Strojno učenje

(engl. *Machine Learning*)

Povijesno, razvoj ML započeo prije 50-tak god.:

Kako **indukcijom** izvesti **novo znanje** iz **primjera ili podataka**?

Razvoj umjetne inteligencije:

- Tradicionalna shema →

  program = algoritam + podaci.

- Nova shema: **ekspertno područno znanje** - temeljni oslonac u rješavanju problema.

Program = algoritam + podaci + znanje

Program = algoritam + podaci + znanje

Važno: djelotvornog automatiziranog **prikupljanja znanja to prvenstveno na temelju primjera**,

- izvođenja novog znanja.

uporaba znanja:
programer → inženjer baze znanja

Izuzetno kompleksan problem!
Primjer*: Kako kodirati znanje o igranju šaha?*
Što znači slaba struktura pješaka, dobro zaštićen kralj, itd…?
Kako znanje velemajstora oblikovati u ako-onda pravila?

Ideja: načiniti sustav koji će automatski stjecati znanje (visoko apstraktne koncepte ili strategije rješavanja problema) kroz iskustvo tj. primjere,
kao što to čini čovjek.

Zašto strojno učenje?

- Rastuće količine informacija
- Velika raspoloživa moć računala
- Potrebe tržišta
- Klasične metode nisu dovoljne ->
  Napredak u izgradnji algoritama i teorije

STROJNO UČENJE (engl. machine learning)

Strojno učenje odnosi se na izgradnju računarskih sustava koji automatski poboljšavaju svoje performanse kroz iskustvo (primjere).

Temeljni pojam u strojnom učenju:

indukcija, generalizacija

Cilj:
Naći opće pravilo koje objašnjava podatke ako je dan uzorak ograničene veličine

Pojam inteligencije obuhvaća postupak učenja →
ML okosnica AI !

Neke uspješne primjene strojnog učenja

1. Otkrivanje znanja u (velikim) skupovima podataka (engl. data mining)
2. Programske implementacije koje nije moguće riješiti klasičnim programiranjem
3. Prilagodljivi programski sustavi
4. Bioinformatika
5. Raspoznavanje govora
6. Raspoznavanje rukom pisanog teksta

1. Otkrivanje znanja u (velikim) skupovima podataka (engl. data mining)

Medicinska dijagnostika

Data:

Patient103 time=1 →	*Patient103* time=2 →	... →	*Patient103* time=n
Age: 23	Age: 23		Age: 23
FirstPregnancy: no	FirstPregnancy: no		FirstPregnancy: no
Anemia: no	Anemia: no		Anemia: no
Diabetes: no	Diabetes: YES		Diabetes: no
PreviousPrematureBirth: no	PreviousPrematureBirth: no		PreviousPrematureBirth: no
Ultrasound: ?	Ultrasound: abnormal		Ultrasound: ?
Elective C-Section: ?	Elective C-Section: no		Elective C-Section: no
Emergency C-Section: ?	Emergency C-Section: ?		**Emergency C-Section: Yes**
...	...		...

- 9714 zapisa, svaki opisuje tijek trudnoće i poroda pacijenta
- svaki zapis sadrži 215 značajki

Učenjem predvidjeti klasu pacijenata s visokim rizikom za hitan carski rez.

One of 18 learned rules:

```
If   No previous vaginal delivery, and
     Abnormal 2nd Trimester Ultrasound, and
     Malpresentation at admission
Then Probability of Emergency C-Section is 0.6
```

Analiza rizika kod dodjele kredita

Data:

Customer103: (time=t0)	*Customer103:* (time=t1)	...	*Customer103:* (time=tn)
Years of credit: 9	Years of credit: 9		Years of credit: 9
Loan balance: $2,400	Loan balance: $3,250		Loan balance: $4,500
Income: $52k	Income: ?		Income: ?
Own House: Yes	Own House: Yes		Own House: Yes
Other delinquent accts: 2	Other delinquent accts: 2		Other delinquent accts: 3
Max billing cycles late: 3	Max billing cycles late: 4		Max billing cycles late: 6
Profitable customer?: ?	Profitable customer?: ?		**Profitable customer?: No**
...	...		...

Rules learned from synthesized data:

```
If   Other-Delinquent-Accounts > 2, and
     Number-Delinquent-Billing-Cycles > 1
Then Profitable-Customer? = No
     [Deny Credit Card application]

If   Other-Delinquent-Accounts = 0, and
     (Income > $30k)  OR  (Years-of-Credit > 3)
Then Profitable-Customer? = Yes
     [Accept Credit Card application]
```


Ostali primjeri predviđanja...

