Modeli za predstavljanje slike i videa

Strojno učenje i duboko učenje

Siniša Šegvić

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

Sadržaj

- **Motivacija**:
  - umjetna inteligencija
  - računalni vid
- Pregled osnova **strojnog učenja**:
  - osnovni pojmovi i tehnike
  - primjeri algoritama
- Suvremeni izazovi i prednosti dubokog učenja

Motivacija: strojno učenje

- **Strojno učenje**: proučava algoritme čija funkcionalnost ovisi o slobodnim parametrima koji se štimaju na primjerima za učenje
  - jedan od središnjih problema umjetne inteligencije
- **Umjetna inteligencija**: proučava izradu strojeva koji "misle":
  - zadatci koji su laki za ljude, a vrlo teški za računala
- Sofisticirano ponašanje često je lakše naučiti nego isprogramirati:
  - primjer: program koji pronalazi sliku s kravom

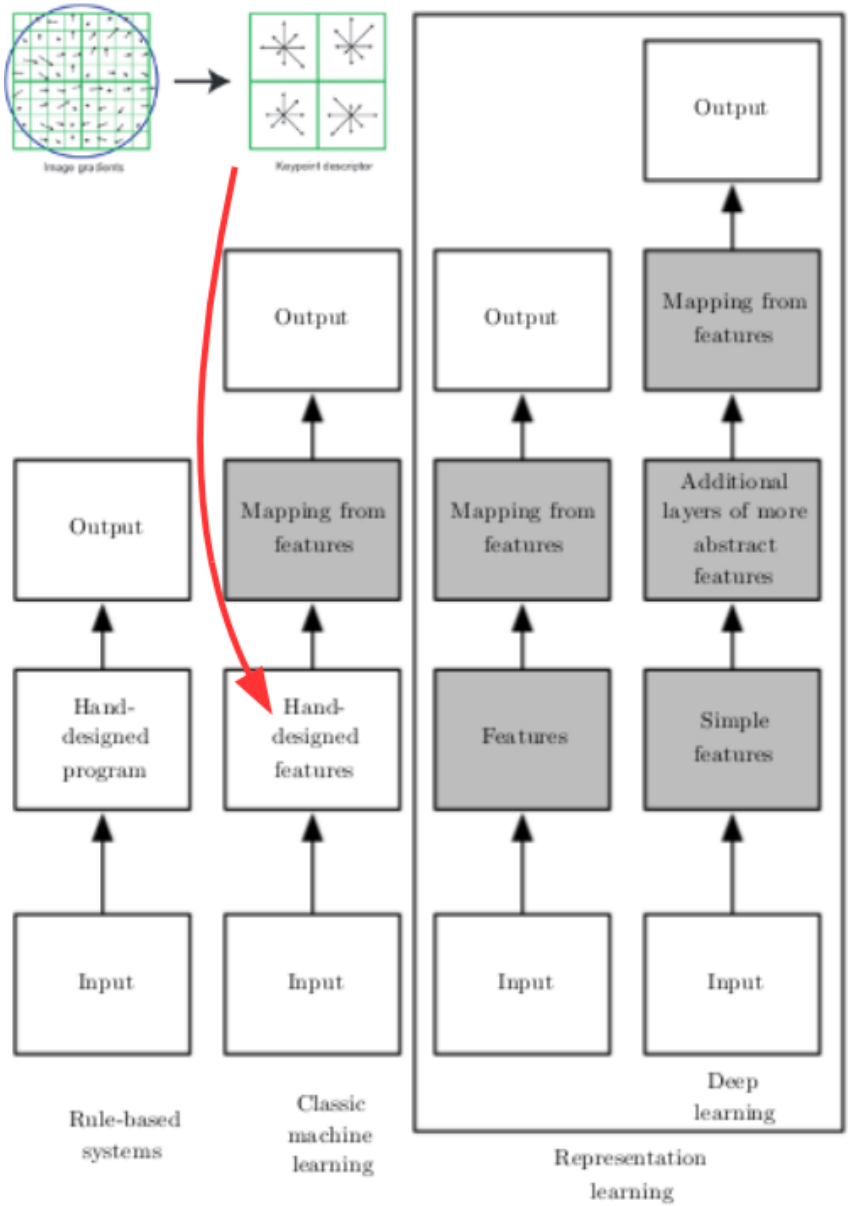
Motivacija: umjetna inteligencija

Odnos AI vs ML:

- rani pristupi: nema učenja
- klasični pristupi: naučiti plitki model na konstruiranim značajkama
- reprezentacijski pristupi: naučiti i značajke i model
- duboki pristupi: odjednom naučiti (engl. end to end) niz transformacija

Učenje postaje sve važnije!

Podatkovne reprezentacije isto!

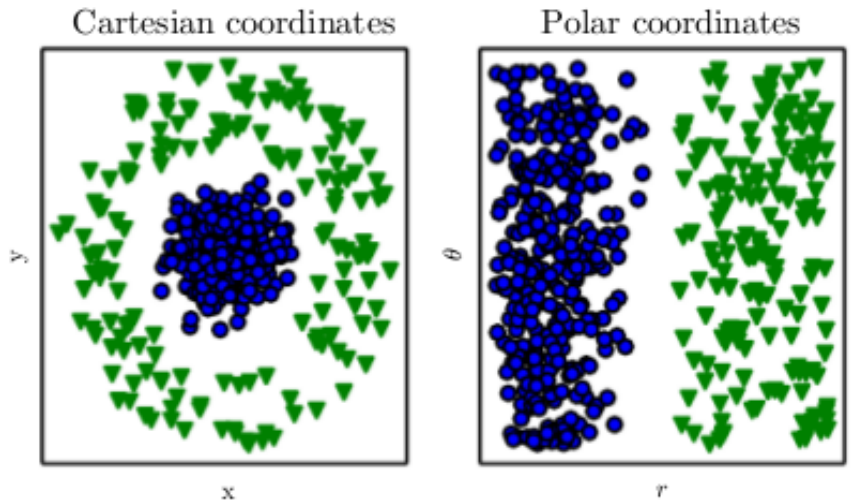


[goodfellow16]

Motivacija: podatkovne reprezentacije

U računarstvu, težina obrade često ovisi o reprezentaciji podataka:

- premalo značajki -> nema dovoljno informacije
- previše značajki -> opasnost od prenaučenosti
- za dani zadatak, jedne značajke mogu biti pogodnije od drugih
  - koliko je MCMLXXI + XIX?
  - a 1971 + 19?



