Uvod u veštačku inteligenciju

Sadržaj

[Uvod 4](#_Toc138782537)

[Veštačka inteligencija 4](#_Toc138782538)

[Šta znači posedovati veštaču inteligenciju –četiri različita pristupa? 4](#_Toc138782539)

[Turingov test. 4](#_Toc138782540)

[Istorija razvoja veštačke inteligencije 5](#_Toc138782541)

[Agenti 5](#_Toc138782542)

[Šta su agenti? Sekvenca opažaja. Funkcija agenta. 5](#_Toc138782543)

[Mera performanse agenta. 5](#_Toc138782544)

[Okruženje zadatka. 5](#_Toc138782545)

[Osobine okruženja zadataka 6](#_Toc138782546)

[Vrste agenata 6](#_Toc138782547)

[Sistemi zasnovani na pravilima 7](#_Toc138782548)

[Šta je ES?Karakteristike ES. 7](#_Toc138782549)

[Predstavljanje znanja pravilima. Sistem produkcionih pravila. 7](#_Toc138782550)

[Struktura ES-a. Glavne komponente ES-a. 7](#_Toc138782551)

[Ciklus mehanizma za zaključivanje 8](#_Toc138782552)

[Promenljive u pravilima. Uslovni element NOT. 8](#_Toc138782553)

[Metode zaključivanja. 8](#_Toc138782554)

[Ulančavanje unazad. Formiranje AND-OR stabla. 8](#_Toc138782555)

[Prednosti ES. Klase ekspertnih sistema. Neke oblasti primene ES. 9](#_Toc138782556)

[Rešavanje problema pretraživanjem 10](#_Toc138782557)

[Neinformisane strategije pretraživanja 10](#_Toc138782558)

[Agent rešavanja problema 10](#_Toc138782559)

[Formalni opis problema 10](#_Toc138782560)

[Prostor stanja problema 10](#_Toc138782561)

[Pretraživanje u širinu 11](#_Toc138782562)

[Pretraživanje sa uniformnim troškom 11](#_Toc138782563)

[Pretraživanje u dubinu 11](#_Toc138782564)

[Iterativno pretraživanje u dubinu 12](#_Toc138782565)

[Dvosmerno pretraživanje 12](#_Toc138782566)

[Informisane strategije pretraživanja 12](#_Toc138782567)

[Informisanost.Heuristika. 12](#_Toc138782568)

[Pretraživanje „prvo najbolji" 13](#_Toc138782569)

[Algoritam A\* 13](#_Toc138782570)

[Uslovi optimalnosti za heuristike 13](#_Toc138782571)

[Pretraživanje sa suprotstavljanjem 13](#_Toc138782572)

[Igra kao problem pretraživanja. 13](#_Toc138782573)

[Minimax algoritam. 14](#_Toc138782574)

[Alfa-beta odsecanje. 14](#_Toc138782575)

[Element slučajnosti u igrama. 15](#_Toc138782576)

[Mašinsko učenje 15](#_Toc138782577)

[Šta je mašinsko učenje? 15](#_Toc138782578)

[Vrste mašinskog učenja. 15](#_Toc138782579)

[Nadgledano mašinsko učenje 16](#_Toc138782580)

[Skup za obučavanje,algoritam za obučavanje, obučeni model. 16](#_Toc138782581)

[Koraci nadgledanog mašinskog učenja. 16](#_Toc138782582)

[Overfitting i underfitting. 17](#_Toc138782583)

[Bias-variance tradeoff. 17](#_Toc138782584)

[Trening, validacioni i testni skup podataka. 18](#_Toc138782585)

[K-unakrsna validacija. 18](#_Toc138782586)

[Ostavi jednog van validacija 19](#_Toc138782587)

[Matrica konfuzije i metrike koje iz nje proizilaze. 20](#_Toc138782588)

[ROC i AUC. 22](#_Toc138782589)

[Obučavanje stabala odlučivanja. 22](#_Toc138782590)

[Entropija.Informaciono poboljšanje. 23](#_Toc138782591)

[Perceptron. Algoritam obučavanja perceptrona. 24](#_Toc138782592)

[Aktivacija neurona. Aktivaciona funkcija. 25](#_Toc138782593)

[Mreže sa propagacijom unapred. 26](#_Toc138782594)

[Način rada Backpropagation algoritma. 27](#_Toc138782595)

[kNN algoritam. 30](#_Toc138782596)

[Linearna regresija 31](#_Toc138782597)

# Uvod

## Veštačka inteligencija

### Šta znači posedovati veštaču inteligenciju –četiri različita pristupa?

Artificial intelligence – razvoj kompjutera sposobnih da obavljaju zadatke koji zahtevaju ljudsku inteligenciju.

Veštačka inteligencija (Artificial intelligence -AI) omogućava mašinama da:

* Uče iz iskustva,
* Prilagođavaju se novim ulazima,
* Obavljaju zadatke koji su inače namenjeni ljudima.

Posedovati veštačku inteligenciju znači:

* Razmišljati kao ljudsko biće
* Razmišljati racionalno ( Reprezentovanje činjenica o svetu korišćenjem logike, korišćenje logičkog zaključivanja)
* Postupati kao ljudsko biće
* Postupati racionalno ( Racionalni agent je onaj agent koji deluje tako da ostvaruje najbolji ishod ili kada postoji neizvesnost, najbolji očekivani ishod. )

### Turingov test.

Da bi mogao da prođe test računar treba da poseduje sledeće sposobnosti:

* predstavljanje znanja (knowledge representation),
* zaključivanje(reasoning),
* obradu prirodnog jezika (natural language processing),
* mašinsko učenje (machine learning)

Potpuni Turingov test zahteva i:

* računarsko viđenje (computer vision)
* robotiku(robotics)

Tjuringov test se sprovodi na sledeći način: čovek ispitivač komunicira sa čovekom i mašinom. Nakon određenog vremena, ako čovek ispitivač ne može sa sigurnošću da odredi koji sagovornik je čovek, a koji mašina, onda je mašina prošla test.

### Istorija razvoja veštačke inteligencije

Početak ( Model veštačkog neurona ). Rađanje AI ( Dartmouth College ). AI Winter. AI postaje industrija ( Komercijalni uspeh ekspertskih sistema ). Deep Blue ( pobeđuje Kasparova u šahu ). IBM Watson ( Računar pobedio u kvizu ). Deep NN.

## Agenti

### Šta su agenti? Sekvenca opažaja. Funkcija agenta.

Pod agentom možemo podrazumevati bilo šta što svoju okolinu spoznaje pomoću senzora, i na nju deluje preko aktuatora (Čovek, robot, softver). Sve što agent u toku vremena(do nekog trenutka) opazi predstavlja njegovu istoriju opažanja(sekvenca opažaja-percept sequence). Agentov izbor akcije u bilo kom datom trenutku može zavisiti od čitave opažajne sekvence uočene do tog trenutka, ali ne i od nečega što on nije opazio. Funkcija agenta je apstraktan matematički opis. Ponašanje agenta je opisano funkcijom agenta koja preslikava bilo koju opažajnu sekvencu u delovanje–akciju. Funkcija agenta se za veštačkog agenta realizuje programom agenta. Program agenta je konkretna realizacija, koja se izvršava u nekom fizičkom sistemu.

### Mera performanse agenta.

Mera performanse vrednuje ponašanje agenta u okruženju. Merom performanse se ocenjuje niz stanja kroz koja prolazi okolina agenta pod njegovim dejstvom. Ako je sekvenca stanja okoline poželjna, onda je agent dobro postupao.

