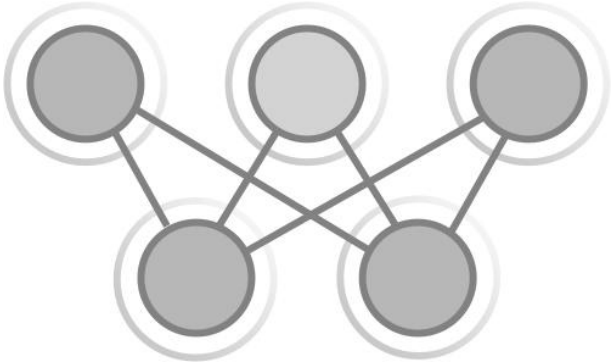
Prof.dr.sc. Bojana Dalbelo Bašić

Fakultet elektrotehnike i računarstva
Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave

www.zemris.fer.hr/~bojana
bojana.dalbelo@fer.hr

Učenje na temelju primjera



Učenje na temelju primjera

- Dosadašnje metode nastoje konstruirati eksplicitan opis ciljne funkcije.
- **Metode učenja na temelju primjera** pohranjuju primjere za učenje.
- Postupak generalizacije odgođen je do trenutka potrebe za klasifikacijom novog uzorka.
  - Metoda *k*-najbližih susjeda
    (engl. k-nearest-neighbor)
  - Metoda lokalne regresije s težinskim faktorima
    (engl. locally weighted regression)
  - Zaključivanje na temelju slučajeva
    (engl. case based reasoning)
  - Radijalne bazne funkcije
    (engl. radial basis functions)

Lazy vs. eager

- **Lazy** (lijene) metode
  - Odgađaju odluku o klasifikaciji sve do trenutka predočavanja novog primjera (upita).
  - Metoda *k* najbližih susjeda, metoda lokalne regresije s težinskim faktorima i zaključivanje na temelju slučajeva.
- **Eager** (marljive, nestrpljive) metode
  - Sve do sada iznesene metode (npr. ID3).
  - Od gornje navedenih: radijalne bazne funkcije.

Prednosti lijenih metoda

- Dvije važne razlike (prednosti) *lijenih* metoda, tj. metoda s odgodom, naspram ostalih metoda:
  - Konstruiraju različitu aproksimaciju ciljne funkcije za svaki različiti novi upit (primjer koji treba biti klasificiran)
  - Umjesto procjene ciljne funkcije, jednom za cijeli prostor, te metode procjenjuju ciljnu funkciju samo **lokalno**, u okolini novog primjera. Takva lokalna procjena ciljne funkcije je pogodna za vrlo kompleksne ciljne funkcije
- Nedostatak metoda učenja na temelju primjera:
  - visoka cijena klasificiranja novog primjera
  - razmatraju se sve značajke nekog primjera prilikom klasifikacije iako samo neke mogu imati utjecaj na ciljnu funkciju (*k*-najbližih susjeda)
  - **složenost modela (a time i broj parametara) raste s brojem primjera** (neparametarske metode!)

Metoda *k* najbližih susjeda

- *Engl. k-nearest-neighbors*, skraćeno *k*-nn
- Ideja je da se novi primjer klasificira tako da se pogledaju njemu najbliži primjeri iz skupa za učenje.
- Primjeri su najčešće točke u *n*-dimenzijskom prostoru R^n, a za račun udaljenosti koristi se euklidska metrika.
  - Moguće je da primjeri budu npr. nizovi znakova, a za udaljenost da se koristi Levenshteinova udaljenost.
- Dva moguća zadatka:
  - Klasifikacija – vrijednosti ciljne funkcije su iz konačnog skupa.
  - Regresija – ciljna funkcija poprima realne vrijednosti.

Metoda *k* najbližih susjeda

- Klasifikacija točaka iz prostora R^n korištenjem euklidske metrike.
- Primjer **x** opisan je vektorom značajki

$$(x_1, x_2, \ldots, x_n)$$

- Euklidska udaljenost dvaju vektora x^a i x^b je

$$d(x^a, x^b) = \sqrt{\sum_{i=1}^{n} (x_i^a - x_i^b)^2}$$

- Za ciljnu funkciju s diskretnim vrijednostima:

$$f: R^n \rightarrow C, \text{ gdje je } C = \{C_1, C_2, \ldots, C_K\}$$

Algoritam *k* najbližih susjeda (k-nn)

- Algoritam za učenje
  - Za svaki primjer za učenje $(x^{(i)}, y^{(i)})$ dodaj primjer na listu primjeri_za_učenje.