[goodfellow16]

- kod teških zadataka značajke nije lako konstruirati ručno
- učenje podatkovnih reprezentacija može olakšati težak problem

Motivacija: duboko učenje

- Dominantni pristup strojnom učenju 1990-2006: ručno konstruirane značajke i plitki klasifikatori s konveksnim gubitkom.
- U to vrijeme, višeslojni modeli nisu bili popularni:
  - \+ veliki reprezentacijski potencijal
  - – teoretska podloga (konvergencija nije zagrantirana)
  - – performansa u praksi.
- Danas duboki modeli na mnogim zadatcima postižu najbolje rezultate.
- Što se dogodilo?
  - ogromni skupovi za učenje (milijuni ručno označenih slika)
  - optimirane matrične operacije (posebno GPU, ali i CPU)
  - nove tehnike učenja (upoznat ćemo ih na ovom kolegiju).

Motivacija: no free lunch

Duboki modeli nisu "radili" kako treba jer nismo imali:

- tehnike koje pospješuju konvergenciju
- dovoljno podataka da ih naučimo
- dovoljno jake strojeve da dočekamo konvergenciju

Međutim, to ne pojašnjava zašto bi duboki modeli bolje generalizirali od drugih modela velikog kapaciteta (kSVM, stabla, ...)

Performansa na neviđenim podatcima ovisi o pristranosti algoritma:

- pretpostavke modela, gubitka ili optimizacijskog postupka
  - ako te pretpostavke odgovaraju podatcima - performansa algoritma će zadovoljavati (no free lunch theorem) [domingos12cacm]
  - logistička regresija: dobar odabir za linearno razdvojive razrede
- u računalnom vidu koristimo duboke modele jer je njihova pristranost prikladna za analizu slika našeg svijeta

Pregled strojnog učenja: osnovni pojmovi

Strojno učenje proučava oblikovanje postupaka čija se uspješnost poboljšava s iskustvom

Algoritam strojnog učenja obuhvaća sljedeća dva postupka:

- postupak obrade podataka (model): $y = f(x|\theta)$
- postupak štimanja slobodnih parametara modela (metoda optimizacije)

Pregled strojnog učenja: model

Nužno je zadati mjeru performanse (učinka) odnosno mjeru pogreške modela:

- npr. točnost klasifikacije na **ispitnom skupu** $\{\mathbf{x}_i, y_i\}$
- npr. površina ispod ROC ili PR krivulje

Razlikujemo empirijsku i generalizacijsku performansu/pogrešku

- empirijska pogreška na ispitnom skupu je procjena generalizacijske pogreške

Kapacitet modela: složenost relacije koju model može opisati

- laka (i gruba) procjena kapaciteta: broj parametara
- jači modeli mogu upiti više podataka
- slabije modele teže je prenaučiti

Pregled strojnog učenja: optimizacija

Štimamo θ tako da na **skupu za učenje** "što ćešće" imamo $f(\mathbf{x}_i) = y_i$

Skup za učenje mora biti uzorkovan iz iste distribucije kao i ispitni skup!

Nužno zadati kriterij optimizacije (gubitak)

- idealno: gubitak == pogreška modela
- u praksi, često biramo **derivabilne** nadomjestke (engl. proxy)
- npr. negativna log-izglednost parametara na skupu za učenje: $\mathcal{L}(\theta) = \sum_i -\log P(y_i|\mathbf{x}_i, \theta)$

Primjer: učimo robota da igra košarku

- pogreška modela: E = 1 - postotak šuta
- derivabilni proxy: L = $\sum_i \delta_i^2$

Pregled strojnog učenja: posredna optimizacija

Specifičnost strojnog učenja: performansu optimiramo posredno

- optimizacijska metoda ne "vidi" gubitak na ispitnom skupu
  - tipično pretpostavljamo da podatke za učenje i ispitivanje generira isti slučajni proces
- najviše čemu se možemo nadati:
  - gubitak proporcionalan empirijskoj pogrešci
  - struktura modela podržava dobru generalizaciju na stvarnim podatcima
- u praksi, gubitak se često ne može poistovijetiti s empirijskom pogreškom
  - tipično zato što formulacija pogreške nije derivabilna
  - tada kažemo da je gubitak zamjena (engl. proxy) za empirijsku pogrešku

Pregled strojnog učenja: posredna optimizacija - primjer

Promotrimo gubitak izražen negativnom log-izglednošću parametara:
$\mathcal{L}(\theta \mid \mathbf{Y}, \mathbf{X}) = -\sum_i \log P(Y = y_i \mid \mathbf{x}_i)$

Neka Y može poprimiti vrijednosti c_0 do c_4, te neka u skupu za učenje imamo $(\mathbf{x}_i, c_2)$ i $(\mathbf{x}_j, c_2)$.

Pretpostavimo da model u i-tom podatku vraća:
$P(Y \mid \mathbf{x}_i) = [\ 0.06 \quad 0.14 \quad 0.7 \quad 0.1 \quad 0.0\ ]^\top$

- podatak $\mathbf{x}_i$ doprinosi gubitku (0.36) iako je ispravno klasificiran!

Pretpostavimo da model u j-tom podatku vraća:
$P(Y \mid \mathbf{x}_j) = [\ 0.04 \quad 0.5 \quad 0.01 \quad 0.18 \quad 0.27\ ]^\top$

- podatak $\mathbf{x}_j$ ostvaruje ogroman doprinos gubitku (4.6)
- može biti problem ako imamo pogrešno označavanje ili ako je $\mathbf{x}_j$ podatak koji odskače (engl. outlier)