### Okruženje zadatka.

Okruženja zadataka su problemi za koje agenti predstavljaju rešenja. Mera performanse agenta, okruženje, senzori i aktuatori sačinjavaju okruženje zadatka. **PEAS** (**P**erformance, **E**nvironment, **A**ctuators, **S**ensors)

### Osobine okruženja zadataka

* Opservabilnost – potpuna/delimična - Senzori detektuju / ne detektuju sve aspekte od važnosti za izbor akcije, pri čemu važnost zavisi od mere performansi.
* Agenti – jednoagentno/višeagentno ( kompetitivno ili kooperativno )
* Determinističko/ stohastičko - Sledeće stanje okruženja je / nije potpuno određeno trenutnim stanjem i akcijom koju vrši agent
* Epizodno / sekvencijalno - Sledeća epizoda ne zavisi od akcija koje su preduzete u prethodnim epizodama
* Statično / dinamično - Da li se okruženje menja dok agent odlučuje o svojoj sledećoj akciji?
* Diskretno / kontinualno - Stanja okruženja, način tretiranja vremena

### Vrste agenata

* Jednostavni refleksni agent – Jednostavno pravilo tipa condition-action omogućava da agent ostvari neophodnu vezu opažaja i akcije
* Refleksni agent zasnovan na modelu - Agent prati kretanje onih delova sveta koje sada ne može videti i na taj način rešava problem delimične opservabilnosti.
* Agent zasnovan na cilju - On beleži stanje sveta kao i skup ciljeva koje pokušava da ostvari, i bira delovanje koje će ga (na kraju) dovesti do ostvarenja njegovih ciljeva
* Agenti koji uče - Učenje dozvoljava agentu da funkcioniše u okruženjima koja su za njega u početku bila nepoznata i postane kompetentniji nego što mu je njegovo početno znanje dozvoljavalo.

# **Sistemi zasnovani na pravilima**

### Šta je ES?Karakteristike ES.

Ekspertni system je kompjuterski program koji simulira process ljudskog rezonovanja i primenom ekspertskog znanja rešava probleme.

Ekspert – osoba koja poseduje znanje ili veštine koje su većini ljudi nepoznate ili nedostupne.

Karakteristike ES:

* Simulira ljudsko razmišljanje o problem
* Rezonuje na osnovu reprezentacije ljudskog znanja
* Probleme rešava korišćenjem heuristika ili aproksimacija, koje za razliku od algoritamskih metoda ne garantuju uspeh
* Obično je ograničen na neki specifičan domen

### Predstavljanje znanja pravilima. Sistem produkcionih pravila.

Pravilo: IF “svetlo\_je\_crveno” THEN “stop”

Sistem produkcionih pravila:

* Sistem radi sa skupom pravila I skupom tvrdnji.
* Tvrdnja je izjava (iskaz) da je nešto tačno.
* Pravila traže određene uzorke (patterns) u tvrdnjama-antecedenti pravila
* I obično stvaraju nove tvrdnje - konsekvente.
* Pravila mogu i da brišu postojeće tvrdnje.

### Struktura ES-a. Glavne komponente ES-a.

Korisnik snabdeva ES tvrdnjama, a za uzvrat dobija ekspertski savet. Interno, ES se sastoji od dve glavne komponente: **baze znanja** i **mehanizma zaključivanja** (inference engine). U bazi je znanje na osnovu koga mehanizam zaključuje. Ovi zaključci su odgovor ES-a na upit korisnika. ES rezonuje na osnovu znanja iz određenog domena –on nije general problem solver.

Glavne komponente ES:

* user interface – interakcija sa korisnicima, razvoj i održavanje baze znanja
* sistem za prikupljanje znanja - omogućava korisniku da automatski unosi znanje u system
* baza znanja - sadrži znanje kodirano pravilima
* radna memorija – sadrži trenutno aktuelne tvrdnje
* mehanizam zaključivanja - vrši zaključivanje izvršavajući pravilo sa najvišim prioritetom u agenda ( lista pravila zadovoljenih tvrdnjama koje su u radnoj memoriji )
* system za objašnjavanje - objašnjava korisniku način rezonovanja ESa

### Ciklus mehanizma za zaključivanje

1. inference engine pronalazi pravila čiji su antecedenti ( leve strane ) zadovoljeni
2. pravila koja su zadovoljena smeštaju se u agendu i nazivaju se aktivirana pravila
3. Razrešavanje konflikta ( bira pravilo iz agende sa najvišim prioritetom )
4. Izvršenje pravila (firing) - sprovodi akcije određene konsekventom ( desnom stranom ) odabranog pravila
5. Update-uje agendu pravila - pravila čiji su antecedent zadovoljeni dodaje u agendu, iz agende uklanja pravila koja nisu zadovoljena

Ciklus se završava kada u agendi nema više pravila ili kada se naiđe na eksplicitnu komandu za zaustavljanje programa

### Promenljive u pravilima. Uslovni element NOT.

Pravila mogu sadržati promenljive. Promenljiva dobija vrednost kroz proces uparivanja tvrdnji sa uslovima pravila. Promenljive omogućavaju uparivanje uslova pravila sa više tvrdnji. Konsekvent može sadržati promenljive koje su se kroz uslov pravila vezale za konkretnu vrednost. Ako postoji NOT u antecendentu, mehanizam za zaključivanje će pretražiti tvrdnje da se „uveri“da nema stavki koje odgovaraju uslovu pravila. Nove promenljive se ne mogu prvi put u pravilu uvoditi pod dejstvom NOT klauzule–mehanizam za zaključivanje neće znati šta da radi sa njima.

### Metode zaključivanja.

* Ulančavanje unapred(Forward chaining) - Zaključivanje od tvrdnji ka zaključcima koji iz njih slede. Tokom procesa ulančavanja unapred nove tvrdnje mogu biti dodate u listu.
* Ulančavanje unazad (Backward chaining) - Zaključivanje od hipoteza (potencijalnih zaključaka) ka tvrdnjama koje podržavaju hipoteze.

### Ulančavanje unazad. Formiranje AND-OR stabla.

1. Kada pokušava da dokaže hipotezu, proces ulančavanja unazad prvo pokušava da nađe odgovarajuću tvrdnju (matching assertion) u listi.
2. Ako odgovarajuća tvrdnja nije pronađena, traži se pravilo koje ima zaključak koji se poklapa sa hipotezom.
3. Uslovi nađenog pravila se nadalje posmatraju kao hipoteze i pokušava se ulančavanje unazad od njih, ka tvrdnjama u listi, kroz ponavljanje koraka 1, 2 i 3.

Tokom procesa ulančavanja unazad nema dodavanja novih tvrdnji u listu. Ako postoji konflikt, uvek se bira pravilo koje je prvo na redu. Ako postoji uparivanje uslova (niza uslova) jednog pravila sa više različitih tvrdnji (grupa tvrdnji) u listi onda se uzima tvrdnja koja je prva po redu.

Formiranje AND-OR stable:

1. U koren stabla stavljamo hipotezu koju želimo da dokažemo.
2. Ako je hipoteza među tvrdnjama, onda smo je dokazali.
3. Ako nije, tražimo pravilo čiji se konsekvent poklapa sa našom hipotezom i stablo razvijamo, tako što dodajemo nove čvorove koji odgovaraju uslovima pravila.

### Prednosti ES. Klase ekspertnih sistema. Neke oblasti primene ES.

Prednosti ekspertnih sistema (ES):

* Sposobnost donošenja odluka na osnovu kompleksnih i velikih količina podataka
* Povećana preciznost i efikasnost u odnosu na čoveka
* Konsistentnost u donošenju odluka
* Mogućnost rada u stanju bez sna i ubrzavanje procesa donošenja odluka.

Klase ekspertnih sistema:

* ES zasnovani na pravilima
* ES zasnovani na znanju
* ES zasnovani na učenju
* ES zasnovani na sistemima za upravljanje bazama znanja.

Neke oblasti primene ekspertnih sistema:

* Medicina
* Bankarstvo i finansije
* Proizvodnja
* Tehnologija informacija
* Vojska i bezbednost
* Ekologija i životna sredina.

# Rešavanje problema pretraživanjem

## Neinformisane strategije pretraživanja

### Agent rešavanja problema

Ono što je karakteristično za njih je da stanja sveta smatraju celinama. Dok rešavaju problem, oni prelaze iz jednog stanja sveta (okruženja u kome se nalaze) u drugo. Ne zanima ih unutrašnja struktura sveta, ona nije vidljiva za algoritme koje ovi agenti koriste prilikom rešavanja problema.

### Formalni opis problema

* Početno stanje od koga agent počinje.
* Opis mogućih akcija koje su na raspolaganju agentu
* Model prelaza(transition model) –šta se dešava kada primenimo neku akciju.
* Testiranje cilja – da li je dato stanje ciljno stanje.
* Funkcija troška putanje koja svakoj putanji dodeljuje numerički iznos troška.

