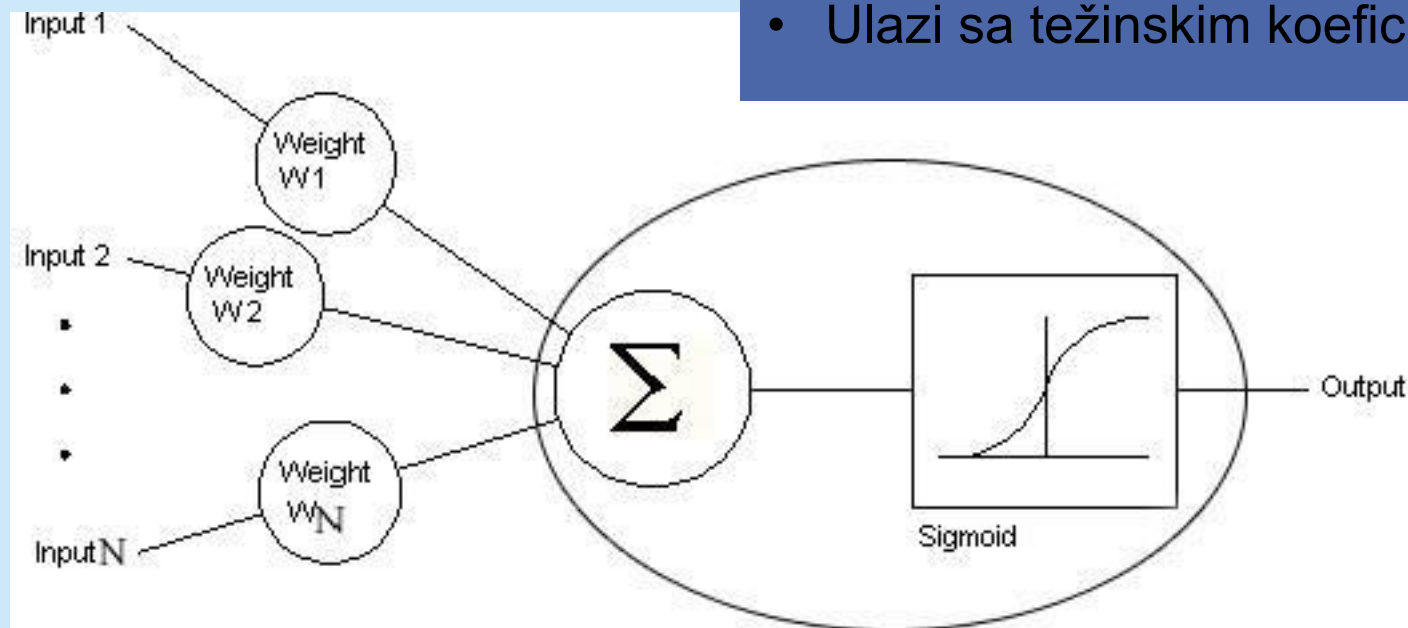
Osnovni delovi:

- telo(soma)
- dendriti(ulazi)
- akson(izlaz)
- sinapse(spojevi)



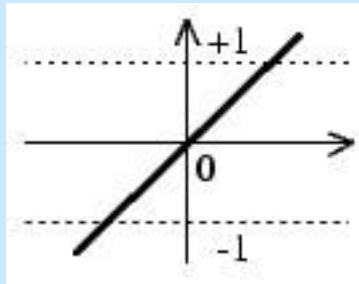
OSNOVNI MODEL VEŠTAČKOG NEURONA

- Ulazna funkcija sumiranja
- Funkcija transfera (ili aktivacije)
- Ulazi sa težinskim koeficijentima