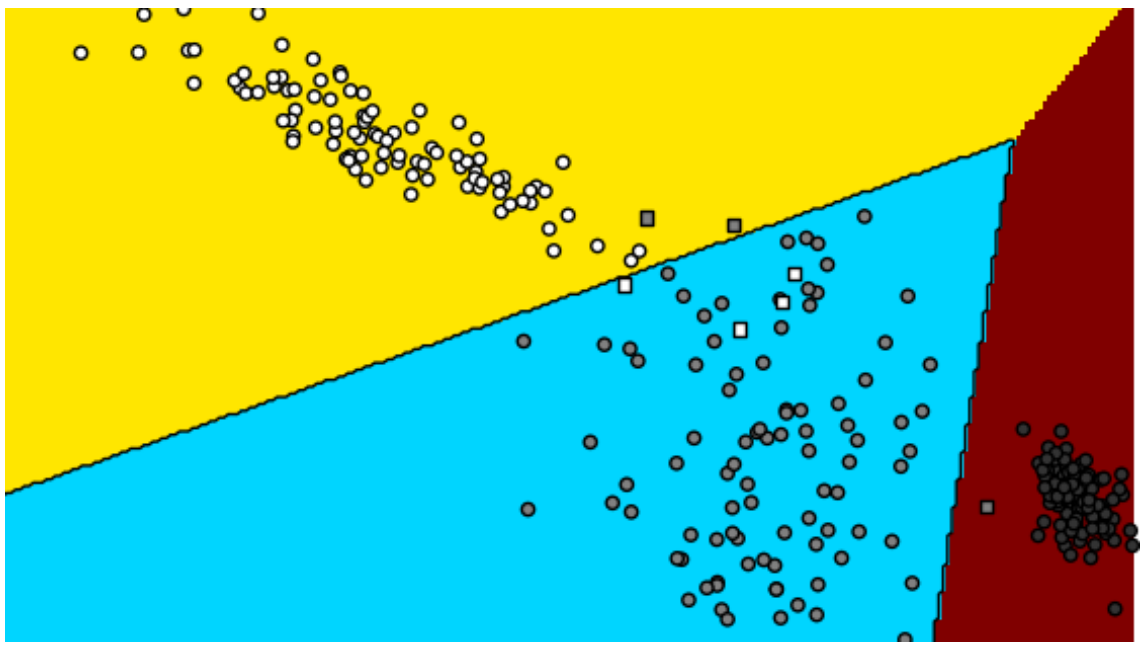
Pregled strojnog učenja: definicije

- Algoritam strojnog učenja definiran je:
  - modelom: postupak obrade sa slobodnim parametrima
  - gubitkom: formalizacija (anti-)dobrote parametara modela
  - metodom optimizacije: način pronalaženja parametara koji minimiziraju gubitak.
- S obzirom na dostupnost podataka za učenje, razlikujemo:
  - nadzirano učenje: željeni izlaz modela dostupan u svakom podatku
    - ◊ tipični zadatci: klasifikacija, regresija
  - nenadzirano učenje: dostupni su samo podatci
    - ◊ tipični zadatci: grupiranje, estimiranje gustoće, smanjenje dimenzionalnosti.
  - podržano (ojačano) učenje: dostupna je povratna veza o kvaliteti međudjelovanja s okolinom.

Pregled strojnog učenja: primjer

Logistička regresija: nadzirana klasifikacija u više razreda

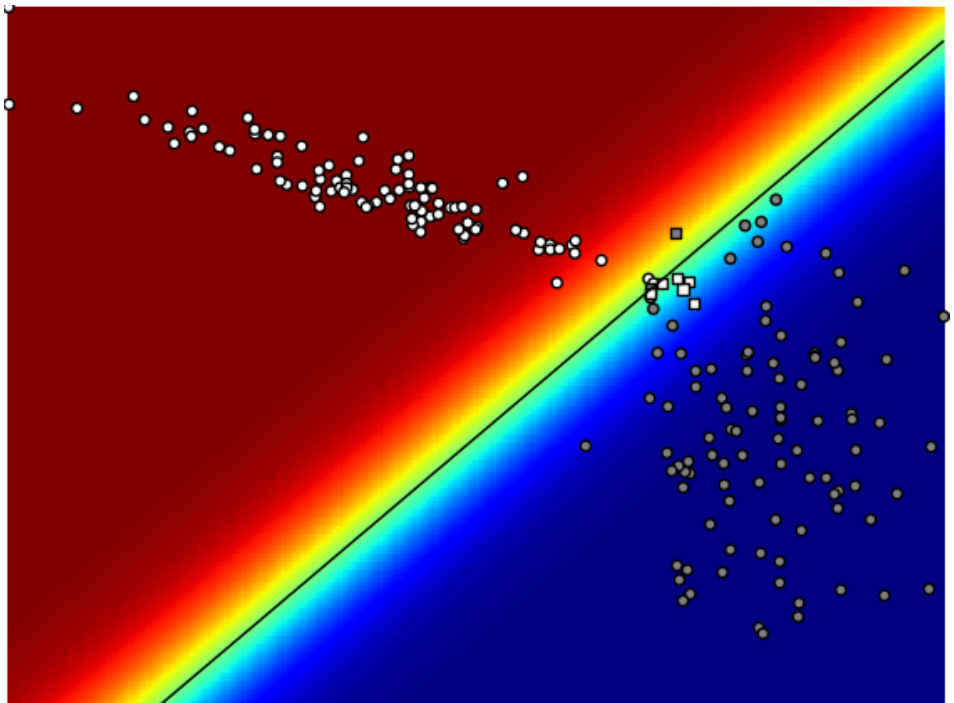
- model vraća aposteriornu distribuciju vjerojatnosti razreda:
  $P(Y \mid \mathbf{x}) = \mathrm{softmax}(\mathbf{w}^\top \mathbf{x} + b)$, $\mathrm{softmax}(\mathbf{x}) = [e^{s_j} / \sum_k e^{s_k}]^\top$.
- gubitak (negativna log-izglednost modela na skupu za učenje):
  $\mathcal{L}(\mathbf{w}, b \mid \mathbf{Y}, \mathbf{X}) = -\sum_i \log P(Y = y_i \mid \mathbf{x}_i)$
- optimizacijski postupak (gradijentni spust):
  $\mathbf{w}_{i+1} = \mathbf{w}_i - \delta \cdot \nabla_{\mathbf{w}_i} \mathcal{L} = \mathbf{w}_i - \delta \cdot (\frac{\partial \mathcal{L}}{\partial \mathbf{w}_i})^\top$



Pregled strojnog učenja: primjer 2

Stroj s potpornim vektorima (nadzirana binarna klasifikacija):