### Prostor stanja problema

Skup svih stanja dostupnih iz početnog stanja bilo kojom sekvencom akcija. Prostor stanja formira usmerenu mrežu ili graf u kojem čvorovi predstavljaju stanja, a veze između čvorova su akcije.

Merenje performansi algoritma:

* Potpunost – da li sigurno pronalazi rešenje?
* Optimalnost – da li pronalazi optimalno rešenje ( ono sa najnižim troškom putanje )?
* Vremenska složenost – koliko vremena treba da se pronađe rešenje?
* Prostorna složenost – koliko memorije je potrebno da bi se seizvršilo pretraživanje?

P – klasa problema koji se mogu rešiti u polinomnom vremenu

NP – klasa problema koji se mogu rešiti nedeterministički u polinomnom vremenu.

𝑏 – factor grananja maksimalan broj sledbenika nekog čvora

𝑑 – dubina najplićeg ciljnog čvora ( broj koraka duž putanje od korena do najplićeg ciljnog čvora)

– broj čvorova na dubini d

𝑚 – maksimalna dužina putanje u prostoru stanja (moguće ∞)

### Pretraživanje u širinu

Nakon razvića korenskog čvora, razvijaju se svi njegovi sledbenici, zatim njihovi sledbenici, itd. Čvorovi na dubini d proširuju se tek nakon što se prošire svi čvorovi na nivou d−1, tj. pretražujemo nivo po nivo. Rub se predstavlja FIFO redom čekanja. Testiranje cilja se primenjuje na čvor kada se on generiše, umesto na čvor koji je izabran za razvijanje.

Osobine:

* Potpunost – ako je najplići ciljni čvor na konačnoj dubini 𝑑 pretraživanje u širinu će ga svakako naći.
* Optimalnost - u svakom koraku proširuje se najplići čvor, pa je strategija optimalna u slučaju kada je: cena prelaza konstantna ili neopadajuća funkcija dubine čvora
* Vremenska složenost: O()
* Prostorna složenost: O()

### Pretraživanje sa uniformnim troškom

Umesto da se razvija najplići čvor, razvija se čvor n sa najnižim troškom putanje g(n). Memorisanje ruba u obliku reda čekanja sa prioritetom, uređenog po g. Testiranje cilja se primenjuje na čvor tek kada je on izabran za razvijanje, umesto kada je prvi put generisan.

Osobine:

* Potpunost – da
* Optimalnost – da
* Vremenska složenost - Ako svaka akcija košta najmanje ε, onda smo na svakom koraku bar za ε bliži cilju. Ako je C\* trošak optimalnog rešenja,onda treba da napravimo koraka. Vremenska složenost - 𝑂().
* Prostorna složenost - 𝑂()

### Pretraživanje u dubinu

Pretraživanje u dubinu uvek prvo proširuje najdublji čvor u trenutnom rubu stabla pretraživanja. Pretraživanje odmah prelazi na najdublji nivo stabla, tamo gde čvorovi nemaju sledbenike. Pošto ti čvorovi budu razvijeni, oni se isključuju iz ruba, pa se pretraživanje vraća na sledeći najdublji rubni čvor.

Osobine:

* Potpunost - ne, jer se može zaglaviti u beskonačnoj petlji.
* Optimalnost - ne, jer ne pretražuje nivo po nivo, tako da može otići u veliku dubinu po jednom podstablu, čak i kada je ciljni čvor na dubini 1 u drugom podstablu.
* Vremenska složenost je određena veličinom prostora pretrage i iznosi 𝑂().
* Prostorna složenost: O(b\*m)

### Iterativno pretraživanje u dubinu

Iterativnim produbljivanjem kombinuju se prednosti pretraživanja u dubinu I pretraživanja u širinu. Ima male memorijske zahteve.

Osobine:

* Potpunost - kada je faktor grananja konačan.
* Optimalnost - kada su sve cene prelaska jednake ili neopadajuća f-ja dubine.
* Vremenska složenost - O()
* Prostorna složenost - O(b\*d)

### Dvosmerno pretraživanje

Vrše se dva istovremena pretraživanja(jedno od starta do cilja, a drugo od cilja do starta).

je mnogo manje od . Privlačno zbog male vremenske složenosti. Međutim, nije lako pretraživati unazad. Potrebne su metode da se izračuna prethodnik stanja x. Rešivi problemi: Slagalica 8, Put kroz Rumuniju. Nerešivi problem: 8 kraljica ( Nema kraljice koja napada drugu kraljicu).

Informisane strategije pretraživanja

### Informisanost.Heuristika.

Efikasnost pretraživanja se može značajno povećati ako imamo način da uredimo moguće poteze, tako da oni potezi koji najviše obećavaju budu primenjeni prvi. U većini situacija možemo na neki način izmeriti koliko ćemo brzo doći do cilja ako odaberemo određenu akciju za prelazak u naredno stanje. Metode koje koriste takve mere nazivaju se informisane ili heurističke metode. Čvor za proširivanje se bira na osnovu funkcije vrednovanja, 𝑓(𝑛). Izbor funkcije vrednovanja definiše strategiju informisanog pretraživanja. Heuristika je iskustveno pravilo o prirodi problema I osobinama cilja, čija je svrha brže usmeravanje pretrage ka cilju.

Pretraživanje „prvo najbolji"

Razumna strategija pretraživanja je da uvek pokušavamo da se krećemo na takav način da minimiziramo približnu udaljenost do cilja. „prvo najbolji“ bira onaj čvor koji se čini najbliži cilju, ne uzimajući u obzir ukupnu cenu puta. Čvorovi se vrednuju samo pomoću heurističke funkcije(𝑓(𝑛)=ℎ(𝑛)).

### Algoritam A\*

Čvorovi se procenjuju pomoću:

* 𝑔(𝑛)−stvarna cena puta od početnog čvora do čvora 𝑛
* ℎ(𝑛)−procenjena cena putanjaod čvora 𝑛 do cilja

𝑓(𝑛)=𝑔(𝑛)+ℎ(𝑛).

Pod uslovom da heuristička funkcija ℎ(𝑛) zadovoljava izvesne uslove, A\* pretraživanje je potpuno i optimalno. Algoritam je identičan sa algoritmom pretraživanje sa uniformnim troškom izuzev što A\* korsti 𝑔(𝑛)+ℎ(𝑛) umesto 𝑔(𝑛). Verzija pretraživanja po stablu algoritma A\* je optimalna ako je h(n) prihvatljiva dok za graf važi da mora da bude konzistenta.

### Uslovi optimalnosti za heuristike

* Prihvatljivost – nikada ne precenjuje trošak stizanja do cilja. Neka je 𝑔(𝑛) stvarni trošak stizanja do 𝑛, ℎ(𝑛) je procenjena najniža cena stizanja od čvora 𝑛 do cilja i ona je uvek manja ili jednaka od prave cene (𝑛). Tada je . Prihvatljive heuristike su po prirodi optimistične jer one smatraju da je trošak rešavanja problema manji nego što je on to stvarno.
* Konzistentnost – za svaki čvor n, h(n) c(n,n’) + h(n’), gde je h heuristika a c funkcija troška prelaska između dva čvora. Svaka konzistentna heuristika je i prihvatljiva.

Pretraživanje sa suprotstavljanjem

### Igra kao problem pretraživanja.

Nepredvidivost uticaja drugih agenata. Kompetitivno okruženje u kome agenti imaju suprotstavljene ciljeve -pretraživanje sa suprotstavljanjem (igra).

* S0-početno stanje
* Igrač(s)–igrač na potezu u stanju s
* Akcije(s)–skup dozvoljenih poteza u datom stanju
* Rezultat(s, a) –model prelaza (rezultat poteza)
* Test-završetka(s)–tačan ukoliko je igra završena
* Korisnost(s, p) –krajnja numerička vrednost (dobitak) igre u stanju sza igrača p.