Customer purchase behavior:

Customer103: (time=t0)	*Customer103:* (time=t1)	...	*Customer103:* (time=tn)
Sex: M	Sex: M		Sex: M
Age: 53	Age: 53		Age: 53
Income: $50k	Income: $50k		Income: $50k
Own House: Yes	Own House: Yes		Own House: Yes
MS Products: Word	MS Products: Word		MS Products: Word
Computer: 386 PC	Computer: Pentium		Computer: Pentium
Purchase Excel?: ?	Purchase Excel?: ?		**Purchase Excel?: Yes**
...	...		...

Customer retention:

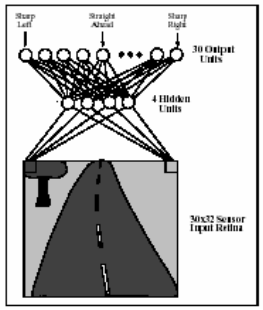
Customer103: (time=t0)	*Customer103:* (time=t1)	...	*Customer103:* (time=tn)
Sex: M	Sex: M		Sex: M
Age: 53	Age: 53		Age: 53
Income: $50k	Income: $50k		Income: $50k
Own House: Yes	Own House: Yes		Own House: Yes
Checking: $5k	Checking: $20k		Checking: $0
Savings: $15k	Savings: $0		Savings: $0
Current-customer?: yes	Current-customer?: yes		**Current-customer?: No**

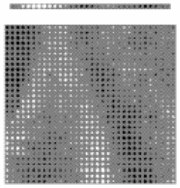
- prevencija “churn”-a.

2. Problemi previše složeni da bi se riješili na uobičajeni način

Računalom upravljano vozilo

ALVINN [Pomerleau] drives 70 mph on highways





Učenje strategija vožnje za autonomno upravljanje automobilom brzinom 70 m/h u duljini od 90 milja.

Slične tehnike koriste se za mnoge druge upravljačke probleme koji se temelje na senzorskim podacima.

Učenje se primjenjuje na različite baze podataka da bi se otkrile pravilnosti implicitno sadržane u podacima.

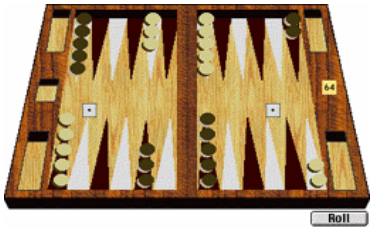
Učenje klasificiranja novih astronomskih struktura

NASA - stabla odluke korištena za klasifikaciju nebeskih objekata dobivena *Palomar Observatory Sky Survey* (Fayyad, et al. 1995). Sustav automatski klasificira sve objekte dobivene pregledom neba - 3 terabyta podataka slika.

Učenje se primjenjuje svuda gdje je potrebno efikasno pretraživati veliki prostor stanja.

3. Učenje igranja backgammon-a na razini svjetskog prvaka

Najuspješnije implementacije igara na računalu temelje se na učenju.



Najbolji svjetski program TD-GAMMON (Tesauro, 1992, 1995) je naučio strategiju igranja (na svjetskoj razini) na temelju više od 10^6 odigranih partija protiv samog sebe.

Raspoznavanje rukom pisanog teksta

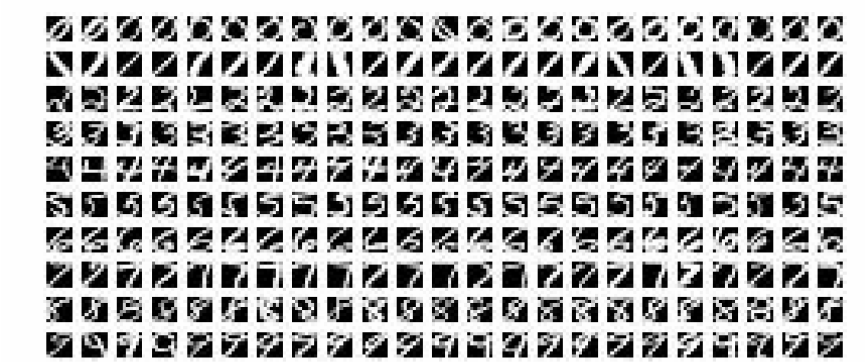
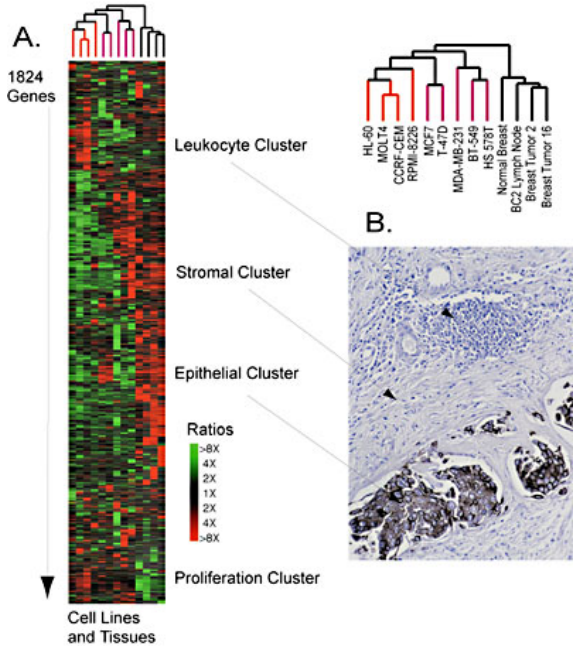