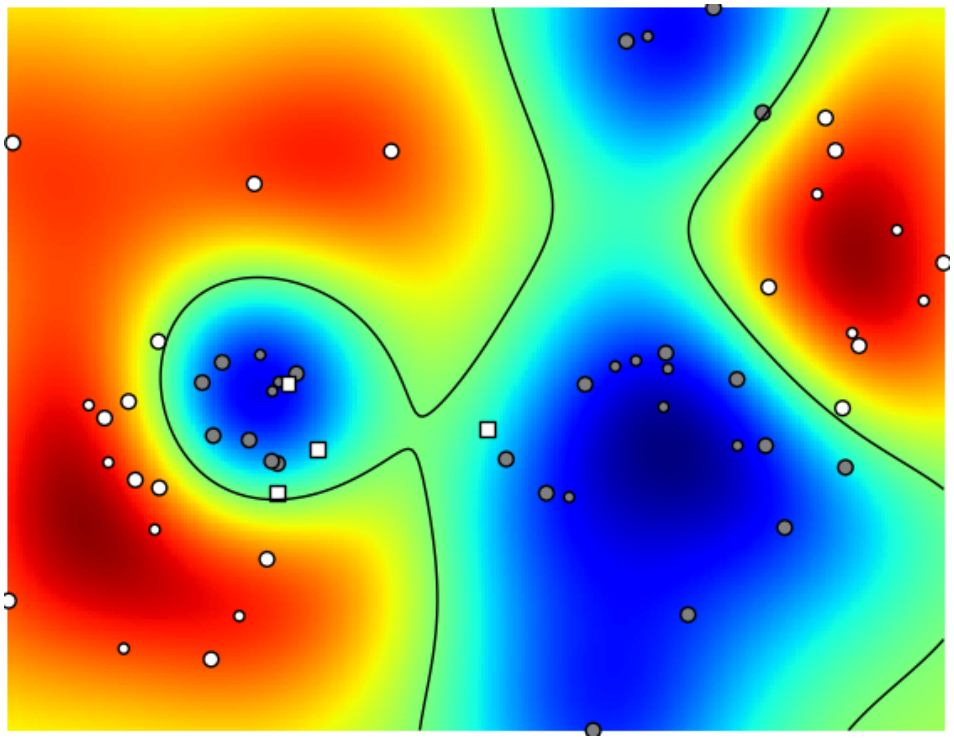
- model (pripadnost podatka): $f(\mathbf{x}) = \begin{cases} c_0 & \text{ako } \mathbf{w}^\top \mathbf{x} + b < 0 \\ c_1 & \text{ako } \mathbf{w}^\top \mathbf{x} + b > 0 \end{cases}$
- gubitak (ukupna povreda margine plus regularizacija):
  $\mathcal{L}(\mathbf{w}, b \mid \mathbf{Y}, \mathbf{X}) = \lambda \|w\|^2 + \frac{1}{n} \sum_{i=1}^{n} \max\left(0, 1 - (-1)^{[\![y_i = c_1]\!]} (\mathbf{w}^\top \mathbf{x}_i + b)\right)$
- optimizacija: kvadratno programiranje ili gradijentni spust



Pregled strojnog učenja: primjer 3

Stroj s potpornim vektorima i jezgrenom funkcijom:

- model (pripadnost podatka): $f(\mathbf{x}) = \begin{cases} c_0 & \text{ako } \sum_i \alpha_i \cdot k(\mathbf{x}_i, \mathbf{x}) < 0 \\ c_1 & \text{ako } \sum_i \alpha_i \cdot k(\mathbf{x}_i, \mathbf{x}) > 0 \end{cases}$
- gubitak (ukupni prijestup margine + regularizacija): $\mathcal{L}(\mathbf{w}, b \mid \mathbf{Y}, \mathbf{X}) = \lambda \|w\|^2 + \frac{1}{n} \sum_{i=1}^{n} \max\left(0, 1 - (-1)^{[\![y_i = c_1]\!]}(\alpha_i \cdot k(\mathbf{x}_i, \mathbf{x}) + b)\right)$
- optimizacija: kvadratno programiranje ili gradijentni spust



Pregled strojnog učenja: kapacitet

Kapacitet je osnovno svojstvo modela:

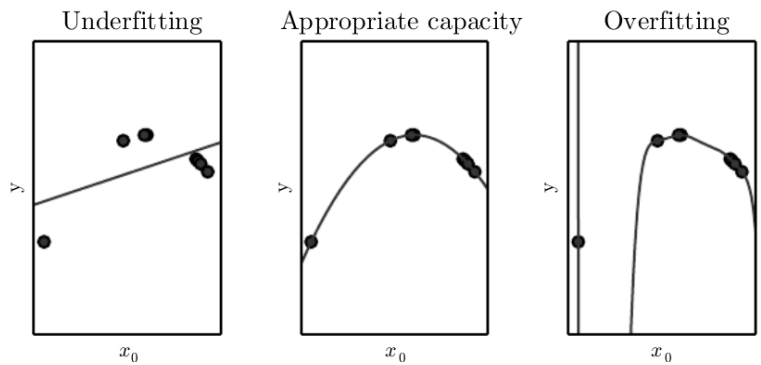
- opisuje sposobnost prilagođavanja podatcima
- najčešće je proporcionalan broju stupnjeva slobode modela
- formalno može se mjeriti brojem podataka (VC dimenzija) koje model može **razbiti** (tj. objasniti uz proizvoljno označavanje)

Modeli malog kapaciteta skloni su podnaučenosti:

- ne postoji skup parametara koji može objasniti skup za učenje.

Modeli velikog kapaciteta skloni su prenaučenosti:

- previše skupova parametara objašnjavaju skup za učenje.

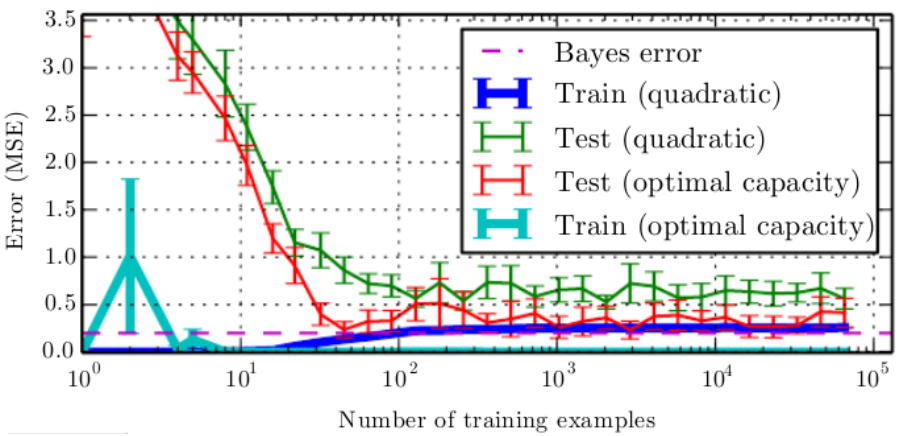


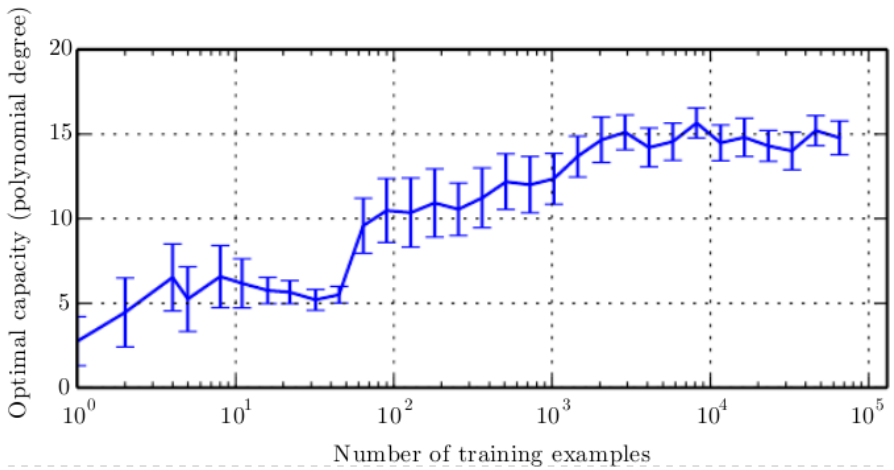
[goodfellow16]