### Minimax algoritam.

Vrednosti igre je u čvorovima listovima iz pozicije igrača koji bira potez u korenom čvoru. Taj igrač ima za cilj da ostvari što veću vrednost igre –MAX igrač. Njegov protivnik želi suprotno –MIN igrač. Agent gleda unapred I evaluira poteze, kako bi izabrao onaj potez koji je najbolji za njega. Jedan igrač bira poteze da maksimizuje svoj dobitak - MAX. Drugi igrač bira poteze da minimizuje svoj gubitak - MIN.

Minimax algoritam:

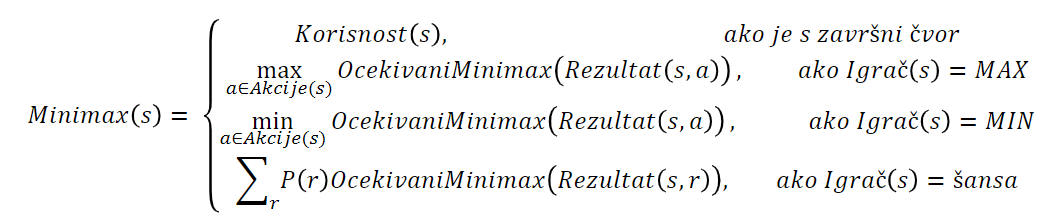
* Ako je dostignuta granica pretrage, vrati vrednost trenutne pozicije u odnosu na odgovarajućeg igrača.
* U suprotnom, ako je na potezu MIN izvrši Minimax algoritam nad svom decom trenutnog čvora i vrati MINIMUM svih rezultata.
* U suprotnom, ako je na potezu MAX izvrši Minimax algoritam nad svom decom trenutnog čvora i vrati MAXIMUM svih rezultata.

Složenost ista kao kod pretrage u dubinu.

### Alfa-beta odsecanje.

Smanjuje broj stanja koja razmatramo. Alfa-beta odsecanje vraća isti redosled poteza kao i Minmax, ali odseca one grane stabla koje nikako ne bi uticale na konačnu odluku. Pratiti vrednost najboljeg poteza tokom minimax pretrage iz perspektive MAX i MIN igrača. Za MAX igrača, ta vrednost je 𝜶(cilj: da bude što VEĆA) – ovo je garantovana najbolja vrednost koju MAX može da dobije(garantovana donja granica za maksimalnu vrednost). Za MIN igrača, ta vrednost je 𝜷(cilj: da bude što MANJA) –ovo je garantovana najbolja vrednost koju MIN može da dobije, tj. izgubi(garantovana gornja granica za minimalnu vrednost). Na početku je 𝛼=−∞;𝛽=+∞; tako da je početni opseg 𝛼,𝛽=[−∞,+∞]. Tokom pretrage stabla igre, ovaj opseg treba da se sužava sve dok 𝛼 ne postane jednako 𝛽. 𝛼 **NIKADA** ne sme postati veće od 𝛽! MAX računa 𝜶; MIN računa 𝜷; odsecanje se vrši svaki put kada 𝛼 premaši 𝛽, ili kada 𝛽 postane manje ili jednako 𝛼. Odsecanje: svaki put kada je 𝛽≤𝛼.

### Element slučajnosti u igrama.



P(r) - verovatnoća

# Mašinsko učenje

### Šta je mašinsko učenje?

Mašinsko učenje je proučavanje računarskih algoritama koji omogućavaju računarskim programima da se automatski poboljšavaju kroz iskustvo.

Klasifikacija - izlaz je jedna vrednost iz konačnog skupa vrednosti

Regresija -izlaz je kontinualna vrednost

### Vrste mašinskog učenja.

* Nadgledano učenje (supervised learning) – Primeri za učenje dati su u obliku parova vrednosti ulaza i izlaza. Uči se funkcija koja preslikava ulaze u izlaze.
* Nenadgledano učenje (unsupervised learning) - Donosi zaključke I pronalazi obrasce iz ulaznih podataka koji nisu označeni. Zadaci:
  + Grupisanje (klasterizacija (Clustering)) – Otkrivanje potencijalno korisnih grupa ulaznih primera koji su po nečemu slični (segmentacija klijenata). Uobičajene tehnike klasterizacije su k-means, hierarchical, meanshiftclustering, i density-based clustering.
  + Prepoznavanje anomalija u podacima (Anomaly detection) Identifikacija retkih pojava(događaja) koje izazivaju sumnje značajnim razlikovanjem od većine podataka.
  + Smanjenje dimenzija (Dimensionality reduction) – Proces redukovanja broja atributa i izdvajanje onih koji imaju najveći značaj za problem koji se rešava. Uobičajena tehnika je PCA (Principal ComponentAnalysis)
* Učenje sa pojačavanjem (reinforcement learning) - Agent uči iz niza pojačavanja –nagrada ili kazni.

## Nadgledano mašinsko učenje

### Skup za obučavanje,algoritam za obučavanje, obučeni model.

Algoritam za obučavanje je postupak koji se izvodi nad podacima da bi se stvorio „model mašinskog učenja“. Model mašinskog učenja je izlaz iz algoritma mašinskog učenja pokrenutog nad podacima. Predstavlja pravila, brojeve i sve druge algoritamski-specifične strukture podataka koje su potrebne za predikciju. Skup za obuku se deli na trening skup i testni skup. Trening skup služi kao ulaz algoritmu za obučavanje kako bi kreirao model. Testni skup služi za ocenjivanje kvaliteta modela.

### Koraci nadgledanog mašinskog učenja.

1. Skup podataka: Pripremite skup podataka koji će se koristiti za obuku modela. Skup podataka treba da sadrži parove ulaznih karakteristika i odgovarajućih ciljnih vrednosti. Podaci se obično dele na dve podskupine: skup za obuku (za učenje modela) i skup za testiranje (za evaluaciju modela).
2. Odabir algoritma: Izaberite odgovarajući algoritam nadziranog učenja za vaš problem. Algoritmi se biraju na osnovu vrste problema (klasifikacija, regresija, segmentacija itd.) i karakteristika podataka.
3. Izbor funkcija i pretprocesiranje podataka: Identifikujte relevantne ulazne karakteristike (funkcije) koje će se koristiti za obuku modela. Ovo može uključivati normalizaciju podataka, skaliranje, enkodiranje kategoričkih varijabli i slične tehnike.
4. Podela podataka: Podelite skup podataka na skup za obuku i skup za validaciju/testiranje. Skup za obuku će se koristiti za obuku modela, dok će se skup za validaciju/testiranje koristiti za evaluaciju performansi modela.
5. Obuka modela: Koristite algoritam nadziranog učenja i skup podataka za obuku modela. Model će prilagođavati parametre na osnovu podataka kako bi naučio odnos između ulaznih karakteristika i ciljnih vrednosti.
6. Validacija modela: Koristite skup za validaciju kako biste procenili performanse obučenog modela. Ovo može uključivati izračunavanje metrika kao što su tačnost, preciznost, odziv, F1-mera, srednja kvadratna greška i druge relevantne metrike.
7. Optimizacija modela: Ukoliko model ne zadovoljava zadovoljavajuće performanse, možete eksperimentisati sa različitim hiperparametrima algoritma, kao što su stopa učenja, broj epoha, dubina mreže i druge konfiguracije, kako biste poboljšali performanse modela.
8. Testiranje modela: Kada ste zadovoljni performansama modela, testirajte ga na nezavisnom skupu podataka (skup za testiranje). Ovo će vam pružiti objektivan uvid u performanse modela na novim, neviđenim podacima.
9. Poboljšavanje modela: Ako performanse modela nisu zadovoljavajuće, možete se vratiti na prethodne korake, prilagoditi parametre, dodati nove karakteristike ili promeniti algoritam kako biste poboljšali performanse modela.
10. Evaluacija i implementacija: Kada ste zadovoljni performansama modela, možete ga implementirati u stvarnom okruženju i koristiti ga za predviđanja ili druge zadatke za koje je dizajniran.

### Overfitting i underfitting.

Trening skup služi za treniranje modela. Validacioni skup služi za odabir konkretnog modela sa određenim parametrima i govori o sposobnosti našeg modela da vrši generalizaciju..Testni skup se koristi za procenu performansi potpuno formiranog modela.