- Algoritam klasifikacije
  - Za dani primjer x_q s nepoznatom klasifikacijom
  - Neka x_1, x_2, …,x_k označavaju k primjera koji su najbliži x_q.
- Vrati

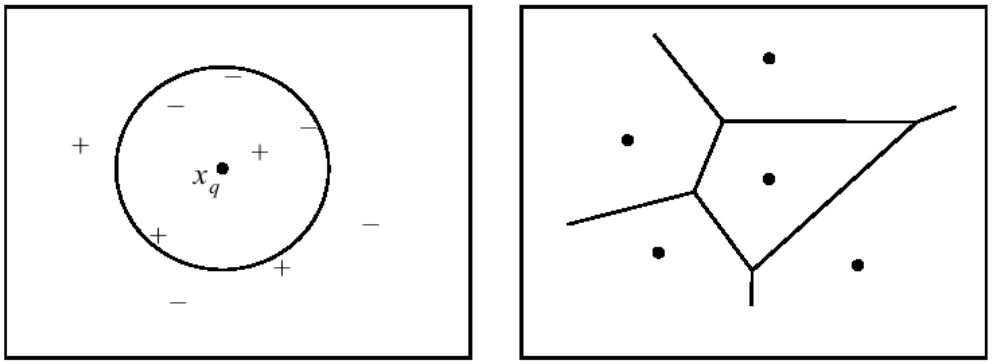
$$h(x_q) = \arg\max_{v \in \{0,1,..,K\}} \sum_{i=1}^{k} \delta(v, y^{(i)})$$

gdje je δ(a,b)=1 ako a=b, 0 inače.

- $h(x_q)$ je najčešća vrijednost ciljne funkcije koja se pojavljuje među *k* primjera za učenje koji su najbliži upitu x_q
- Napomena: treba birati neparan *k* kako bi $h(x_q)$ bila deterministička funkcija

Algoritam *k* najbližih susjeda (k-nn)

- Razlika kod 1-nn i 5-nn algoritma:



- *k*-nn algoritam nikad ne oblikuje eksplicitnu hipotezu za ciljnu funkciju
- Za 1-nn možemo je predočiti Voronoyjevim dijagramom. Decizijska površina je konveksni poliedar koji okružuje svaki primjer za učenje.

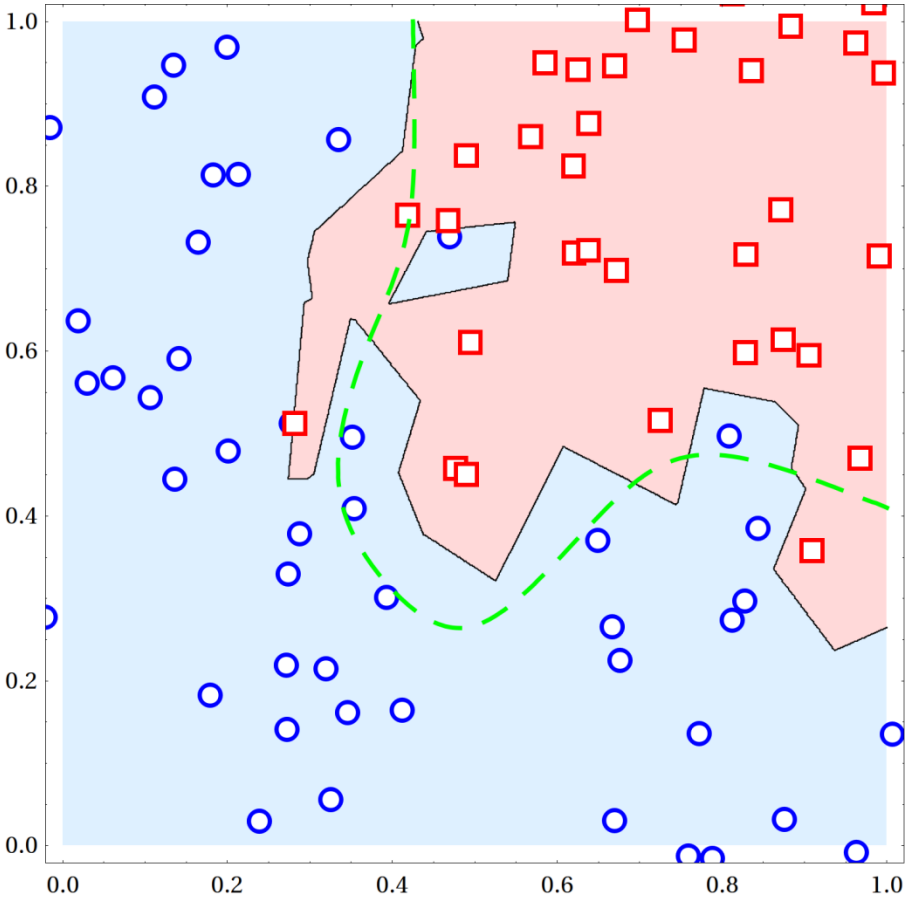
Primjer rada algoritma *k*-nn

- Na primjeru će se vidjeti da se rastom *k* smanjuje varijanca, ali se povećava pristranost.
- Uz poznatu pravu distribuciju svih primjera konstruiran je optimalan Bayesov klasifikator čija je decizijska granica prikazana zelenom iscrtkanom crtom.