Pregled strojnog učenja: utjecaj podataka

Empirijska i generalizacijska pogreška ovise o skupu za učenje

- uvedimo neizbježnu (ili Bayesovu) pogrešku (savršeni model):
  - uzroci: pogrešno označavanje, stohastički problem, ili šum
- kako skup za učenje raste, potkapacitirani algoritam konvergira grešci koja je veća od neizbježne (zeleni graf)
- ovisno o broju podataka za učenje, algoritam s prilagođenim kapacitetom (5) može biti lošiji od pot(pre-)kapacitiranog algoritma





[goodfellow16]

Pregled strojnog učenja: lex parsimonae

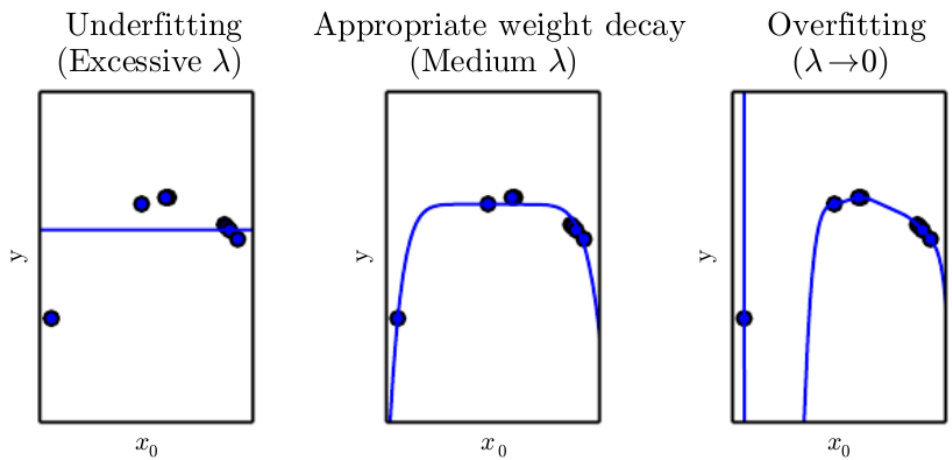
Ograničavanje kapaciteta modela nije jedini način usklađivanja algoritma s pristranošću podataka.

Drugi način je zadržati visoki kapacitet ali uvesti preferenciju prema **jednostavnijim** modelima modificiranjem **gubitka**.

Ako promatramo regresiju, jedan način regulariziranja gubitka bio bi:

$$\mathcal{L}(\mathbf{w}) = \lambda \mathbf{w}^\top \mathbf{w} + \sum_i (\sum_j w_j x_i{}^j + b - y_i)^2.$$

Algoritam sada apriorno preferira rješenja u kojima promjene ulaza vode na blage promjene odluke modela.



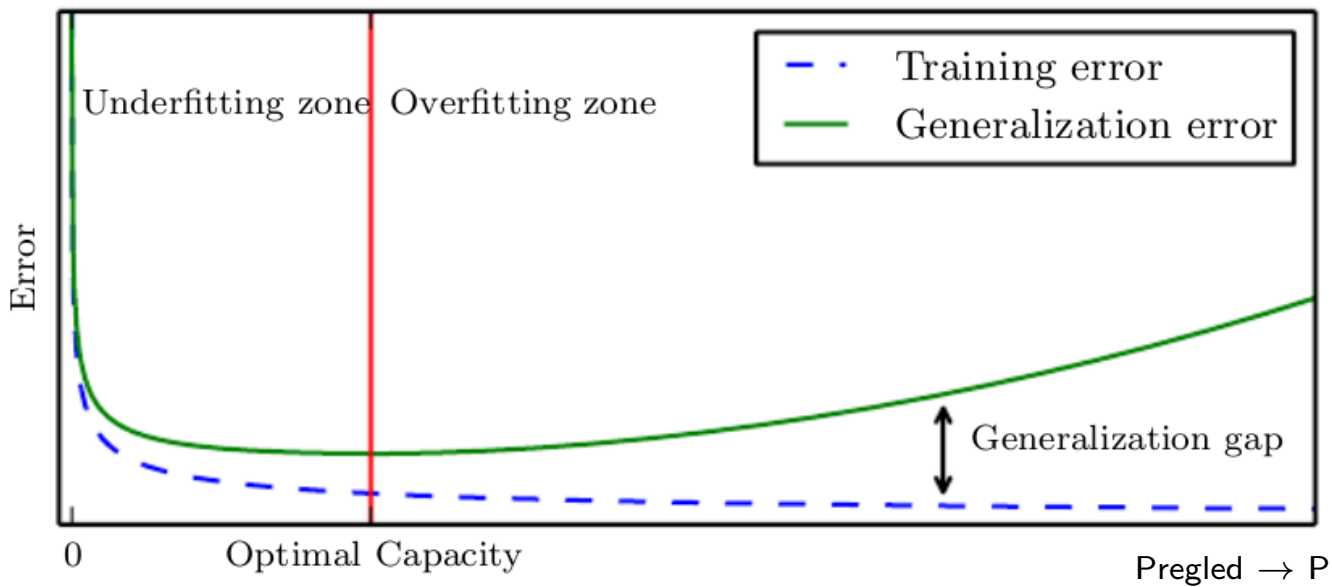
[goodfellow16]

Pregled strojnog učenja: regularizacija

Svaku modifikaciju koja namjerava **poboljšati generalizaciju** bez smanjivanja pogreške učenja nazivamo regularizacijom algoritma.