Preučavanje (overfitting) – greška validacije je velika a greška nad trening skupom je mala

Nedovoljna obučenost (underfitting) – greška validacije je velika i greška nad trening skupom je velika

Good fit – Greška validacije je mala, ali je malo veća od trening greške

### Bias-variance tradeoff.

Bias se odnosi na grešku modela koja nastaje kao posledica uprošćavanja pretpostavki ili pogrešnih pretpostavki koje model pravi u odnosu na podatke za učenje kako bi što lakše naučio ciljnu funkciju.

Varijansa oslikava preprilagođenost modela trening skupu.Meri odstupanje predviđanja našeg modela od očekivanog predviđanja.

A picture containing text, screenshot, line, diagram

Description automatically generated

### Trening, validacioni i testni skup podataka.

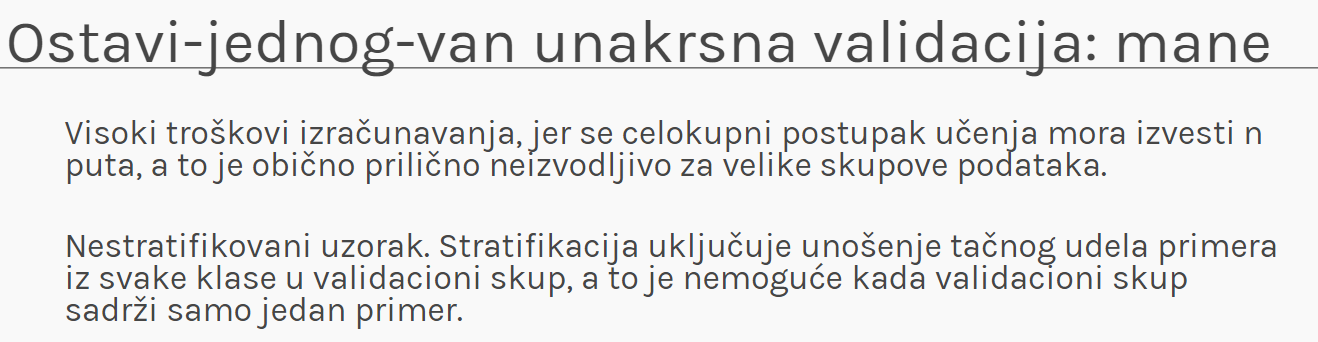
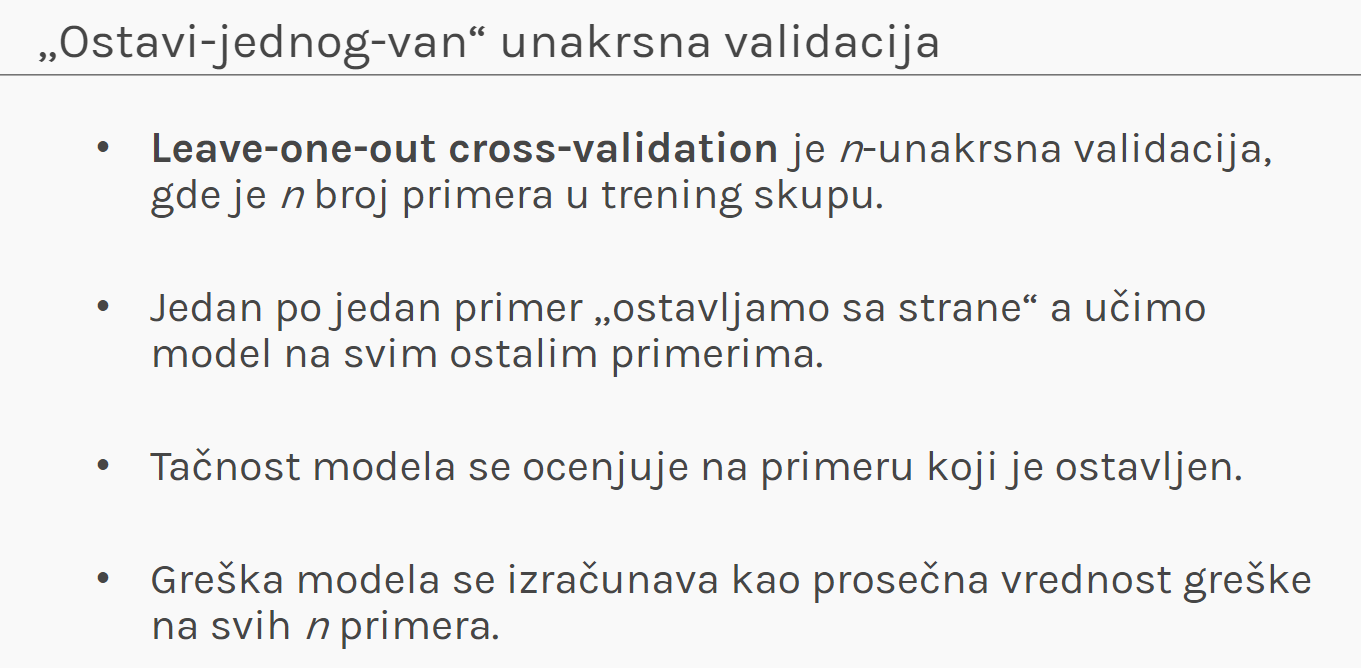
Trening skup služi za treniranje modela. Validacioni skup služi za odabir konkretnog modela sa određenim parametrima.Testni skup se koristi za procenu performansi potpuno formiranog modela.

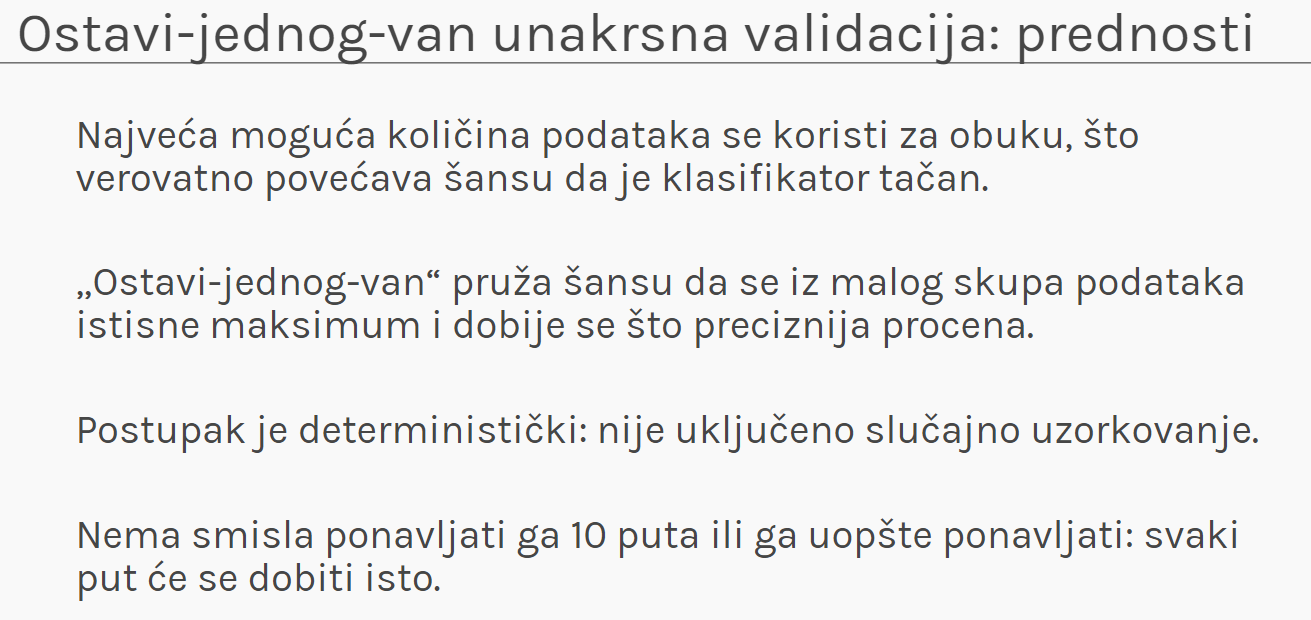
Trening skup (60% izvornog skupa podataka) se koristi za izgradnju modela. Naš algoritam pokušava da se prilagodi trening podacima.U ovoj fazi obično kreiramo više modela sa različitim vrednostima parametara ili korišćenjem različitih ML algoritama kako bismo uporedili njihove performanse tokom faze validacije. Validacioni skup podataka(20% izvornog skupa podataka) se koristi za poređenje performansi modela koji su formirani na osnovu trening skupa. Biramo model koji ima najbolje performanse. Testni skup(20% prvobitnog skupa podataka): Sada smo odabrali željeni model, ali još ne znamo kako će se ponašati na potpuno novim podacima iz stvarnog sveta. Dakle, primenjujemo odabrani model na test skupu da bismo videli kako će rezultate dati.