Fig. 3. Images of handwritten digits, normalized for horizontal and vertical scale and translation and sampled on an 8×8 pixel grid. Different writing angles introduce different levels of shearing in each image.

BIOINFORMATIKA



Zaključak:

Izuzetno široka primjena metoda strojnog učenje (umjetne inteligencije) na rješavanje složenih problema na sva područja ljudske djelatnosti!

Literatura:

- **Mitchell, T.: "*Machine Learning*", McGraw-Hill Comp., 1997.**
  (Prof. Tom M. Mitchell, Carnegie Mellon University, http://www.cs.cmu.edu/~tom)

1. Strojno učenje

Strojno učenje bavi se izgradnjom računarskih sustava koji automatski *poboljšavaju* svoje performanse kroz *iskustvo*.

Kakav učinak bi imao postizanje cilja?

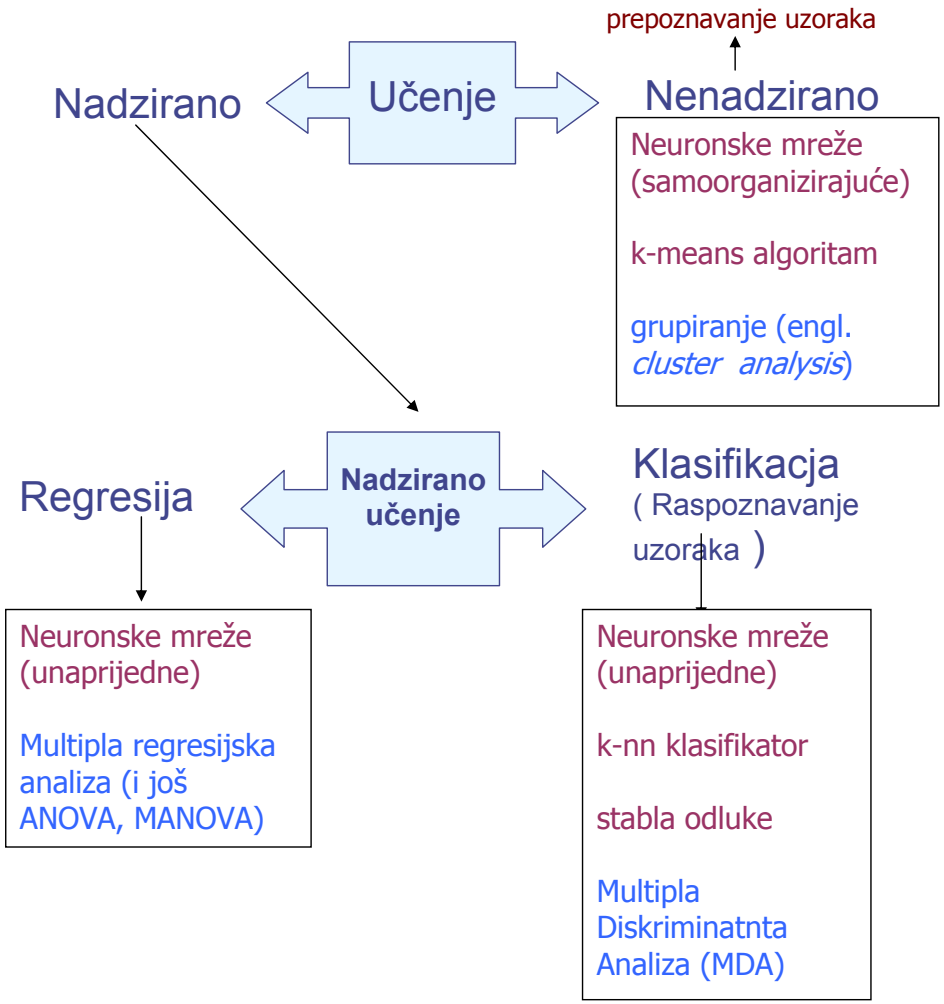
razumijevanje algoritama za strojno učenje ⟺ razumijevanje ljudske sposobnosti (ili ograničenja) za učenje.