Regularizacija se može primijeniti na sve dijelove algoritma:

- gubitak: kažnjavanje norme vektora parametara
- optimizacijsku metodu: rano zaustavljanje učenja
- podatke: rastresanje, zašumljivanje oznaka
- model: kapacitet, vezivanje parametara.



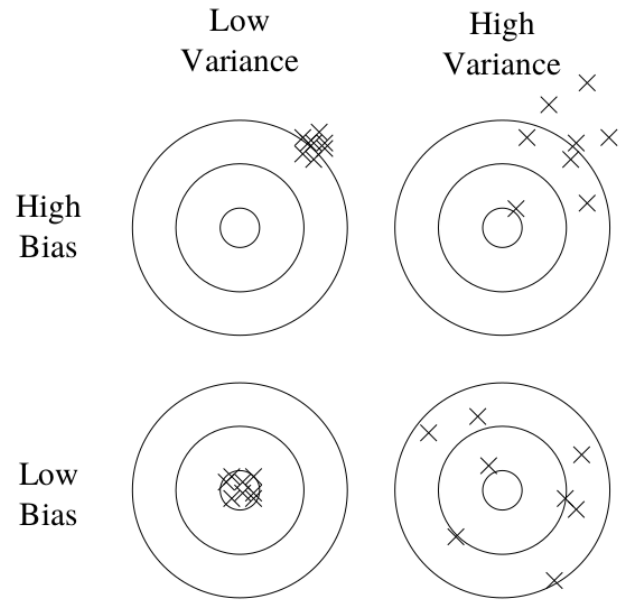
Pregled strojnog učenja: statistički pogled

Podnaučenost i prenaučenost možemo pojasniti ako generalizacijsku pogrešku rastavimo na doprinose pristranosti i varijance.

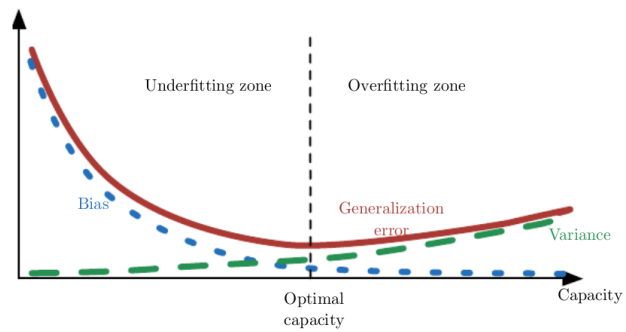
Pristranost algoritma: ugrađena sklonost prema nekim rješenjima.

Varijanca algoritma: sklonost da rezultat jako varira o ulaznim podatcima.

Regularizacija algoritma smanjuje varijancu i povećava pristranost.



[domingos16cacm]



[goodfellow16]

Za najbolje rezultate mjeru regularizacije treba prilagoditi podatcima:

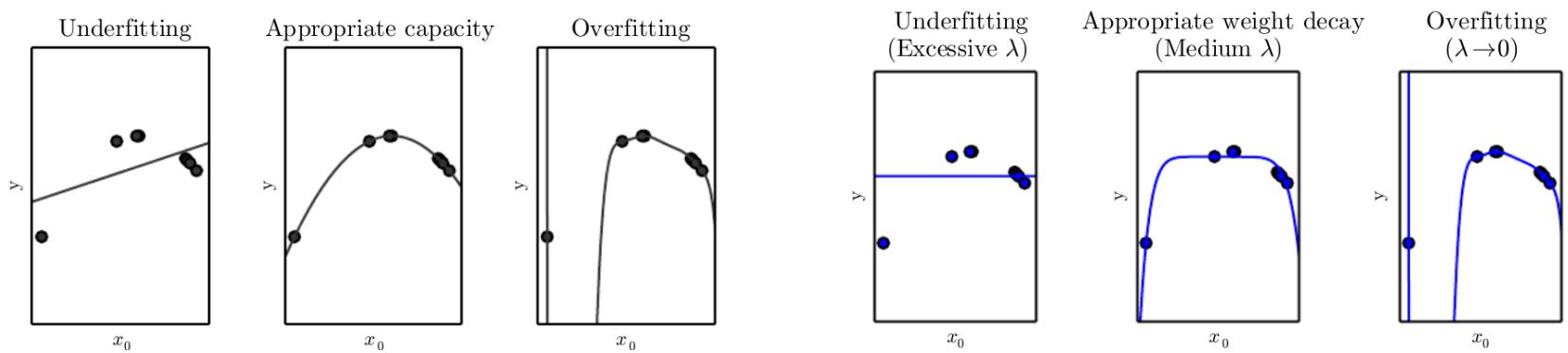
- izbjeći i lošu pristranost i pretjeranu varijancu.

Pregled strojnog učenja: hiperparametri

Većina algoritama ima tzv. hiperparametre koji reguliraju njihovo ponašanje, a nisu obuhvaćeni optimizacijskim postupkom

Primjeri: kapacitet modela, faktor kazne parametarske norme, korak optimizacije, broj epoha učenja...

Odabir takvih parametara najčešće provodimo iscrpnim ili slučajnim pretraživanjem prostora parametara na validacijskom skupu



[goodfellow16]

Pregled strojnog učenja: gubitak

Najintuitivniji gubitak je srednja kvadratna pogreška:

$$\mathcal{L}(\theta) = \sum_i (\text{model}(\mathbf{x}_i) - y_i)^2$$

Taj gubitak nije prikladan za probabilističku klasifikaciju:

- ignorira se struktura vjerojatnosne distribucije
- npr. L([1,0,0],[0,0,1]) vs L([0.5,0.5,0],[0,0,1])

Dosljednija formulacija gubitka u probabilističkom okruženju jest negativna log-izglednost parametara modela:

$$\mathcal{L}(\theta) = -\sum_i \log \mathsf{P}_{\text{model}}(y_i|\mathbf{x}_i, \theta).$$

Može se pokazati da negativna log-izglednost odgovara unakrsnoj entropiji (ekvivalent KL udaljenosti).

Ako modeliramo regresijsko odstupanje Gaussovom razdiobom, negativna log-izglednost se svodi na **kvadratni gubitak**.