### K-unakrsna validacija.

Početni skup podataka za obučavanje se deli na 𝑘 delova ili podskupova. –Najčešće se uzima 10 podskupova. Potom se izvršava 𝑘 tura učenja. U svakoj turi se (𝑘−1)/𝑘 primera koristi za obučavanje, a ostalih 1/𝑘 primera se koristi za izračunavanje stope greške. Na kraju se ukupna greška izračunava kao prosečna greška u prethodnih 𝑘 slučajeva.

### Ostavi jednog van validacija





### Matrica konfuzije i metrike koje iz nje proizilaze.

U slučaju binarne klasifikacije jedna klasa se naziva pozitivnom, a druga negativnom. Stvarno pozitivni(true positive-TP) instance su pozitivne instance koje su od strane modela prepoznate kao pozitivne. Stvarno negativne(true negative-TN) instance su negativne instance koje su od strane modela prepoznate kao negativne. Lažno pozitivne(false positive-FP) instance su negativne instance koje su od strane modela proglašene pozitivnim. Lažno negativne(false negative-FN) instance su pozitivne instance koje su od strane modela proglašene negativnim.

A picture containing text, screenshot, font, number

Description automatically generated

Metrike:

1. Tačnost

A picture containing text, font, white, line

Description automatically generated

1. Preciznost – visoka cena lažno pozitivnih klasifikacija – primer spam

A picture containing font, white, number, text

Description automatically generated

1. Recall (Sensitivity, True positive rate) – visoka cena lažno negativnih – primer dijagnoza bolesti

A picture containing font, text, white, typography

Description automatically generated

1. F1 score

A picture containing text, font, white, line

Description automatically generated

1. Specificity

A picture containing font, text, white, calligraphy

Description automatically generated

1. False positive rate = 1 - Specificity

A white background with black text

Description automatically generated with low confidence

### ROC i AUC.

ROC grafikoni su dvodimenzioni grafikoni kod kojih je na x osi je prikazan False Positive Rate a na y osi prikazan Recall (Sensitivity, True positive rate).

Dijagonala y=x predstavlja strategiju slučajnog pogađanja klase. Klasifikatori koji se na njoj nalaze su dobri koliko i obično bacanje novčića. Klasifikatori koji se nalaze u donjem trouglu imaju gore performanse i od slučajnog pogađanja. Negacija takvog klasifikatora nalaziće se u gornjem trouglu.

Radi poređenja klasifikatora korišćenjem ROC krive možese koristiti jedna skalarna vrednost koja predstavlja očekivane performance klasifikatora. Uobičajena metoda određivanja te skalarne vrednosti je izračunavanje površine ispod ROC krive, skraćeno AUC. AUC klasifikator je ekvivalentan verovatnoći da će klasifikator rangirati slučajno izabranu pozitivnu instancu više od slučajno izabrane negativne instance.

### Obučavanje stabala odlučivanja.

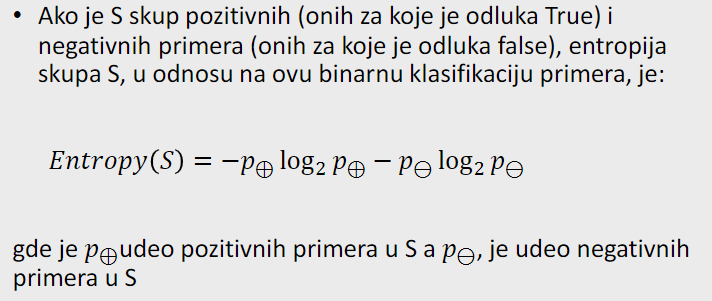
Stablo odlučivanja je dijagram oblika stabla koji se koristi za utvrđivanje toka akcija u procesu odlučivanja. Svaka grana predstavlja moguće odluke, pojave ili reakcije. U svakom čvoru odluke (Decision node)testira po jedan atribut. Čvor list (Leaf node): određuje vrednost ciljne funkcije (rezultat klasifikacije). Svaka grana (branch) koja polazi iz nekog čvora odgovara jednoj od mogućih vrednosti atributa koji je testiran u tom čvoru. Putanja odlučivanja: niz čvorova i grana od korenogčvora do jednog od listovakoji predstavlja konačnu odluku.

Obučavanje:

Treba izabrati najvažniji atribut – onaj koji je najkorisniji za klasifikaciju primera. Najkorisniji atribut je onaj koji vrši najbolju podelu primera na podskupove u kojima su samo pozitivni ili samo negativni primeri. Odabirom najkorisnijeg atributa se praktično kreira koreni čvor stabla. Primere dalje smeštamo u podstabla na osnovu njihovih vrednosti najvažnijeg atributa. Novi čvorovi stabla dalje se generišu rekurzivno na osnovu sledećeg „najboljeg atributa“.

### Entropija.Informaciono poboljšanje.

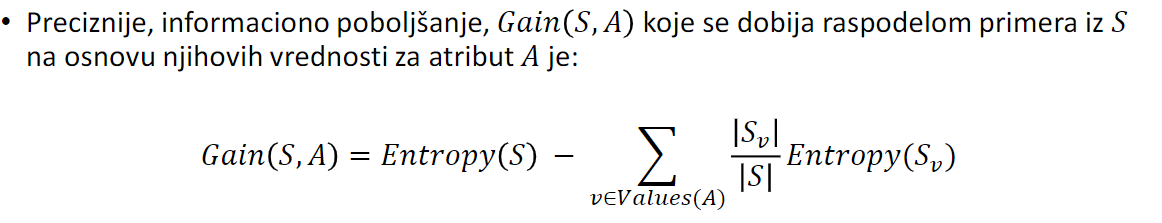
Selekcija trenutno najboljeg atributa u svakom koraku konstruisanja stabla odlukemože se vršiti korišćenjem mere koja se naziva Informaciono poboljšanje(Information gain). Meri koliko dobro dati atribut razdvaja trening primere prema ciljnoj klasifikaciji. Entropijom merimo homogenost (uređenost) skupa podataka.



A picture containing text, screenshot, font, algebra

Description automatically generated

Entropija je mera neizvesnosti kolekcije trening primera, koja nam omogućava da definišemo meru efikasnosti nekog atributa u klasifikovanju trening podataka. Ta mera, information gain, predstavlja očekivano umanjenje entropije nakon podele trening primera korišćenjem nekog atributa.



### Perceptron. Algoritam obučavanja perceptrona.

Trening skupovi sa dva atributa u kojima se grupe primera koji pripadaju dvema klasama mogu odvojiti pravom, nazivaju se linearno separabilni. Perceptron je jednostavan algoritam učenja koji može da odredi (nauči) hiperravan koja razdvaja dve klase primera u linearno separabilnim skupovima obuke.

A picture containing text, screenshot, diagram, line

Description automatically generated

Perceptron daje izlaz 1, kada su mu na ulazu date koordinate neke tačke koja se nalazi sa jedne strane hiper-ravni, a izlaz 0 kada je na ulazu tačka koja se nalazi sa druge strane hiper-ravni.

Da bi perceptron mogao da razdvaja bilo koje linearno separabilnepodatke prag mora biti neka konstanta 𝜃.