- Idealno: Što veći broj primjera N; velik ali ne prevelik *k* (ne premalen, inače je prevelika varijanca, ali ne prevelik, inače je prevelika pristranost)
  - *k* definira složenost modela (*k* je hiperparametar)

Primjer rada algoritma *k*-nn

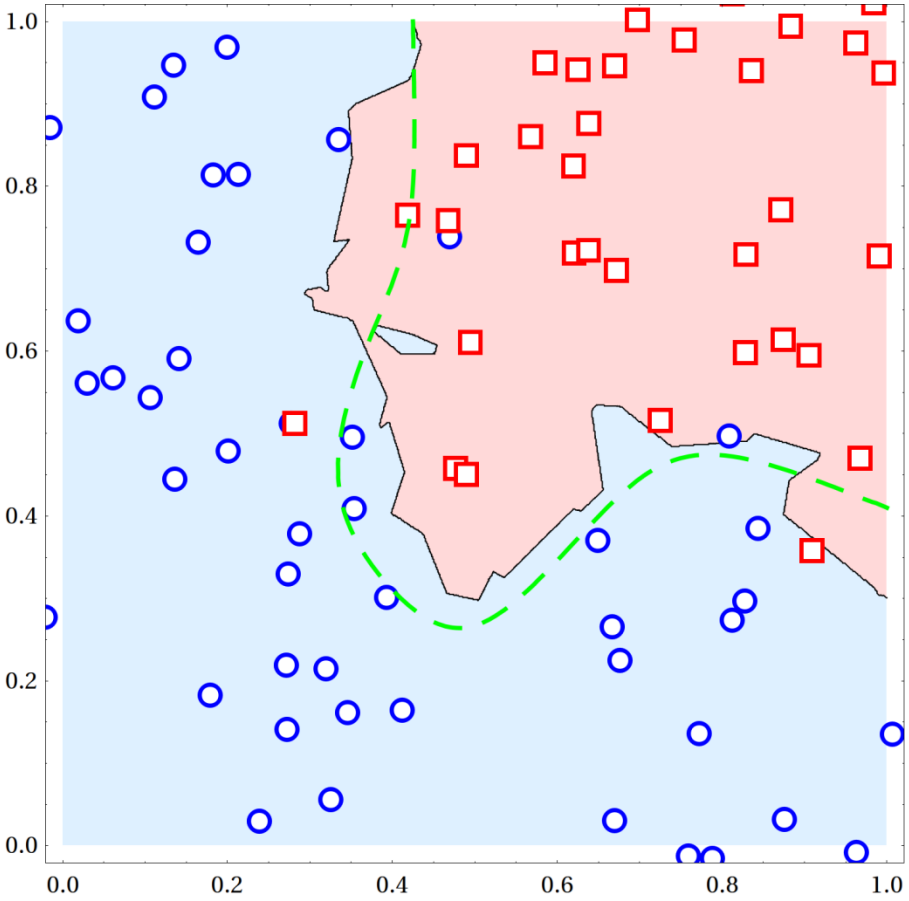
- k = 1, N=100



Visoka varijanca
(granica između klasa je nezaglađena i vrlo bi se razlikovala za neki drugi skup primjera D)