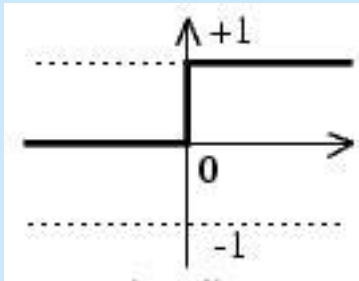


$$\text{output} = f(w_1 \text{in}_1 + \dots + w_n \text{in}_n)$$

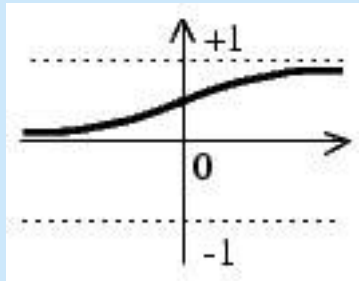
FUNKCIJE TRANSFERA



Linearna funkcija

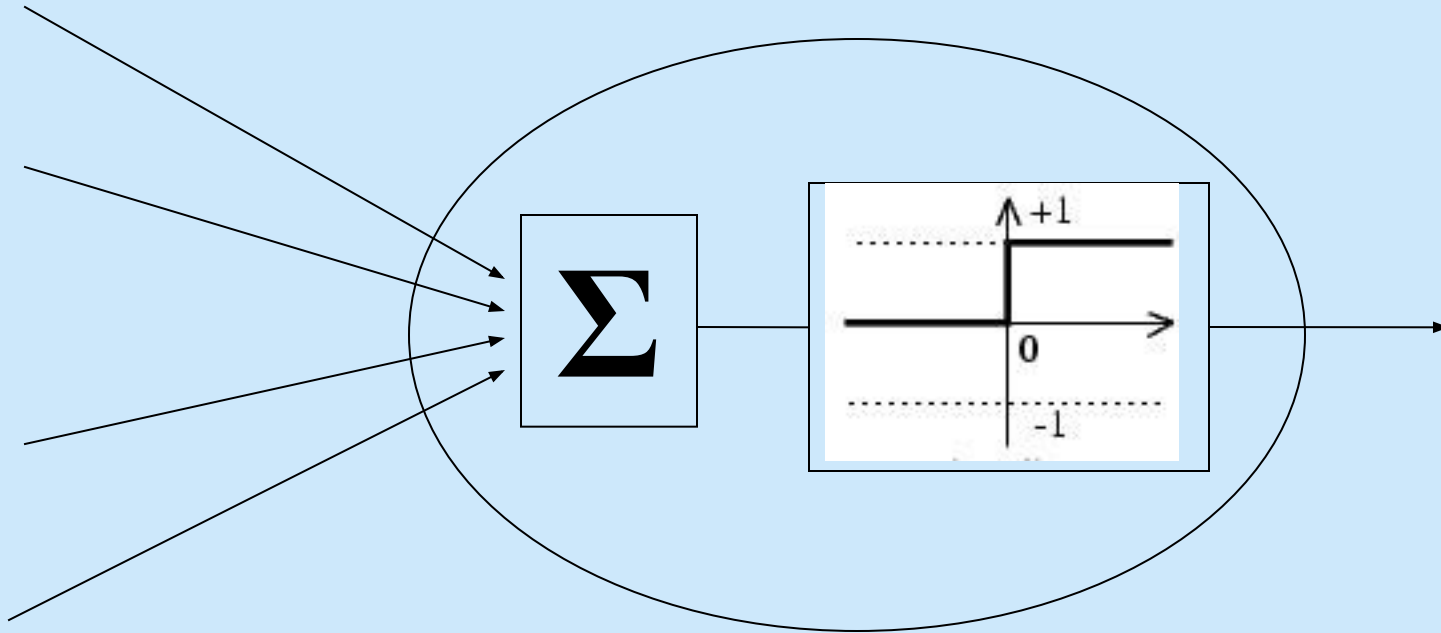


Hevisajdova funkcija (step)