Pregled strojnog učenja: SGD

Jedna od trenutno najpopularnijih optimizacijskih metoda u strojnom učenju jest stohastički gradijentni spust

Gradijentni spust po negativnoj log-izglednosti mora izračunati:

$$\nabla_\theta \mathcal{L}(\theta) = \frac{1}{N} \sum_i^N \nabla_\theta L(\mathbf{x}_i, y_i, \theta).$$

Kod teških problema ($N \sim 10^6$) ima smisla ublažiti petlju po N

Problem rješavamo odvajanjem podataka u manje grupe (engl. batch):

$$\nabla_\theta \mathcal{L}(\theta) = \frac{1}{N'} \sum_i^{N'} \nabla_\theta L(\mathbf{x}_i, y_i, \theta).$$

- korak optimizacije izvodi se u vremenu O(N'·dim(θ)), N'<<N
- grupe se slučajno formiraju nakon svake epohe (stohastika!)
- u velikim modelima (dim$(\theta) \sim 10^9$): brže od metoda višeg reda
- SVM s jezgrom: prostor $\sim O(N^2)$ (Gramova matrica), vrijeme $\sim O(N^3)$.
- koristi se i kod velikih linearnih modela!

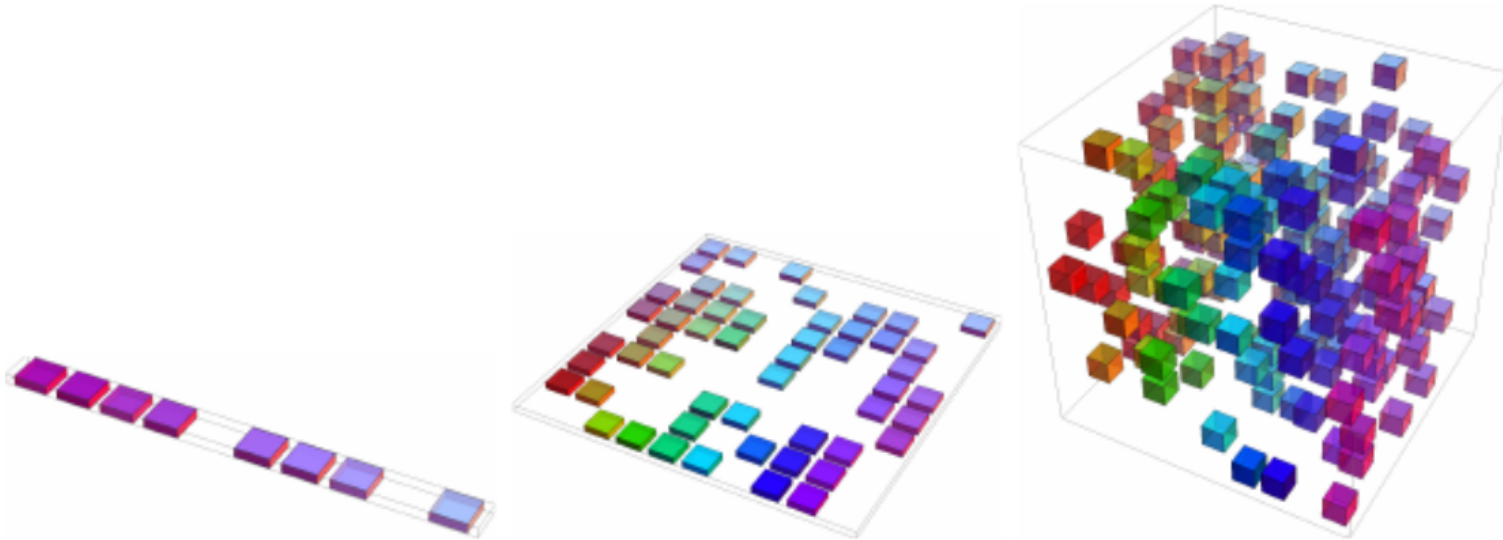
Prema dubokom učenju: izazov

Problemi UI-razine povezani su s visoko-dimenzionalnim podatcima (npr. d=10000)

Izazov: broj "zanimljivih" konfiguracija raste barem u $O(2^d)$

U pesimističnom slučaju (pristranost klasifikatora ne pogađa pristranost podataka) treba nam $O(2^d)$ primjera za učenje!

Taj izazov nazivamo prokletstvom porasta dimenzionalnosti (engl. curse of dimensionality).



[goodfellow16]

Prema dubokom učenju: klasični odgovor

Klasični klasifikatori pristupaju tom problemu pod pretpostavkom glatkoće (ili lokalne konstantnosti) modela:

> *rezultat modela ne bi se smio "mnogo" mijenjati unutar kompaktne regije prostora podataka*

Ovakvu pristranost implementiraju k-NN, jezgrene metode, stabla

- posljedično, svi ti pristupi zahtijevaju O(k) primjera za učenje za razlikovanje O(k) regija u prostoru podataka

Glatkoća se slabo ponaša u visoko dimenzionalnim prostorima:

- pogotovo u računalnom vidu gdje posmaci i deformacije dijelova ne mijenjaju identitet objekta
- trebaju nam modeli čija interna građa odgovara **građi stvarnih podataka**.

Prema dubokom učenju: kompozicijski modeli

Temeljna pretpostavka dubokih modela: podatci su generirani kompozicijom faktora kroz više razina hijerarhije