Primjer rada algoritma *k*-nn

- k = 3, N=100



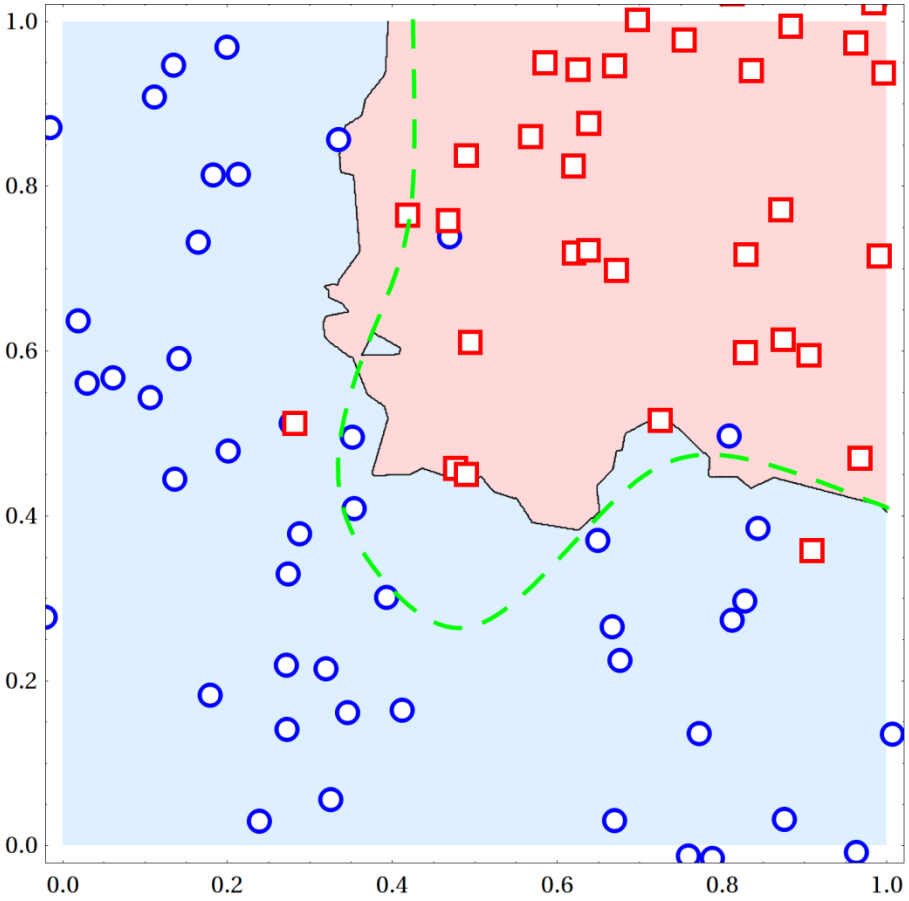
Ovo je dobar odabir parametra *k*

Primijetite da može biti E(h|D)>0, ako k>1

(idealno, primjeri s kojima hipoteza nije konzistentna su oni koji sačinjavaju šum)