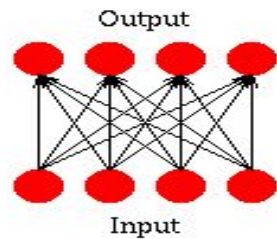
Sigmoidna funkcija

McCULLOCH PITS NEURON THRESHOLD LOGIC UNIT

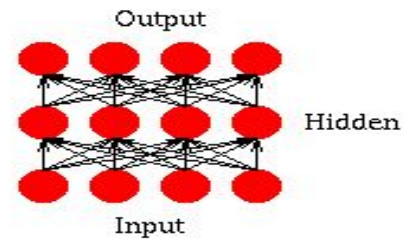


$$y = \text{STEP} (w_1 u_1 + \dots + w_n u_n)$$

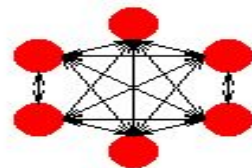
ARHITEKTURE VNM



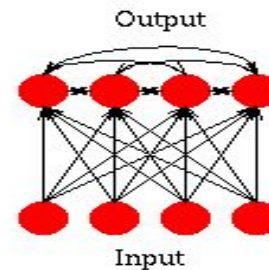
Single Layer Feedforward



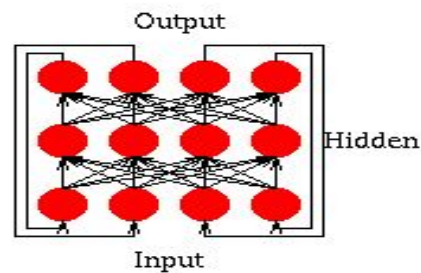
Multi Layer Feedforward



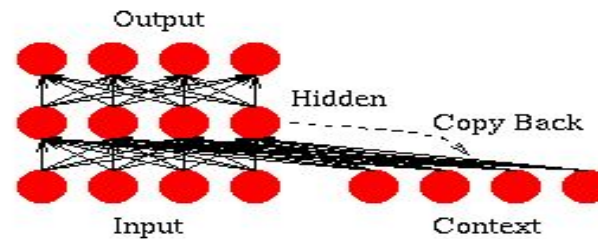
Fully Recurrent Network



Competitive Network



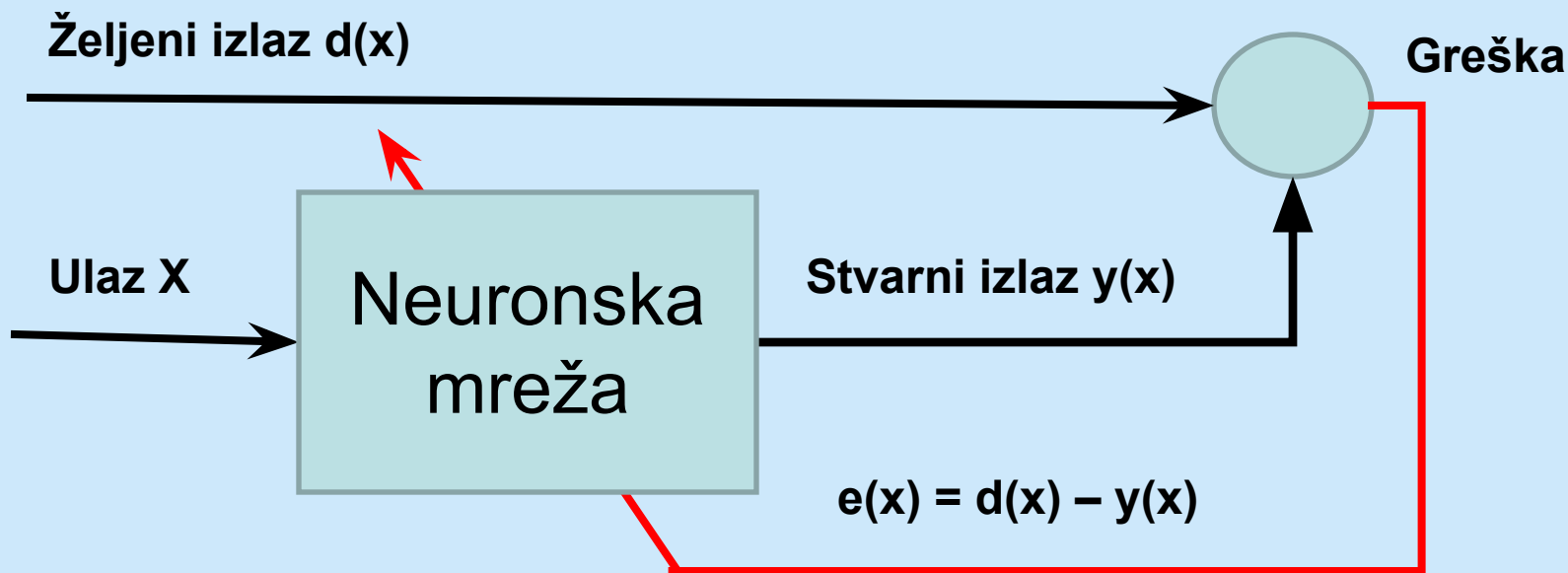
Jordan Network



Simple Recurrent Network

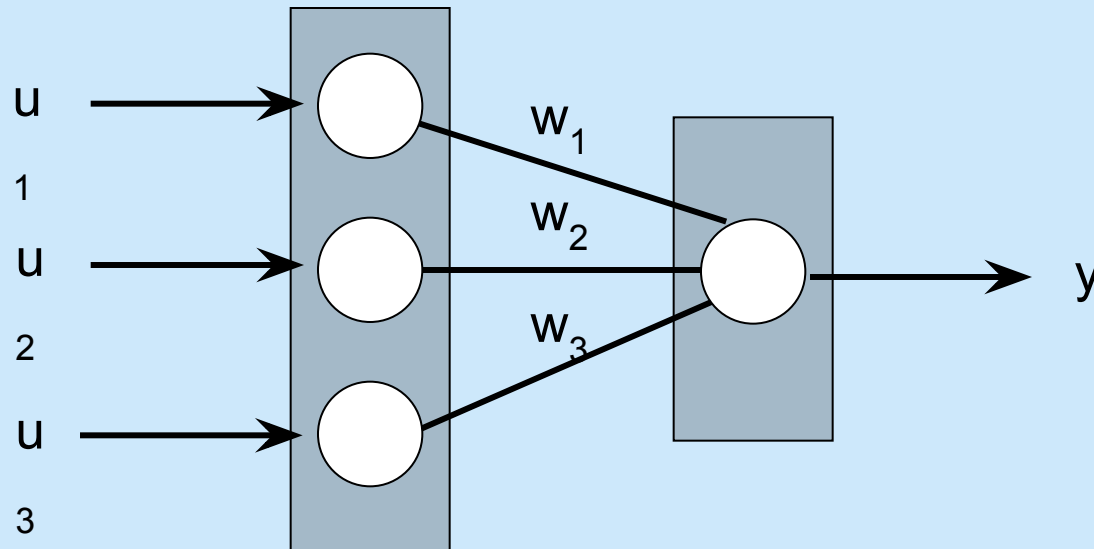
ALGORITAM ZA UČENJE SA NADGLEDANJEM

Opšti princip: minimizacija ukupne greške kroz iterativnu proceduru



$$w_{ji}(k+1) = w_{ji}(k) + \Delta w_{ji}(k) = w_{ji}(k) + \mu E(y_j(k), d_j(k))$$