Nema univerzalnog algoritma za učenje! – ipak izumljeni su efikasni algoritmi koji rješavaju određen tip problema (+ bolje teoretsko razumijevanje učenja).

Postoji puno uspješnih implementacija algoritama za učenje (primjeri…)

Napredak u teorijskim razmatranjima.

Vrste učenja



Podržano učenje (eng. reinforcement learning)

Tri najčešća oblika primjene algoritama strojnog učenja:

- **Otkrivanje znanja u skupovima podataka** – (engl. *data mining*) –
- **Programske implementacije koje nije moguće napraviti na klasičan način**

Programi koji se trebaju dinamički mijenjati i prilagođavati nekim uvjetima

Interdisciplinarnost strojnog učenja

Što slijedi ...

Danas: vrh ledenog brijega

Prva generacija algoritama: neuronske mreže, stabla odluka, regresija,...

Primjena dobro oblikovanih baza podataka

Obećavajuća industrija

Mogućnosti u budućnosti: ....

• 1.1. Dobro postavljeni problem učenja

Da bi dobro definirali problem učenja potrebno je definirati tri klase:

Zadatak T - ...
Mjera P – mjera uspješnosti koju povećavamo
Iskustvo E – izvor za stjecanje iskustva

Definicija

Za računalski program (sustav) kaže se da **uči** kroz iskustvo **E** u odnosu na neki skup zadataka **T** i s obzirom na neku mjeru uspješnosti **P**,
ako se povećava uspješnost obavljanja zadataka **T,** kroz iskustvo **E**, mjerena mjerom uspješnosti **P**.

Primjer:

Zadatak T: raspoznavanje i klasificiranje riječi pisanih rukopisom
Mjera P: postotak ispravno klasificiranih riječi
Iskustvo E: (primjeri za učenje) - baza podataka rukopisom napisanih riječi sa pripadnom klasifikacijom

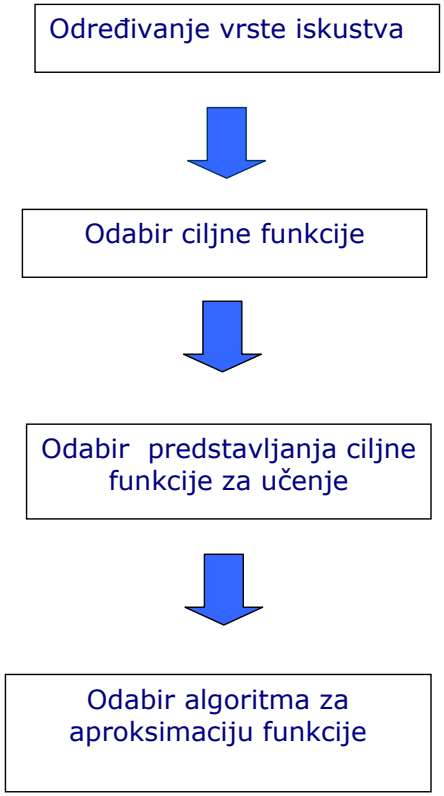
Primjer:

Zadatak T: vožnja 4-tračnom auto-cestom uz uporabu senzora vida
Mjera P: pređena prosječna udaljenost prije nastanka pogreške
Iskustvo E: niz slika i upravljačkih komandi snimljenih za vrijeme vožnje čovjeka

Vrlo općenita definicija učenja…

1.2 Oblikovanje sustava koji uči

Koraci u oblikovanju sustava koji uči:



1.2.1 **Odabir načina stjecanja iskustva (engl. *training experience*)**

Primjer:

Zadatak T: igra DAME
Mjera P: postotak pobjeda
Iskustvo E: igranje protiv samog sebe

primjeri za učenje

Odabir iskustva - značajan utjecaj na uspjeh ili neuspjeh učenja!

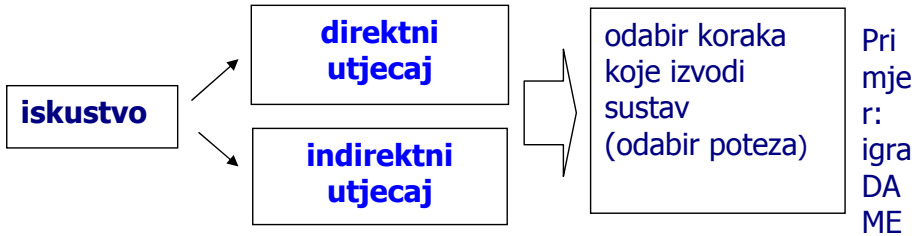
Tri važna svojstva iskustva:

1. Da li primjeri za stjecanje znanja imaju **direktni ili indirektni povratni utjecaj** na odabir koraka koje izvodi sustav?