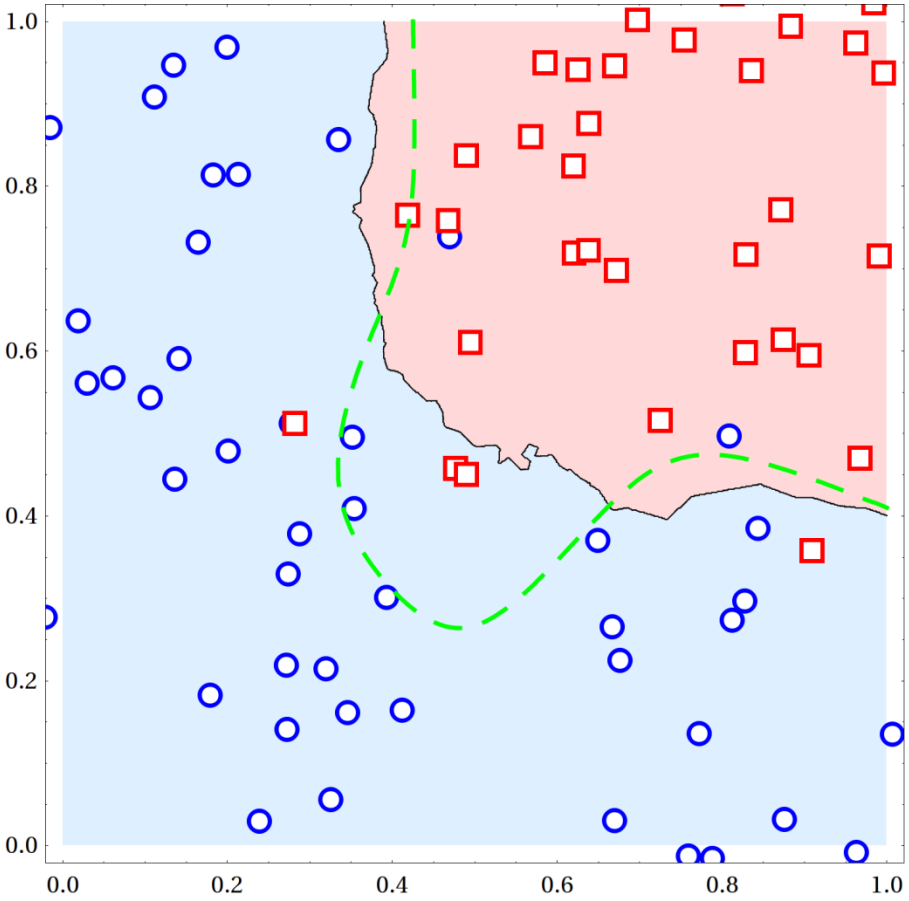
Primjer rada algoritma *k*-nn

- k = 5, N=100



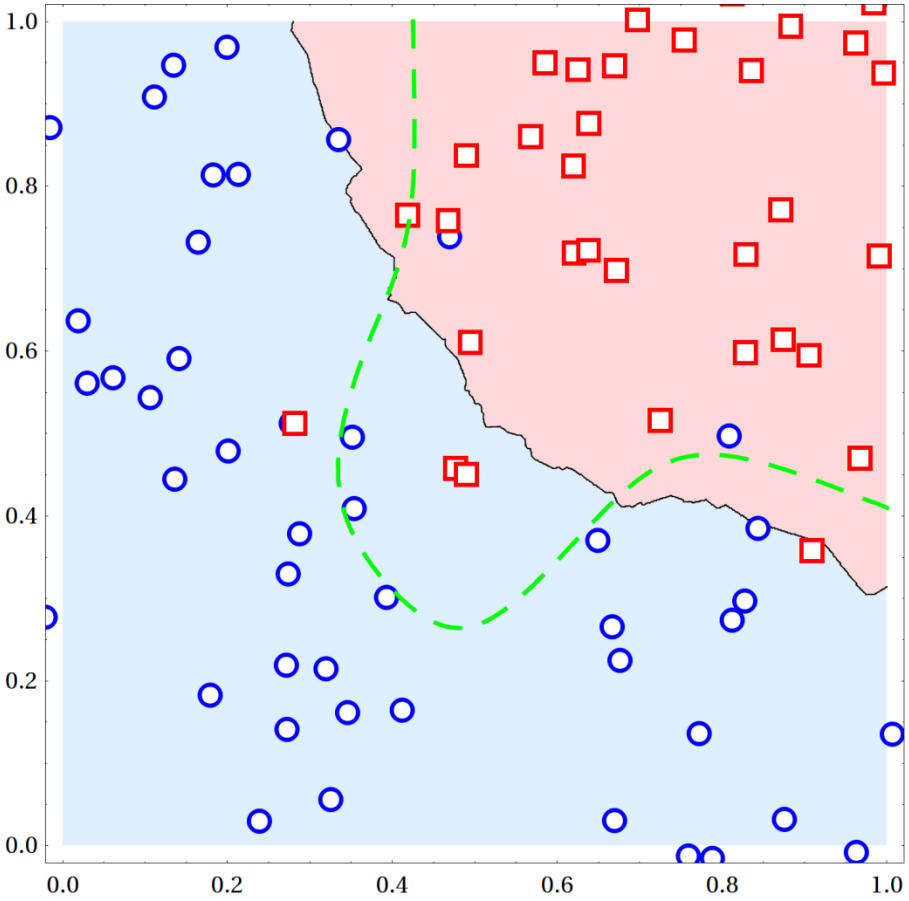
Primjer rada algoritma *k*-nn

- k = 15, N=100



Primjer rada algoritma *k*-nn

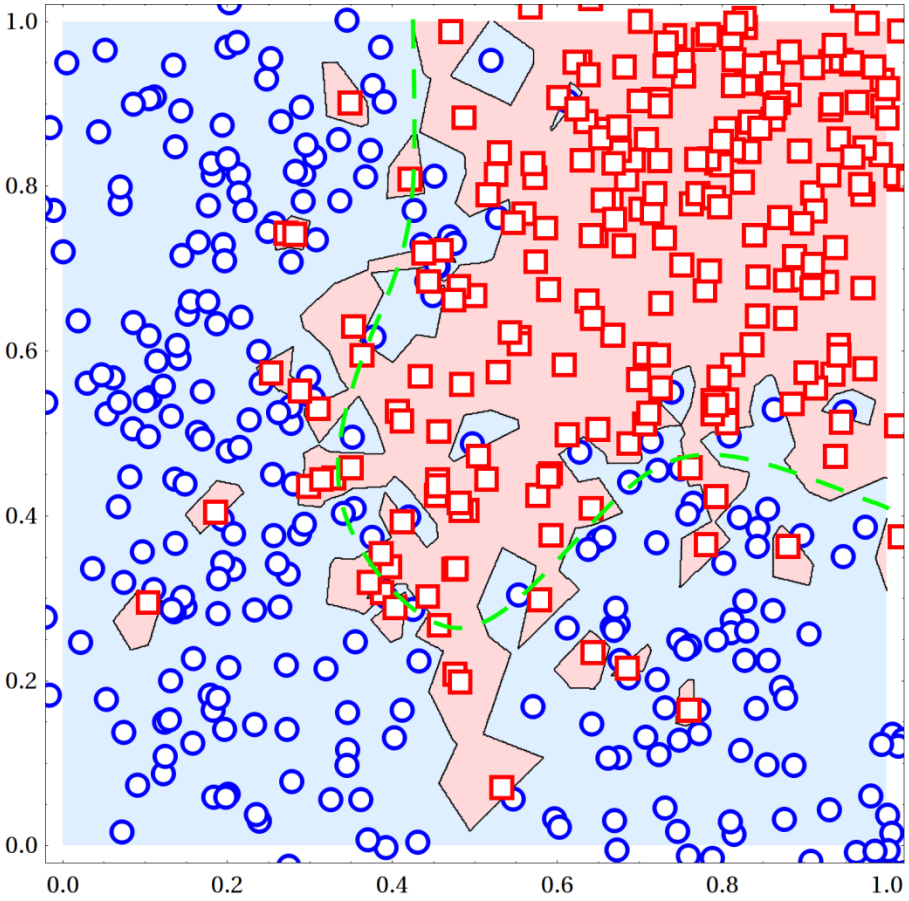
- k = 31, N=100



Visoka pristranost (granica je previše zaglađena)