ADALINE



Linearna funkcija transfera -

Linearna kombinacija ulaza

$$y = w_1 u_1 + w_2 u_2 + \dots + w_n u_n,$$

Učenje metodom najmanjih kvadrata – u suštini linearna regresija

LMS UČENJE

LMS pravilo se može izraziti kroz sledeće jednačine:

(1) greška izlaznog neurona za p -ti uzorak iz skupa za trening

$$\varepsilon_p = y_p - d_p$$

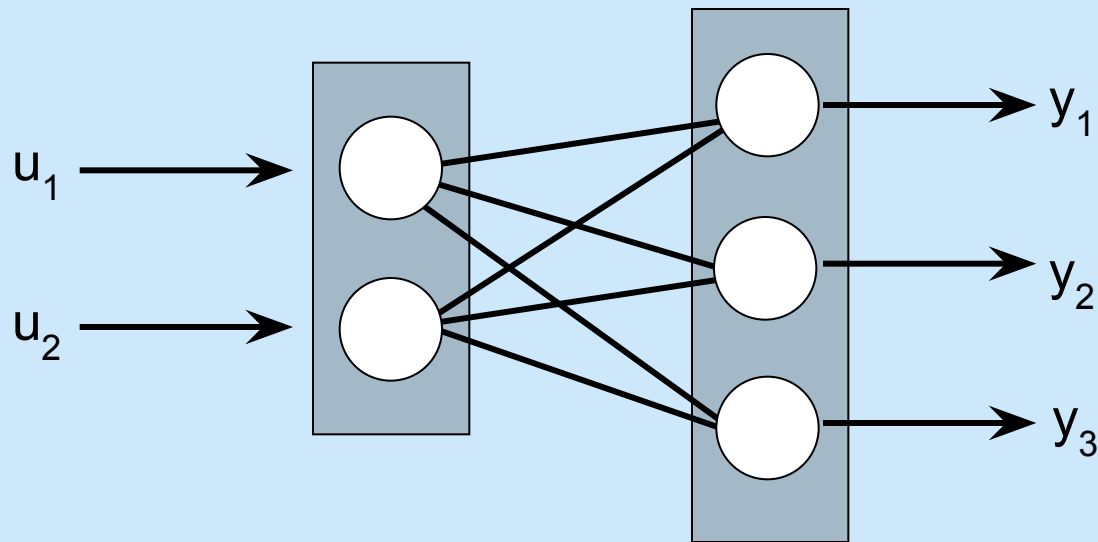
(2) promena težine veze proporcionalno grešci

$$w_{ji}(k+1) = w_{ji}(k) + \mu \varepsilon(k) u_{ji}(k)$$

(3) ukupna greška mreže za sve uzorke iz skupa za trening (kriterijum za zaustavljanje treninga – mreža je naučila kada je greška svedena na prihvatljivu meru)