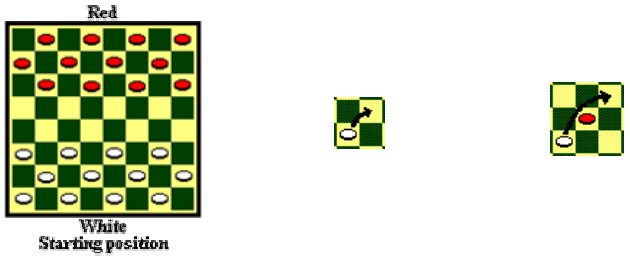
2. U kojoj mjeri **učenik kontrolira slijed primjera** za učenje?

3. Kako dobro primjeri za učenje predstavljaju **raspodjelu** one vrste **primjera** nad kojim će se u konačnosti mjeriti performanse sustava?

1. atribut iskustva



(*engl. checkers, draughts*)



Igra DAME – sustav uči:

- **direktno** – primjeri za učenje = pojedinačna stanja na ploči + pridružen ispravan korak.
- **indirektno** – primjeri za učenje = niz poteza + konačni ishod različitih igara.

Ispravnosti određenog poteza **IZVODI SE** (u ranoj fazi igre) indirektno iz činjenice da li je takva igra dobivena ili izgubljena.

Problem **pridjeljivanja ocjene** (*engl. credit assignment*) ili određivanja stupnja u kojem svaki potez u nizu zaslužuje nagradu ili kaznu za konačni ishod igre.

Pridjeljivanje ocjene -> težak problem (zašto?)

Dakle, **učenje izravnom povratnom vezom** *(engl. direct training feedback*) **je lakše oblikovati nego indirektno učenje**.

2.važan atribut iskustva (*engl. training expirience*) je stupanj u kojem učenik **kontrolira slijed primjera za učenje.**

Učitelj izabire informativne primjere
(npr. stanje na ploči) i daje korektan potez.

ILI

Učenik upravlja stanjima na ploči i indirektnom klasifikacijom primjera za učenje
(slučaj kada program igra protiv samog sebe).
Opcija: učenik eksperimentira s novim stanjima na ploči ili igra manje varijacije linije poteza koje smatra obećavajućim.

3. važan atribut skupa primjera za učenje

Reprezentativnost primjera za učenje.

Primjer: izvor iskustva: sustav igra protiv samog sebe.
Učenik možda nikada neće uočiti važna stanja na ploči koja bi mogla nastati u slučaju da sustav igra protiv čovjeka!

U teoriji strojnog učenja – važna pretpostavka:
raspodjela primjera za učenje = raspodjeli primjera za testiranje.
(To pravilo je često narušeno u praksi)

Važna odluka u oblikovanju našeg sustava: naš sistem će učiti igrajući sam protiv sebe. (Kakav je to tip iskustva?)
Prednost: nije potreban učitelj pa sistem može generirati proizvoljno primjera za učenje.

Primjer: Igra DAME

Zadatak **T** – igra DAME
Mjera uspješnosti **P**: postotak igara dobivenih na svjetskom turniru
Iskustvo **E**: igre igrane protiv samog sebe.

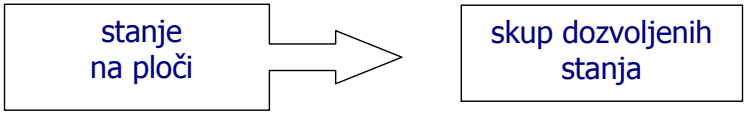
Do sada opisani izbori pri modeliranju:
- koji tip iskustva?

Da bi do kraja oblikovali sustav za učenje moramo odlučiti:
- koji točan tip znanja za učenje?
- kako predstaviti ciljno znanje?
- Koji algoritam učenje upotrijebiti?

1.2.2 Odabir ciljne funkcije

Koji tip znanja će se usvajati i
kako će ga koristiti sustav za igru?

Igra DAME:



Program mora naučiti kako izabrati najbolji potez iz skupa dozvoljenih poteza.

Optimizacijski problem. Ovakav zadatak predstavnik je široke klase zadataka: veliki prostor dozvoljenih stanja → poznat à priori, ali najbolja strategija nepoznata
(*primjer: raspoređivanje zadataka i upravljanje proizvodnim procesom*).