Primjer rada algoritma *k*-nn

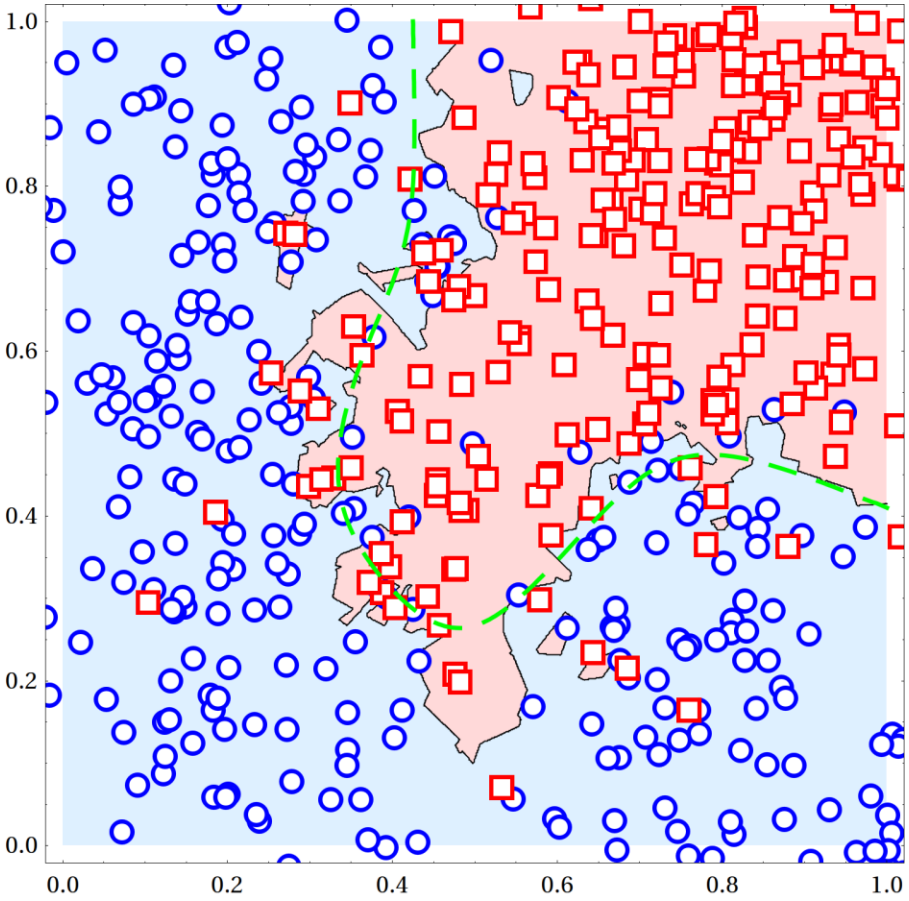
- k = 1, N=600



Puno primjera uvijek pomaže, ali k mora biti veći. Ovdje je previsoka varijanca.