$$E = \frac{1}{2n} \sum_{p=1}^n \varepsilon_p^2$$

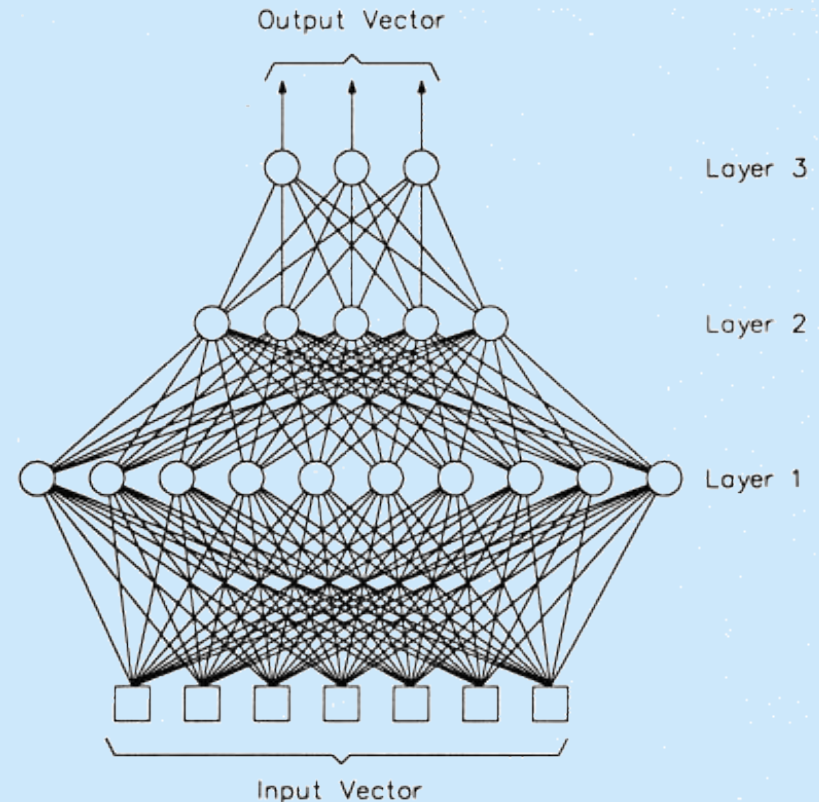
PERCEPTRON



- Step funkcija transfera -
- Perceptron learning - prvi algoritam za učenje nelinearnih sistema
- Samo za linearno separabilne probleme

VIŠESLOJNI PERCEPTRON

- Proširenje osnovnog perceptrona – ima jedan ili više *skrivenih* slojeva neurona između ulaznog i izlaznog
- Glatke/diferencijabilne funkcije transfera u neuronima (sigmoid, tanh, relu)
- Koristi Backpropagation algoritam za učenje koji se zasniva na LMS algoritmu, i minimizaciji funkcije greške.
- Mogu da rešavaju složene nelinearne probleme



BACKPROPAGATION ALGORITAM

- Iterativna minimizacija funkcije greške gradijentnom metodom
- Korišćenje diferencijabilnih funkcija transfera (sigmoid), i primena pravila za izvod složenih funkcija omogućavaju prenošenje greške sa izlaza unazad, sa sloja na sloj.

$$w(k+1) = w(k) + u * dE/dw$$

$$dE/dw = dE/dy * dy/ds * ds / dw$$

BACKPROPAGATION ALGORITAM

- Formula za izlazni sloj

$$w(k+1) = w(k) + \mu \varepsilon(k) u(k) f'(net(k))$$

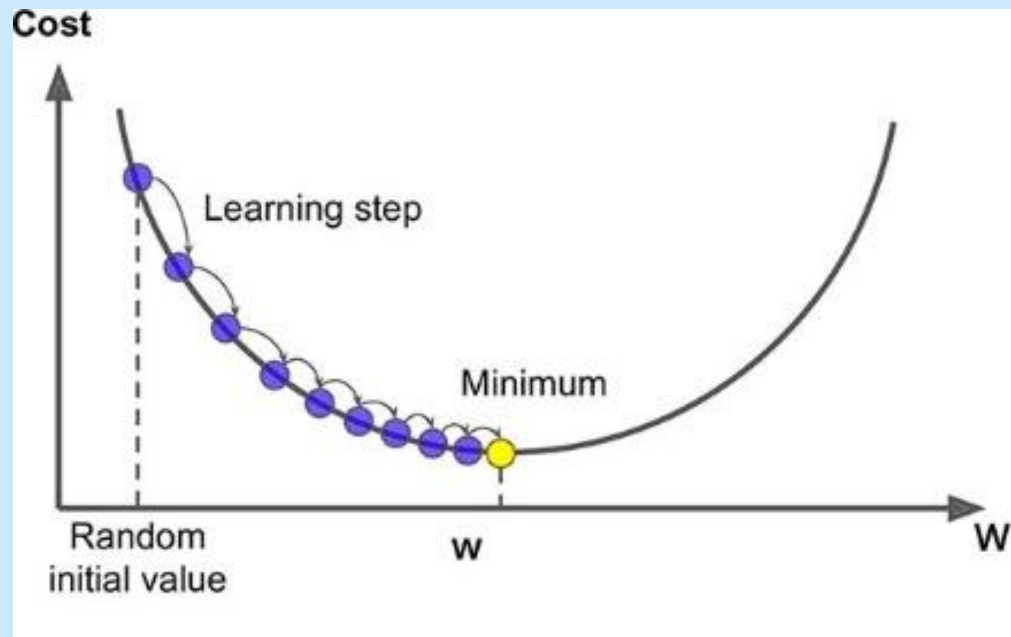
- Formula za podešavanje skrivenih slojeva

$$w_{ji}(k+1) = w_{ji}(k) + \mu f'(net_j(k)) \left(\sum_a \varepsilon_a(k) f'(net_a(k)) w_{aj}(k) \right) u_{ji}$$