A picture containing text, screenshot, diagram, font

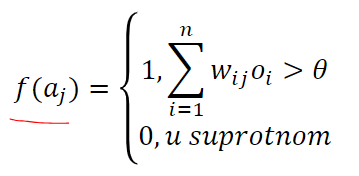
Description automatically generated

A picture containing text, screenshot, font, number

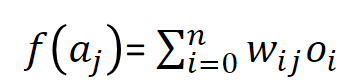
Description automatically generated

### Aktivacija neurona. Aktivaciona funkcija.

Funkcija neurona je jako jednostavna. Težinska suma ulaza u neuron propušta se kroz aktivacionu funkciju (ili funkciju transfera), koja određuje izlaz iz neurona. Prag aktivaciona funkcija.



Linearna aktivaciona funkcija.



ReLU aktivaciona funkcija. A picture containing font, text, handwriting, calligraphy

Description automatically generated

Sigmoid aktivaciona funkcija.

A picture containing font, line, white, text

Description automatically generated

### Mreže sa propagacijom unapred.

Propagacija unapred (forward propagation):

* Propagacija unapred je proces prenošenja ulaznih podataka kroz neuronsku mrežu kako bi se izračunale izlazne vrednosti.
* U ovom procesu, podaci se prenose od sloja do sloja, pri čemu svaki sloj obavlja određene matematičke operacije (npr. množenje sa težinama, primena aktivacione funkcije) nad ulaznim podacima kako bi generisao izlazne vrednosti.
* Propagacija unapred se odvija bez povratnih informacija i koristi se za izračunavanje izlaza neuronske mreže na osnovu trenutnih ulaznih podataka i parametara mreže.

### Način rada Backpropagation algoritma.

Propagacija unazad (backpropagation):

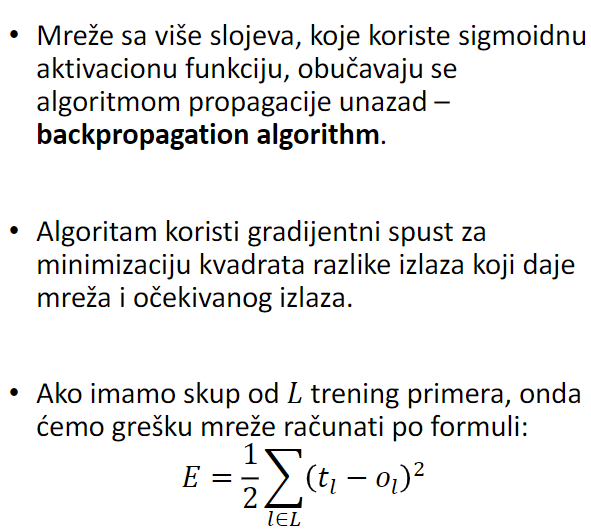
* Propagacija unazad je proces kroz koji se greška izlaznih vrednosti neuronske mreže propagira unazad kroz mrežu kako bi se prilagodili parametri i poboljšala tačnost predikcija.
* Nakon izračunavanja izlaznih vrednosti u procesu propagacije unapred, greška se računa upoređivanjem stvarnih izlaza sa očekivanim izlazima.
* Zatim, greška se propagira unazad kroz neuronsku mrežu, sloj po sloj, koristeći algoritam za propagaciju greške (kao što je unazadno propagiranje greške).
* Tokom propagacije unazad, težine između neurona se ažuriraju kako bi se minimizovala greška i poboljšala tačnost predikcija u budućnosti.

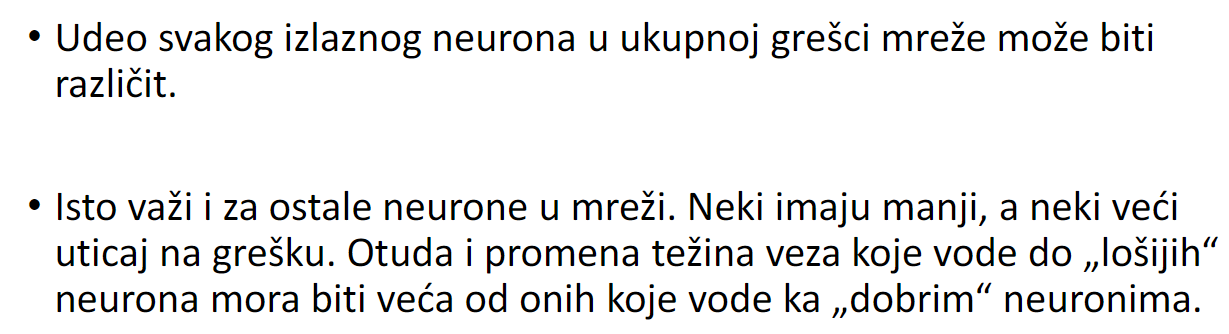
A picture containing text, font, screenshot, diagram

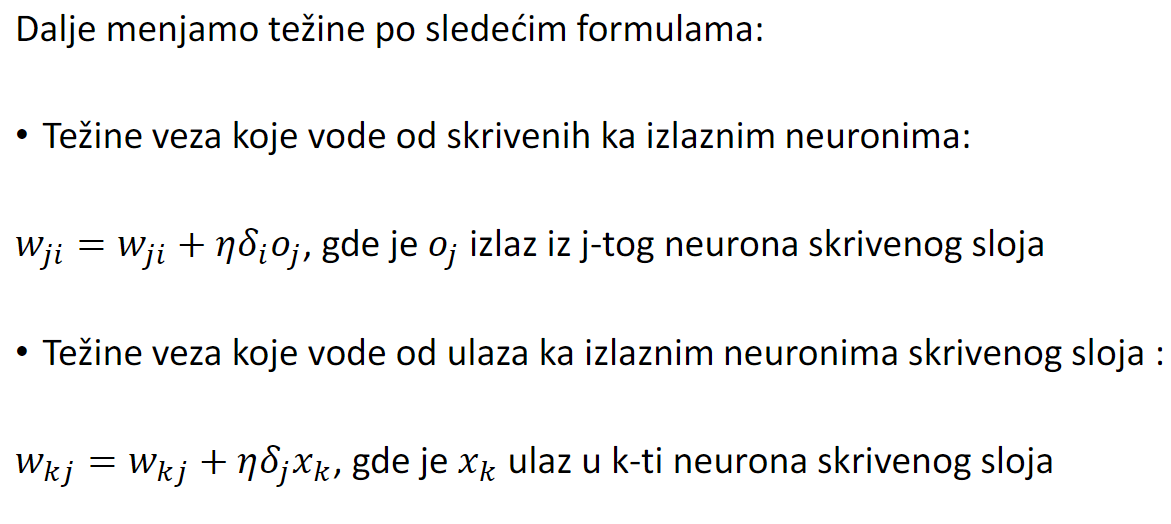
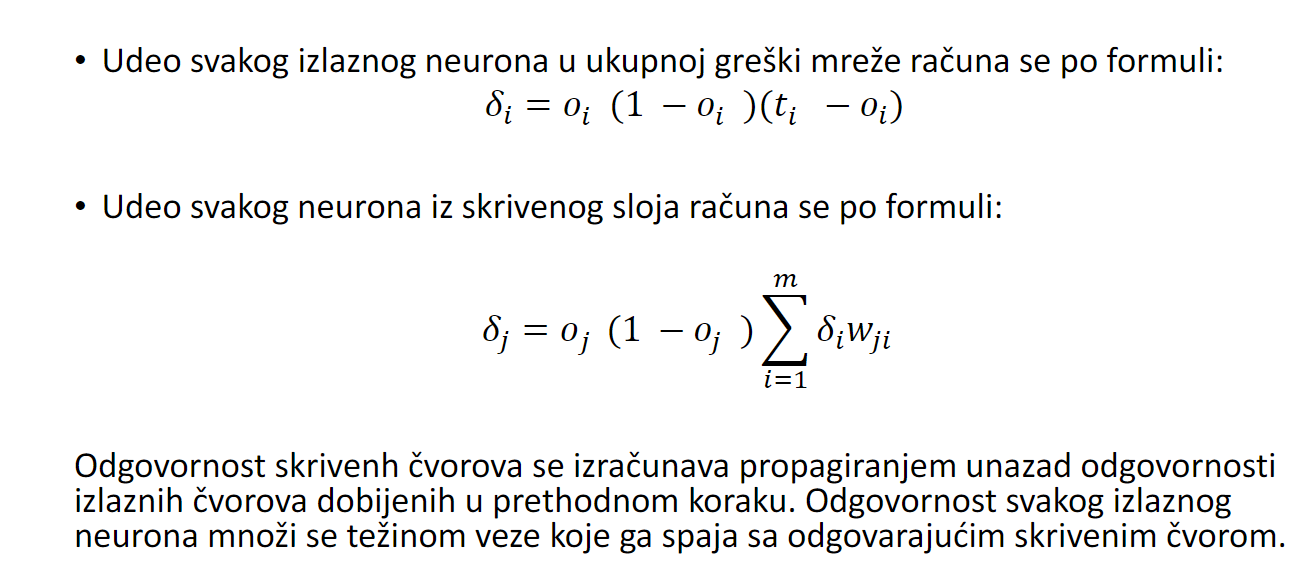
Description automatically generated