Što će program učiti?

Funkciju koja izabire najbolji potez iz skupa dozvoljenih poteza

Nazovimo tu funkciju *ChooseMove:*

$$ChooseMove: B \rightarrow M$$

B - skup stanja (situacija na ploči)
M – skup dozvoljenih poteza

Problem poboljšavanja uspješnosti P pri rješavanju zadatka T svodi se na učenje neke ciljne funkcije kao što je *ChooseMove*.

Ključni izbor prilikom dizajna je izbor ciljne funkcije.

ChooseMove je dobar izbor za ciljnu funkciju
ali
ChooseMove je teško učiti kada je na raspolaganju indirektno iskustvo

Alternativna ciljna funkcija je **evaluacijska funkcija** (**V**) - pridružuje numeričku vrijednost svakom pojedinačnom stanju.

$$\mathbf{V : B \rightarrow \Re}$$

B *skup svih dozvoljenih stanja,*
$\Re$ *skup realnih brojeva.*

Ako sistem može uspješno naučiti funkciju **V** tada
lako može izabrati najbolji potez u bilo kojoj situaciji na ploči.

Što treba biti vrijednost funkcije **V(b)** za neko stanje na ploči **b**?

Načelno, svaka funkcija koja pridružuje veću vrijednost boljim stanjima će nas zadovoljavati.

Odabrat ćemo jednu od **puno** mogućih definicija:

Definicija evaluacijske funkcije **V** za igru DAME
(jedna od puno mogućih):

1. Ako je b završno stanje na ploči i pobjeda
   tada **V(b)=100**
2. Ako je b završno stanje na ploči i poraz
   tada **V(b)=-100**
3. Ako je b završno stanje na ploči i nerješeno
   tada **V(b)=0**
4. Ako b nije završno stanje u igri tada **V(b) = V(b')**, gdje je b' najbolje završno stanje koje se može dosegnuti iz trenutnog stanja b, igrajući optimalno do kraja igre (uz pretpostavku da to čini i protivnik).

Ovakva rekurzivna definicija **V(b)** je ispravna,
ali
nije upotrebljiva za igru dame

(nije ju moguće efikasno računati, osim u 1.-3.)

Računanje vrijednosti **V** (kod 4.) podrazumijeva pretraživanje unaprijed optimalne linije igre sve do kraja igre.

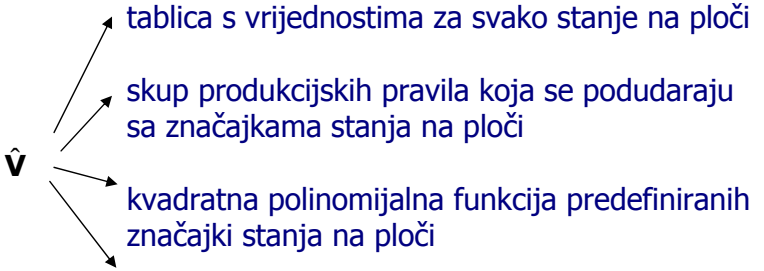
Tražimo operativnu definicije **V** !

U stvarnosti algoritam učenja radi samo sa aproksimacijom ciljne funkcije (označavamo je s $\hat{V}$).

Proces učenja ciljne funkcije naziva se
aproksimacija funkcije.

1.2.3 Izbor reprezentacije za ciljnu funkciju.

Kada smo odabrali idealnu ciljnu funkciju **V** – treba izabrati predstavljanje funkcije $\hat{V}$ za implementaciju i učenje.



Ekspresivnost funkcije $\hat{V}$ (što bliža V) **vs.** Količina podataka za učenje koja je potrebna za izbor između alternativnih hipoteza

Jedno moguće jednostavno rješenje:

$\hat{V}$ će se računati kao linearna kombinacija slijedećih značajki:
- x_1 – broj crnih pločica na ploči
- x_2 - broj crvenih pločica na ploči
- x_3 - broj crnih kraljeva na ploči
- x_4 - broj crvenih kraljeva na ploči
- x_5 - broj crnih pločica koje su ugrožene crvenima
- x_6 - broj crvenih pločica koje su ugrožene crnima

$$\hat{V}(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$$

Težinski faktori w_0 .. w_6 podešavaju se u postupku učenja.
w_0 .. w_6 – određuju važnost pojedinih značajki.

Do sada…

Specifikacija zadaće učenja

*Zadatak **T*** – igra DAME
*Mjera uspješnosti **P**:* postotak igara dobivenih na svjetskom turniru
*Iskustvo **E**:* igre igrane protiv samog sebe.