OSNOVNI KONTROLNI PARAMETRI BACKPROPAGATION ALGORITMA

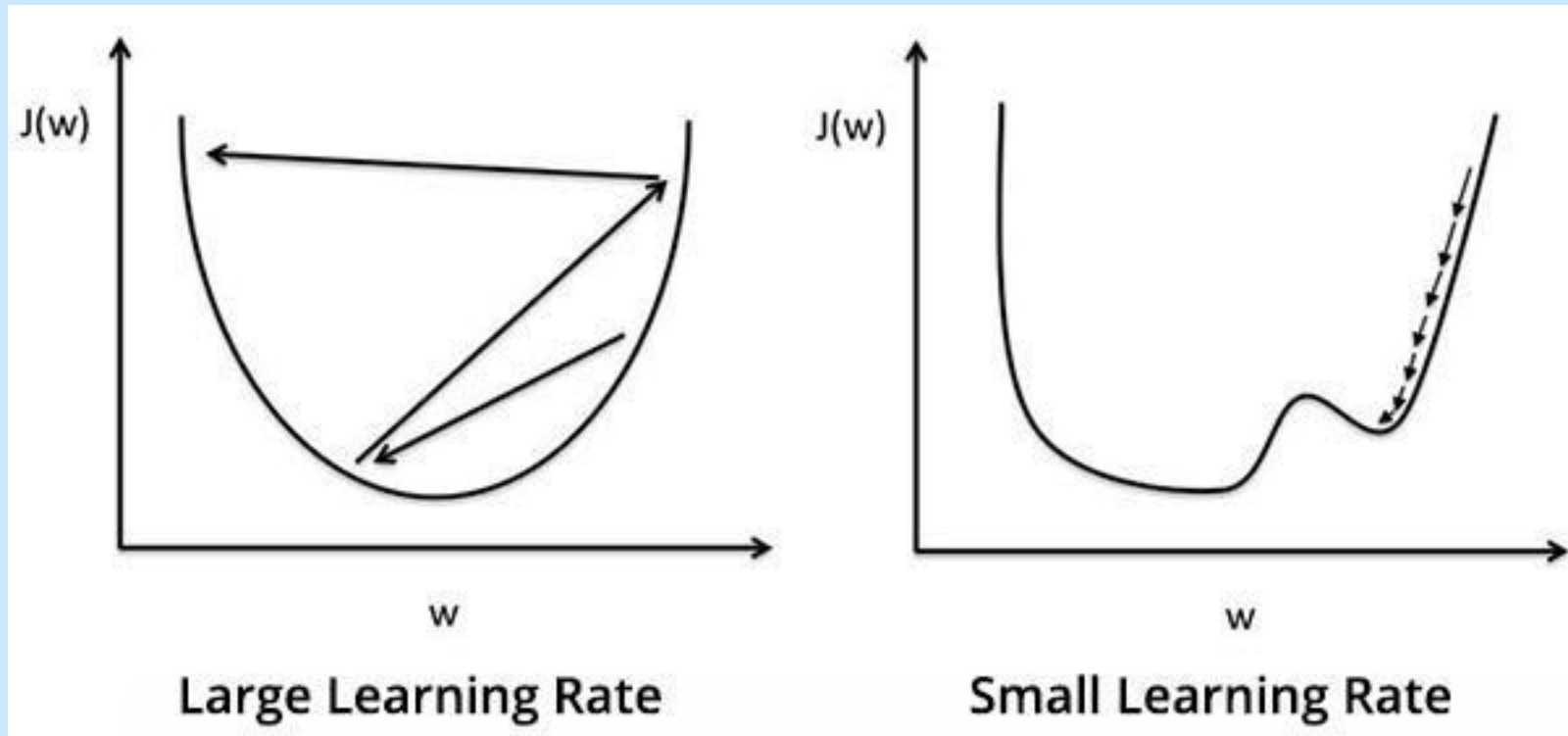
- **Max error** – trening se zaustavlja kada ukupna greška padne ispod ove vrednosti
- **Max iterations** – trening se zaustavlja posle zadatog broja iteracija
- **Learning rate** – određuje u kojoj meri greška utiče na promenu težine
- **Batch mode** – da li se promene težine vrše za svaki par (ulaz, izlaz) ili za skup parova