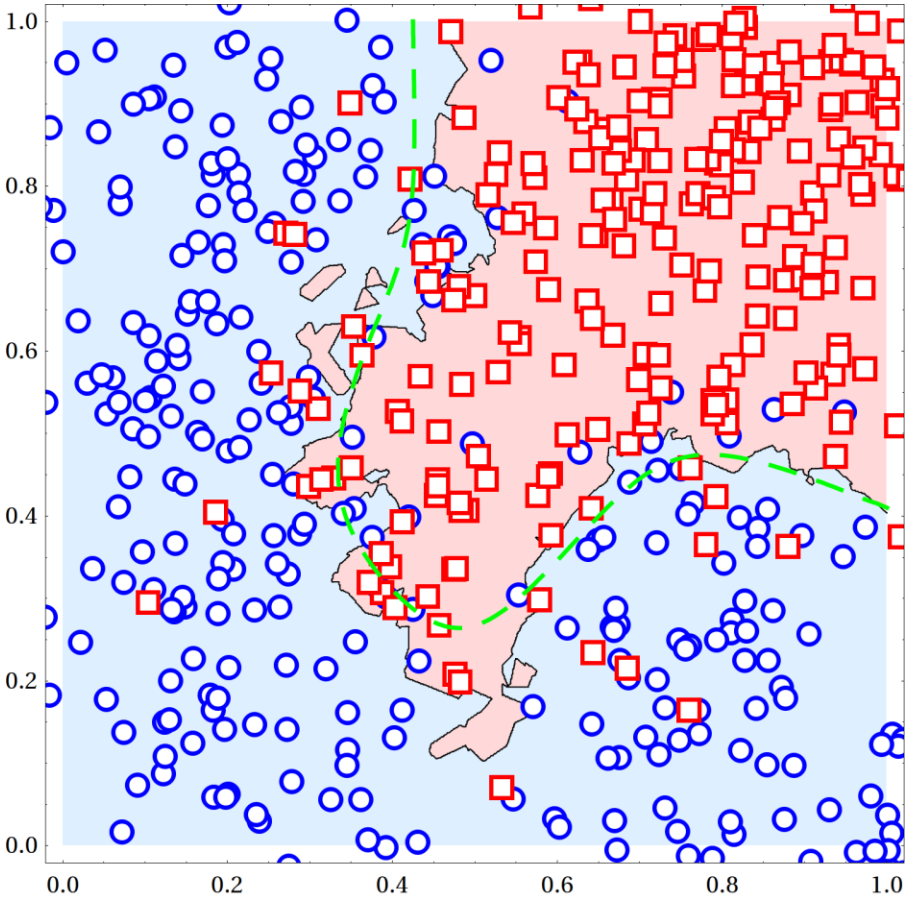
Primjer rada algoritma *k*-nn

- k = 3, N=600



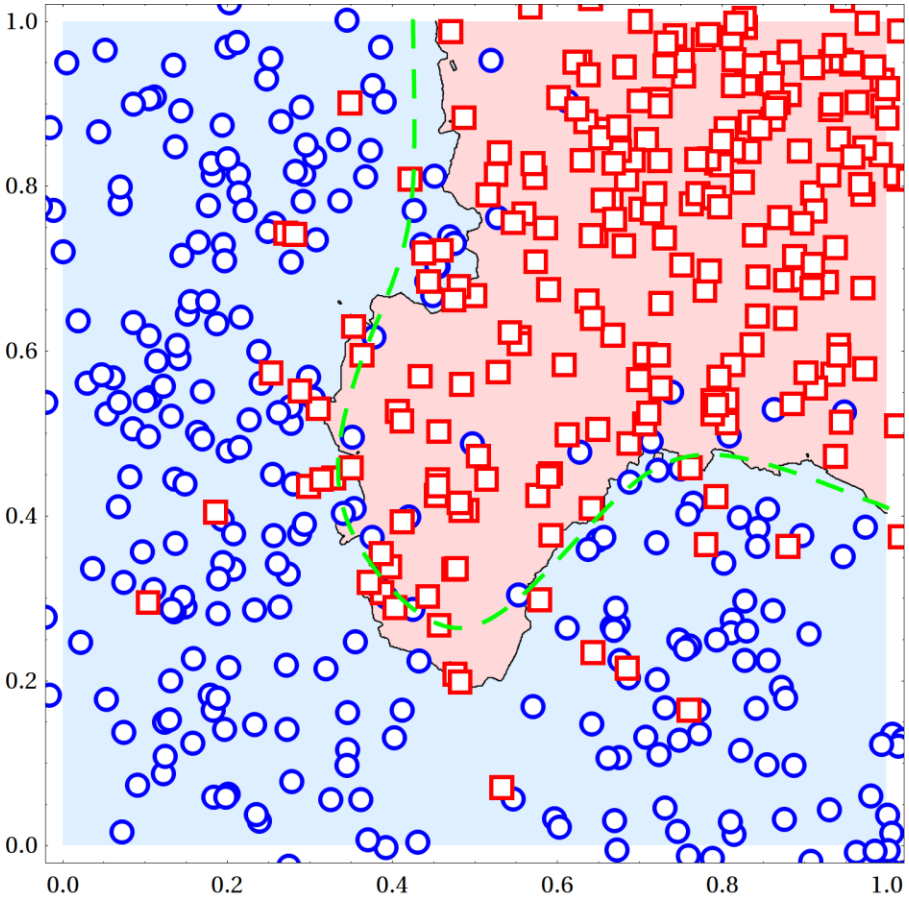
Primjer rada algoritma *k*-nn

- k = 5, N=600



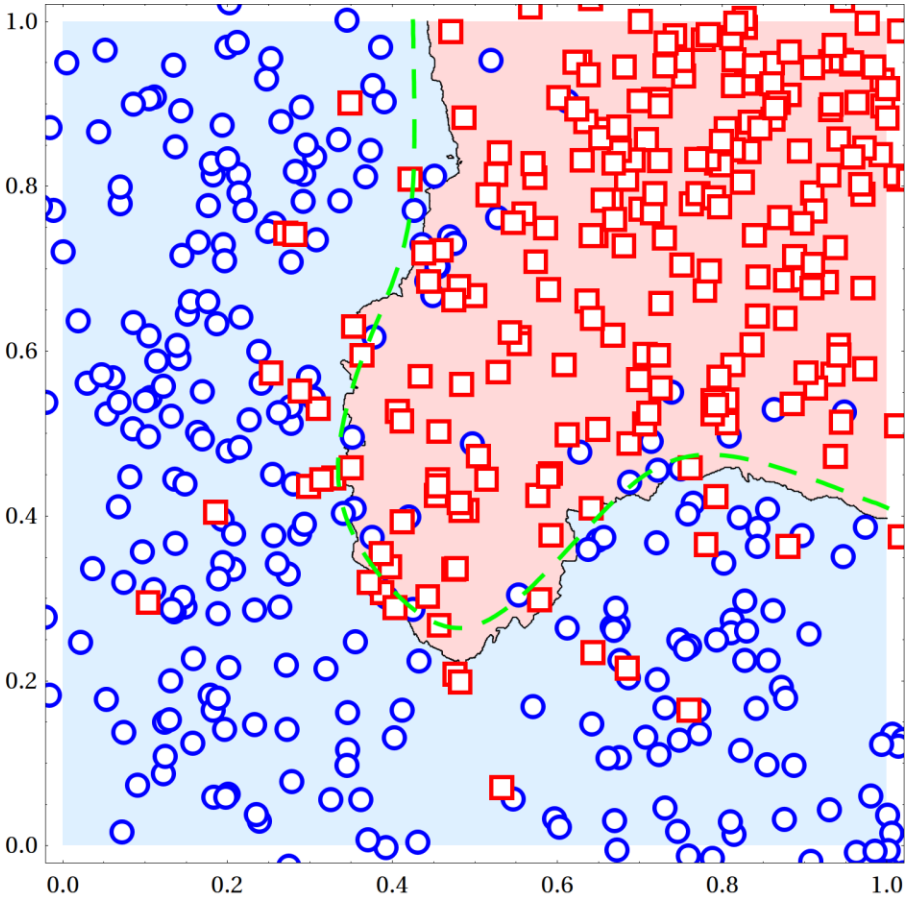
Primjer rada algoritma *k*-nn

- k = 15, N=600



Primjer rada algoritma *k*-nn

- k = 31, N=600



Ovo je dobar odabir parametra *k* (velik, ali ne prevelik; red veličine manji od N)

Parametar *k* određuje složenost modela, pa **optimalnu** vrijednost za *k* možemo odrediti **unakrsnom provjerom** (kao što to radimo i kod drugih algoritama)

Regresija pomoću algoritma *k*-nn

- Umjesto najčešće pojavljivane vrijednosti ciljne funkcije, odgovor na upit je srednja vrijednost ciljnih funkcija *k* najbližih susjeda.

$$h(x_q) = \frac{1}{k}\sum_{i=1}^{k} y^{(i)}$$

Modifikacija algoritma *k*-nn uvođenjem težinskih faktora udaljenosti

- Uvođenje težinskih faktora w_i za svaki od *k* susjeda, koji ovisi o njegovoj udaljenosti od upita x_q.

$$h(x_q) = \arg\max_{v \in \{0,1,\ldots,K\}} \sum_{i=1}^{k} w_i \delta(v, y^{(i)})$$

$$w_i = \frac{1}{d(x_i, x_q)^2}$$

- U slučaju $x_i = x_q$ pridružujemo funkciji vrijednost $y^{(i)}$

Modifikacija algoritma *k*-nn uvođenjem težinskih faktora udaljenosti

- U slučaju regresije (kontinuirane ciljne funkcije):