Implementacija - skup odabranih rješenja za model

Ciljna funkcija: $\mathbf{V} : B \rightarrow \Re$

Predstavljanje ciljne funkcije

$\hat{\mathbf{V}}(\mathbf{b}) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$

Važno: Učinak oblikovanja modela je reduciranje problema sa učenja strategije igre na učenje vrijednosti koeficijenata w_0 .. w_6.

1.2.4 Izbor algoritma za aproksimaciju funkcije

Učenje funkcije $\hat{\mathbf{V}}$ zahtijeva skup primjera za učenje, svaki primjer je par (b, V_{train}(b)).

Primjer:

((x_1 = 3, x_2 = 0, x_3 = 1, x_4 = 0, x_5 =0, x_6 = 0), +100)

Slijedi opis postupka kojim se:

- izvode primjeri za učenje iz indirektnog iskustva dostupnog učeniku u našem primjeru.
- podešavaju težine w_i tako da najbolje odgovaraju primjerima za učenje.

1.2.4.1 Procjena vrijednosti primjera za učenje

Igra DAME:

Dostupna informacija → da li je igra dobivena ili ne (indirektno iskustvo).

Potrebno → primjeri za učenje koji pridružuju numeričke vrijednosti pojedinim stanjima na ploči.

Završno stanje - lako pridjeljivanje vrijednosti,
(Brojna) među stanja - ?
(*dobivena ili izgubljena igra ne znači nužno da su svi potezi kroz igru loši ili dobri*)

Usprkos nejasnoćama u načinu procjene vrijednosti međustanja, jedan jednostavan pristup je dokazano uspješan.

Pravilo procjene vrijednosti V za učenje:

$$\mathbf{V}_{train}(b) \leftarrow \hat{\mathbf{V}}(sljedbenik(b))$$

učenikova tekuća aproksimacija funkcije V.

sljedbenik(b) – slijedeće stanje koje slijedi b i za kojeg je opet na potezu program.

Začuđujuće: tekuća verzija aproksimacije $\hat{\mathbf{V}}$ koristi se za procjenu vrijednosti primjera za učenje $\mathbf{V}_{train}(b)$, a zatim će ta vrijednost biti korištena za podešavanje same te funkcije $\hat{\mathbf{V}}$!

Intuitivno, procjene $\hat{\mathbf{V}}$ bit će preciznije što smo bliže završetku igre.

Pristup iterativnoj procjeni vrijednosti za učenje, temeljenoj na procjeni slijedećeg stanja, konvergira prema točnoj ocjeni V_{train}.

1.2.4.2 Podešavanje težina

TRAŽIMO - algoritam za određivanje w_i da bi najbolje aproksimirali *(engl. fit)* skup primjera za učenje $\{(b, V_{train}(b))\}$?

Jedan pristup određivanju **najbolje hipoteze (= skup težina, $w_0..w_6$)** jest da minimiziramo kvadrat pogreške E između vrijednosti za učenje $V_{train}(b)$ i vrijednosti dobivenih na temelju hipoteze $\hat{\mathbf{V}}(b)$.

$$E = \sum_{(b, V_{train}(b)) \in \textbf{Skupa primjera za ucenje}} (V_{train}(b) - \hat{V}(b))^2$$

Može se pokazati
minimizacija E ≡ nalaženju najvjerojatnije hipoteze,
za dani skup uzoraka za učenje.

Postoji više algoritama koji nalaze vrijednosti težinskih faktora na način da minimiziraju E.

Zahtjevi na algoritam za učenje:
- postupno podešavanje težina s novim primjerima i
- robustnost na pogreške u primjerima za učenje.

Jedan takav algoritam je **LMS** (**L**east **M**ean **S**quares).
Za svaki pojedini primjer korigiraju se težine u nekom manjem iznosu, u smjeru koji smanjuje pogrešku.

Taj se algoritam može shvatiti kao pretraživanje prostora mogućih hipoteza (težinskih koeficijenata) uz stohastički gradijentni silazak tako da se minimizira pogreška E.