GRADIJENTNI SPUST



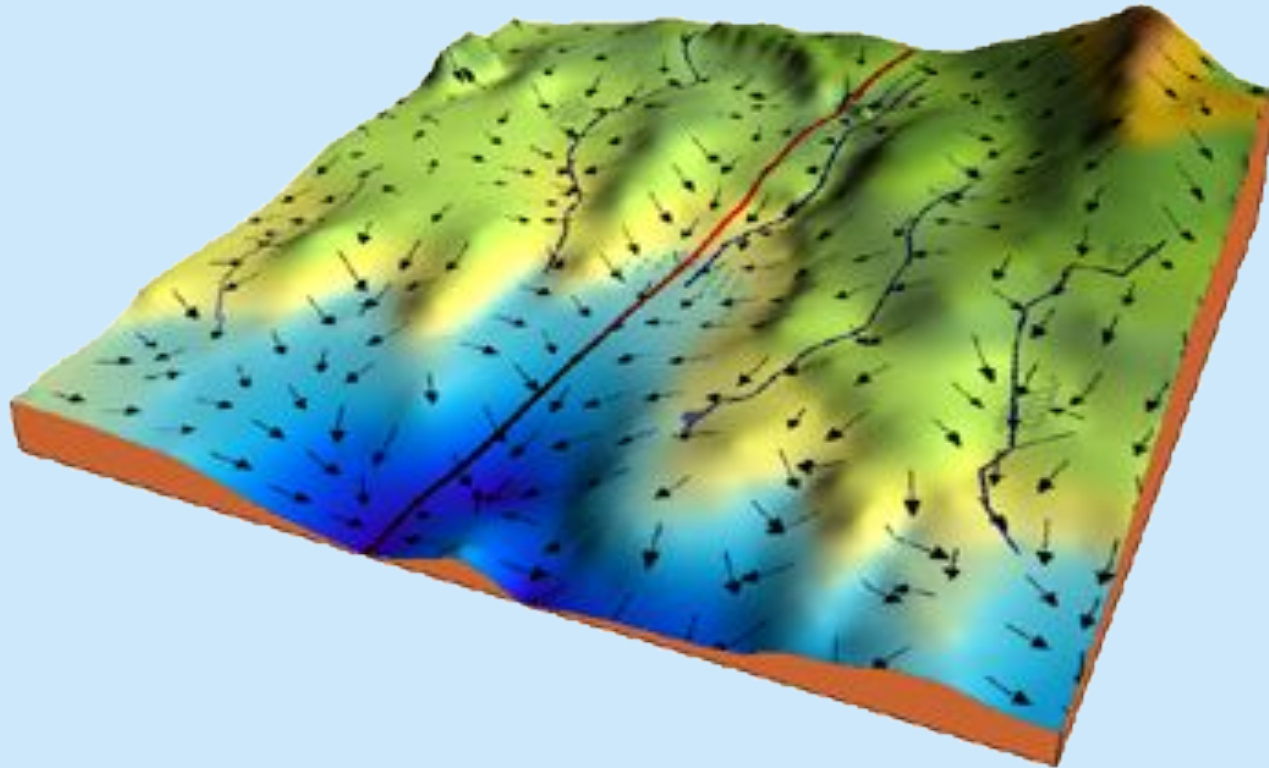
<https://saugatbhattarai.com.np/what-is-gradient-descent-in-machine-learning/>