Takva pretpostavka pruža mogućnost da se model izrazi manjim brojem parametara

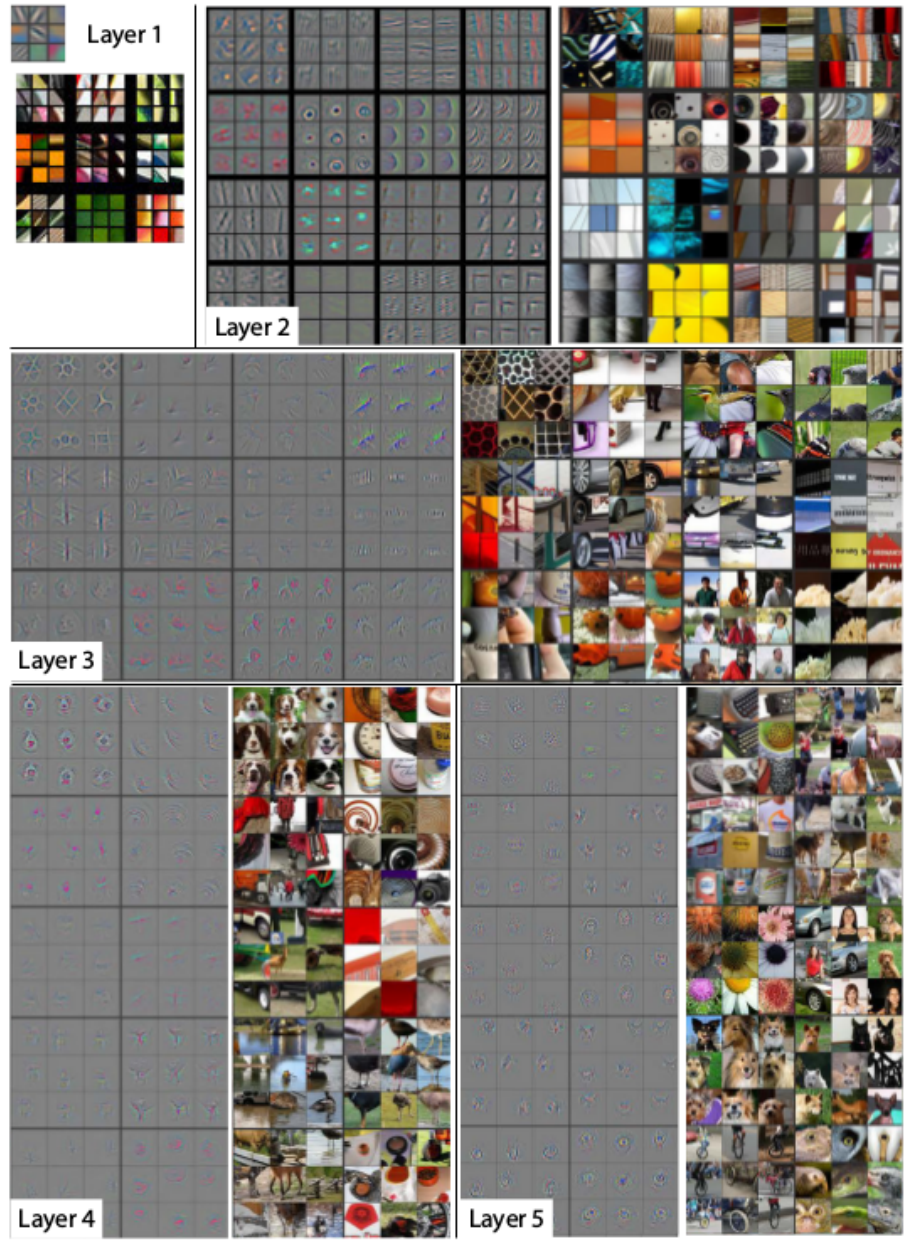
Nadalje, postoji mogućnost nezavisnog učenja parametara značajki niže razine: jedan primjer za učenje može doprinositi nezavisnom učenju modela u više regija prostora parametara.

Ove dvije mogućnosti pružaju potencijal suprotstavljanja eksponencijskom izazovu porasta dimenzionalnosti.

Prema dubokom učenju: kompozicijski modeli (2)

Kompozitna struktura: važan oblik pristranosti dubokih modela

- podatkovna reprezentacija na razini n gradi se od reprezentacija na razini n-1
- to svojstvo dubokih modela dobro odgovara podatcima u mnogim teškim zadatcima
  - npr. auto ima kotače, kotači imaju naplatke, naplatci imaju rupice
  - slova čine riječi, riječi sintagme, a sintagme rečenice.



[zeiler14eccv]

Prema dubokom učenju: primjer

Pokušajmo prednosti kompozicijske strukture ilustrirati na primjeru modela za **binarnu klasifikaciju lica**:

- pretpostavimo da nam treba n_O podataka za naučiti detekciju oka...
- ...i n_N parametara za naučiti detekciju nosa.

Algoritam učenja plitkog modela treba vidjeti svaku kombinaciju oka i nosa da bi naučio prepoznati lice:

- potreban broj podataka za učenje: $O(n_O \cdot n_N)$.

Duboki klasifikator ima *šansu* odvojeno naučiti parametre oka i nosa (razina k) te nakon toga nad istim podatcima naučiti da lice ima dva oka i nos (razina k+1)

- potreban broj podataka za učenje: $O(n_O) + O(n_N)$

Zaključak: kod artikuliranih podataka kompozicijski model ima bolji generalizacijski potencijal od plitkog modela.

Prema dubokom učenju: učenje mnogostrukosti

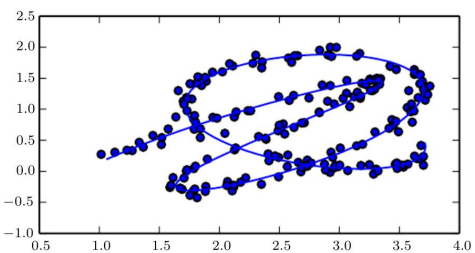
Mnogostrukost: povezani skup $\{\mathbf{x}_i\} \in \mathbb{R}^n$ koji se lokalno može aproksimirati nižedimenzionalnim skupom $\{\mathbf{x}'_i\} \in \mathbb{R}^m$, $m << n$.

Još jedan način pristupanja porastu dimenzionalnosti: pretpostaviti da podatci stanuju na mnogostrukosti

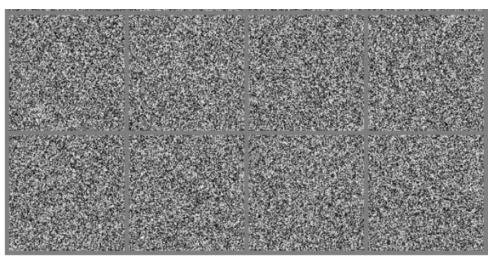
- model se u tom slučaju može specijalizirati na dijelove podatkovnog prostora s velikom gustoćom vjerojatnosti podataka

Koliko je ta pretpostavka zadovoljena u praksi?

- većina svih mogućih ulaznih vektora ne predstavlja valjane podatke
- možemo zamisliti skup transformacija koje definiraju obrise mnogostrukosti: svjetlina, kontrast, rotacija objekta itd.



[goodfellow16]



Prema dubokom učenju: DU vs neuronske mreže

Sličnost: potpuno povezani duboki regresijski model je ekvivalentan **višerazinskom perceptronu**

Razlike:

- duboki model nema ambiciju modelirati mozak
- duboki modeli koriste napredne regularizacijske tehnike:
  - dijeljenje parametara, izostavljanje čvorova, normalizacija po grupi
- specifičnosti dubokog učenja
  - nove aktivacije: ReLU, maxout, softmax, SELU
  - novi slojevi: sažimanje, pažnja, interpoliranje ...
  - preskočne, rezidualne i multipleksirane veze
  - arhitekure sa samo jednim potpuno povezanim slojem
  - metrike inducirane korespondencijskim ugrađivanjem
- neki formalizmi neuronskih mreža nisu relevantni za duboko učenje

ZAHVALA

Ova predavanja proizišla su iz istraživanja koje je financirala Hrvatska zaklada za znanost projektom I-2433-2014 MultiCLoD.

`http://multiclod.zemris.fer.hr`