LMS algoritam

Za svaki primjer za učenje (b, $V_{train}(b)$)

- Izračunaj $\hat{\mathbf{V}}(b)$ uz trenutne vrijednosti težinskih koeficijenata
- Za svaki težinu w_i, podesi

$$w_i = w_i + \eta\,(V_{train}(b) - \hat{\mathbf{V}}(b))\,x_i$$

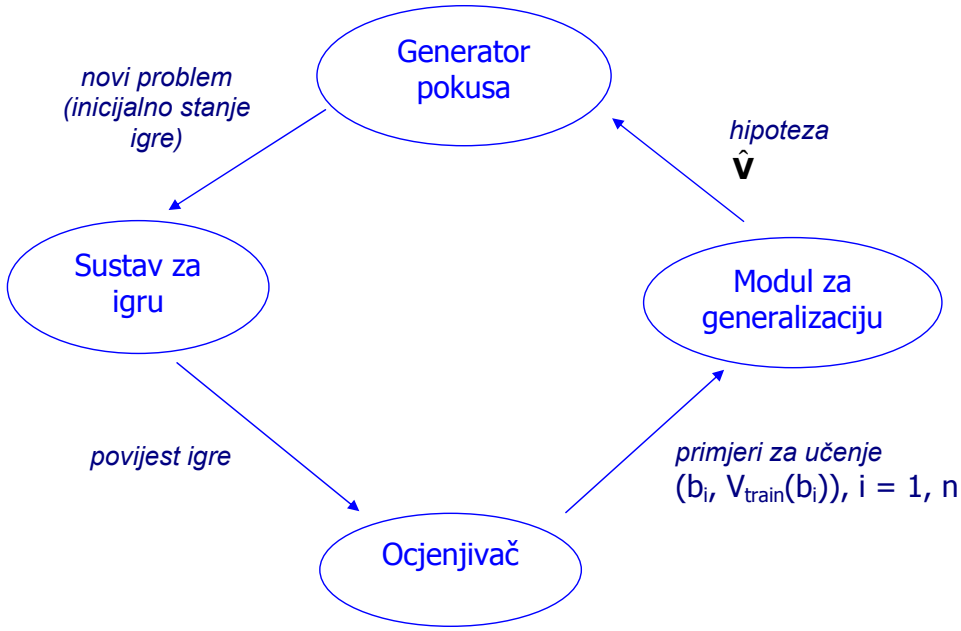
η konstanta (npr. 0.1) oblikuje veličinu korekcije težina.

Ako je $(V_{train}(b) - \hat{\mathbf{V}}(b)) = 0$ tada nema korekcije.

1.2.5 Završno oblikovanje

Program koji uči igrati DAME može biti opisan sa 4 programska modula koji predstavljaju centralnu komponentu u većini sustava strojnog učenja.

1. Sustav za igru
2. Ocjenjivač
3. Modul za generalizaciju
4. Generator pokusa



- **Sustav za igru** (engl. ***Performance System***) modul koji rješava zadanu zadaću igre.

  Ulaz: instanca nove situacije (nove igre)
  Izlaz: trag svojih odluka
  (rješenja tj. proizvodi povijest igre).

  U našem slučaju, sustav za izvođenje izabire slijedeći potez u svakom koraku na temelju naučene evaluacijske funkcije $\hat{\mathbf{V}}$. Dakle, očekujemo poboljšanje ponašanja s povećanjem točnosti funkcije $\hat{\mathbf{V}}$.

- **Ocjenjivač (**engl. ***Critic*****)**

  Ulaz: povijest (trag) igre
  Izlaz: primjeri za učenje ciljne funkcije

  $$\mathbf{V}_{train}(b) \leftarrow \hat{\mathbf{V}}(sljedbenik(b))$$

- **Modul za generalizaciju (**engl. ***Generalizer*****)**

  Ulaz: primjeri za učenje
  Izlaz: hipoteza – procjena ciljne funkcije $\hat{\mathbf{V}}$.

  Modul poopćuje sa pojedinačnih slučajeva (primjeri za učenje) postavljajući hipotezu o općenitoj funkciji koja pokriva cijeli skup primjera za učenje i iznad njih.
  U našem slučaju to je LMS algoritam, a izlazna hipoteza je funkcija $\hat{\mathbf{V}}$ zadana sa $w_0..w_6$.

- **Generator pokusa (**engl. ***Experiment Generator*****)**

  Ulaz: trenutna hipoteza (trenutno naučena funkcija cilja)
  Izlaz: novi problem (npr. novo početno stanje ploče) koji će maksimizirati učenje cijelog sustava.

  U našem primjeru: *početno stanje na ploči*.
  Sofisticirana strategija: inicijalna stanja su neka odabrana među-stanja (posebno oblikovana s ciljem da istražuju neka određene dijelove prostora stanja).

Izbori koje smo donijeli tijekom oblikovanja sustava za DAME čine poseban slučaj za

- modul igre,
- modul ocjenjivač,
- modula za generalizaciju i
- generatora pokusa.

Mnogi sustavi za strojno učenje mogu biti opisani na ovaj način .

Slijed mogućih izbora pri oblikovanju sustava strojnog učenja