PREMALI ILI PREVELIKI LEARNING RATE



<https://saugatbhattarai.com.np/what-is-gradient-descent-in-machine-learning/>

GRADIJENTNI SPUST U 2D SA NE-TRIVIJALNOM POVRŠINOM

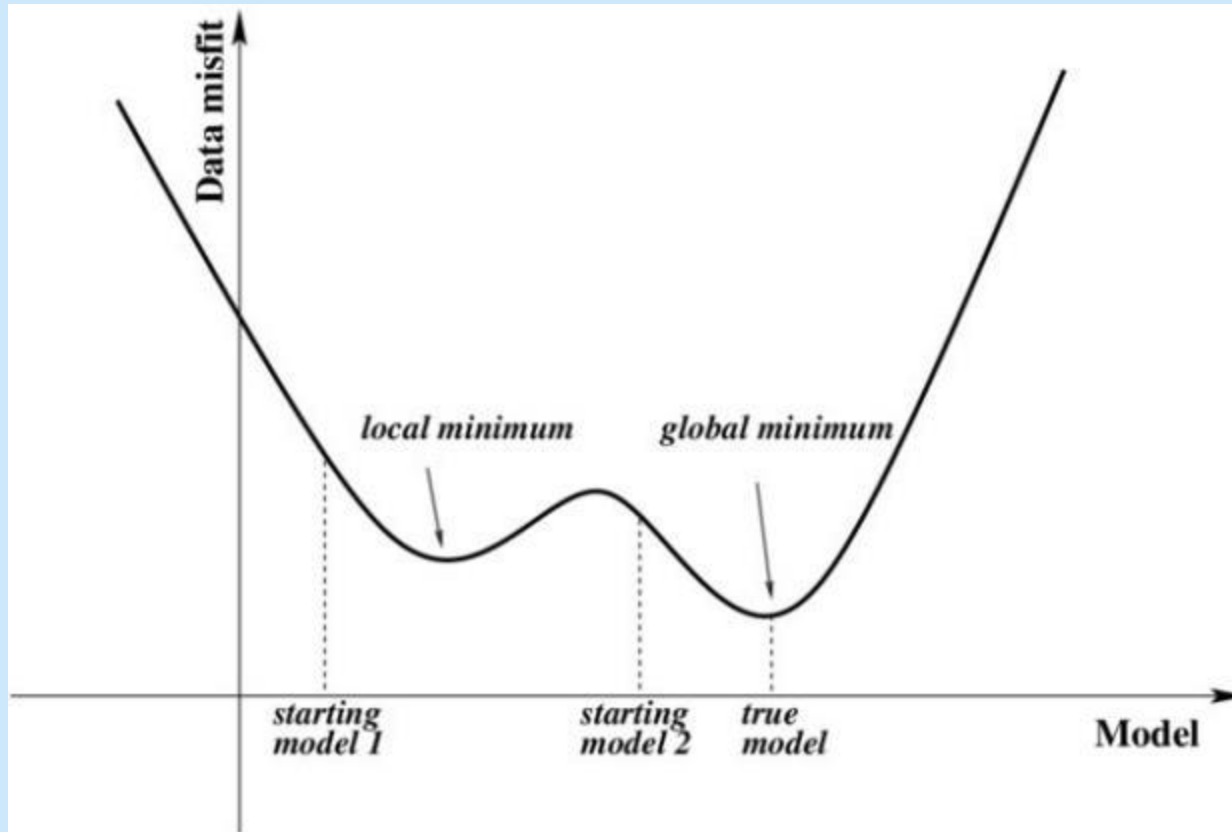


https://ml-cheatsheet.readthedocs.io/en/latest/gradient_descent.html

PROBLEMI BACKPROPAGATION ALGORITMA

- Problem lokalnog minimuma
- Problem nestajućih ili eksplozivnih gradijenta
- Problem mrtvih neurona
- Zaustavljanje algoritma:
 - spora konvergencija
 - rast greške
 - rano zaustavljanje - greška za određeni data set, izbegavanje overfittinga

PROBLEM LOKALNOG MINIMUMA



https://www.researchgate.net/publication/267820876_Full_waveform_inversion_with_image-guided_gradient/figures?lo=1

VARIJACIJE BACKPROPAGATION ALGORITMA

- Standardni backpropagation
- Momentum
- Quick propagation
- Resilient propagation
- Optimizacija zasnovana na gradijentim višeg reda (procena izvoda drugog reda i heseovematrice)
- I razne druge varijacije

PROCEDURA REŠAVANJE PROBLEMA POMOĆU NM

- Prikupljanje i priprema podataka (normalizacija, filtriranje, ...)
- Trening mreže
- Testiranje mreže
- Određivanje optimalnih parametara mreže i treninga eksperimentalnim putem (broj neurona, broj slojeva neurona, parametri algoritma za učenje, podaci za trening)
- Trening, test i validation set

PRIPREMA PODATAKA

- Filtriranje
- Normalizacija, standardizacija
- Redukcija dimenzionalnosti (PCA)
- Uspeh rešavanja u potpunosti zavisi od podataka koji se koriste za trening mreže

TRENING MREŽE

Određivanje optimalnih parametara mreže i algoritma za trening

Broj skrivenih slojeva i broj neurona u svakom sloju (više ne znači bolje, cilj je imati što manje)

Learning rate i momentum

Dinamičko podešavanje parametara

Validacija parametara (sa validation skupom)

Trening i test set

Pretreniravanje i generalizacija

KADA SE KORISTE NM

- Kada nema jasno definisanog matematičkog modela ili drugog rešenja
- Kada je potrebna otpornost na nepotpun ili pogrešan ulaz
- Kada je potrebna sposobnost učenja
- Visokodimenzionalnost
- Kada se sa NM postižu bolji rezultati nego sa alternativnim rešenjima (npr. odziv u realnom vremenu, tolerancija na greške)

VRSTE PROBLEMA ZA KOJE SE KORISTE VNM

- Klasifikacija
- Regresija/aproksimacija
- Klasterizacija
- Predviđanje vremenskih serija
- Optimizacija, kontrola, upravljanje
- Prepoznavanje (oblika, govora, vektora...)
- Obrada signala
- Modeliranje sistema

PROBLEMI U PRIMENI VNM

- Nedostatak semantike u strukturi
- Da li je neki problem uopšte rešiv sa NM?
- Problemi sa određivanjem arhitekture, parametara algoritma za učenje i treningom za određenu primenu
- Plastičnost / stabilnost

DODATNI LINKOVI I LITERATURA

- Sajt Neuroph projekta
 - <http://neuroph.sourceforge.net>
 - <http://github.com/neuroph>
- Sajt predmeta *Inteligentni sistemi*

<http://ai.fon.bg.ac.rs/osnovne/inteligentni-sistemi/neuronske-mreze/>

Knjige

- Neural Networks - A Systematic Introduction , besplatna online knjiga
- Neural Networks For Pattern Recognition, C. Bishop
- Fundamentals of Neural Networks, Lauren Fauset