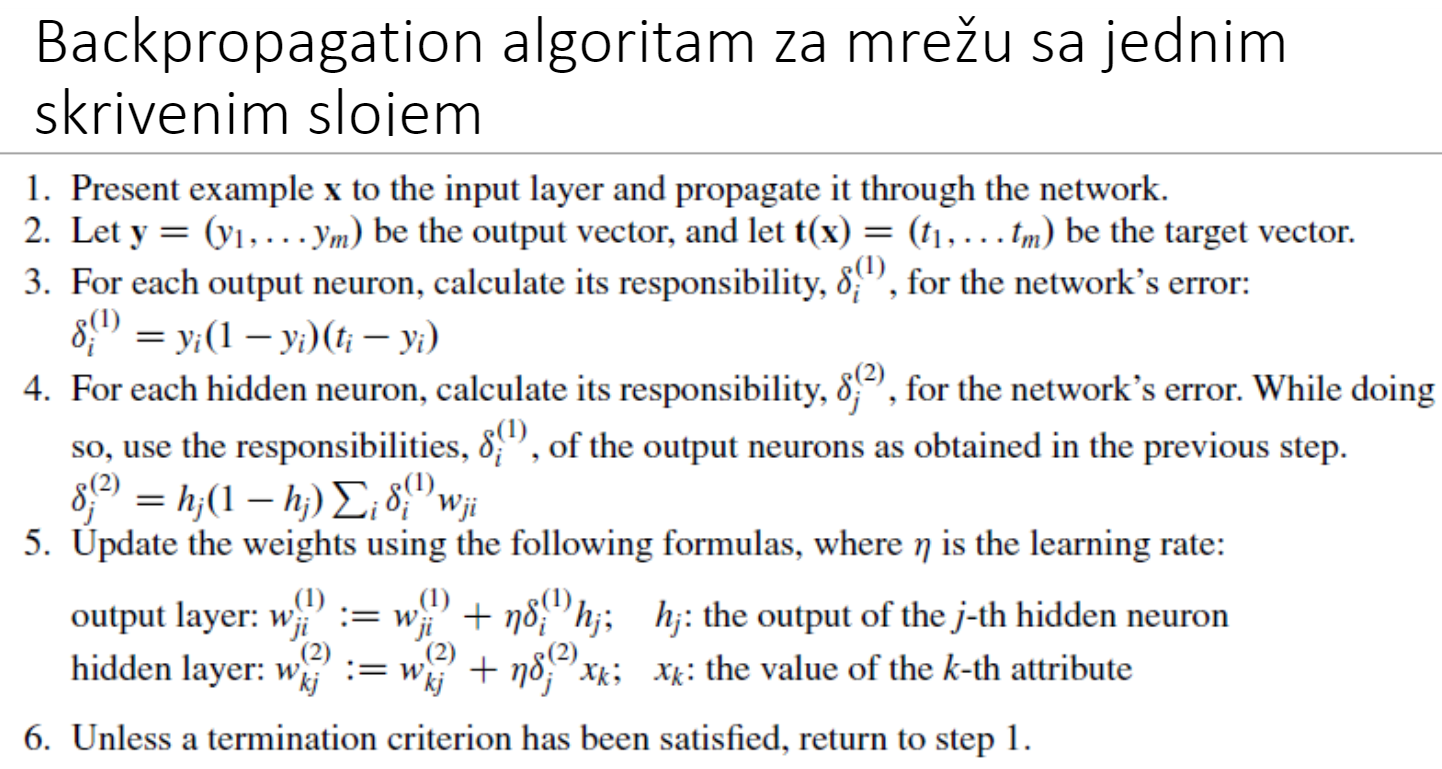
A picture containing text, screenshot, diagram, line

Description automatically generated









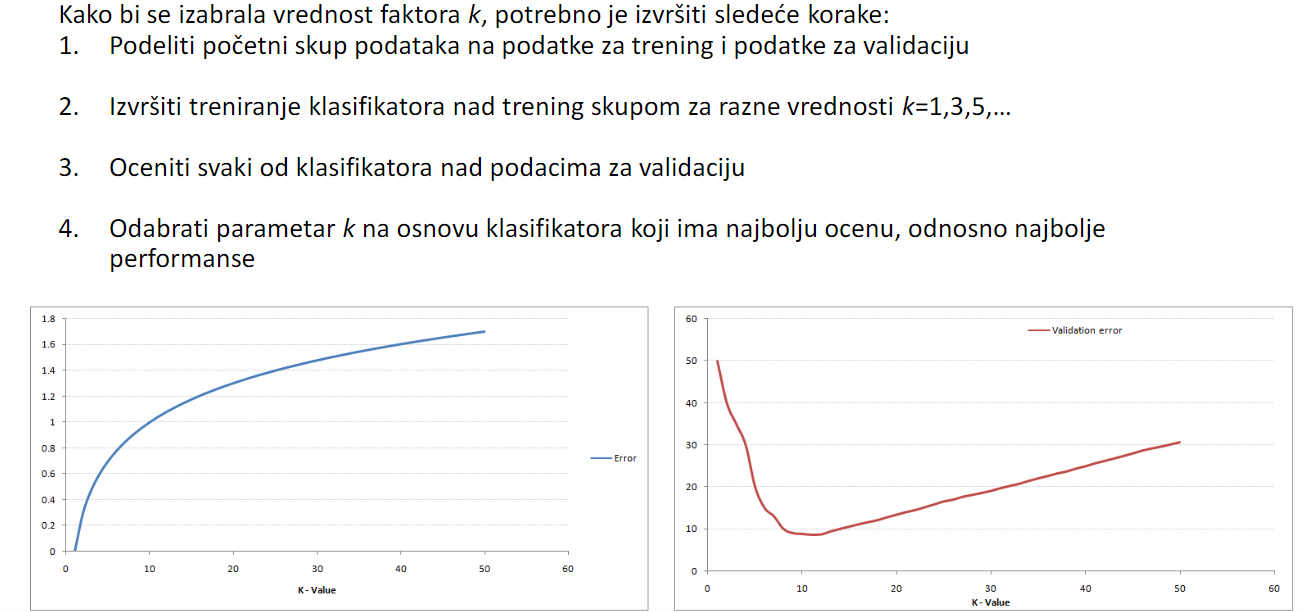
**POGLEDATI I PREZENTACIJU**

### kNN algoritam.

A picture containing text, screenshot, font, algebra

Description automatically generated

Granice između klasa zavise od faktora k.



A picture containing text, screenshot, font, algebra

Description automatically generated

### Linearna regresija

Linearna regresija se koristi za predviđanje kontinualnih izlaznih vrednosti na osnovu ulaza. Linearna regresija modelira odnos dve veličine određivanjem linearne jednačine kojom su te dve veličine povezane. Eksperimentom na osnovu ulaza generišemo izlaz i želimo da odredimo relaciju između ulaza i izlaza.

Hipoteza ℎ - Funkcija koja mapira 𝑥 u 𝑦. ℎ(𝑥)=𝑤0+𝑤1𝑥. Zadatak algoritma je odrediti w0 i w1 Rezidual je razlika između stvarne i prediktovane vrednosti izlaza.

A picture containing text, font, screenshot

Description automatically generated

A picture containing text, font, screenshot, line

Description automatically generated