$$h(x_q) = \frac{\sum_{i=1}^{k} w_i y^{(i)}}{\sum_{i=1}^{k} w_i}$$

- Ovo je tzv. **Shepardova** metoda

Primjedbe na algoritam *k*-nn

- Prednosti:
  - efikasna induktivna metoda
  - robusna na šum u primjerima za učenje
  - (Cover i Hart) Ako broj primjera za učenje teži u beskonačno, onda je greška 1-nn klasifikatora najviše dva puta veća od greške optimalnog Bayesovog klasifikatora.

- *Induktivna pristranost:*
  - pretpostavka da je klasifikacija upita x_q slična klasifikaciji primjera u blizini.

- Udaljenost se računa na temelju svih značajki (za razliku od ID3 ili učenja skupova pravila koji odabiru podskupove značajki pri formiranju hipoteze).

Primjedbe na algoritam *k*-nn

- ***«Curse of dimensionality»*** – osjetljivost algoritma *k*-nn na sve značajke bez obzira na dimenziju prostora (broj značajki) i njihov značaj za ciljnu funkciju.
  - Moguće rješenje: rastezanje ili stiskanje osi euklidskog prostora (množenje vrijednosti značajki faktorima) da bi se smanjio utjecaj nevažnih značajki.
  - Općenitije rješenje: smanjenje dimenzionalnosti ili odabir podskupa značajki

- Praktična tema vezana za *k*-nn je efikasno **indeksiranje prostora primjera** zbog brzog dohvata primjera kod novog upita.

Nazivlje

- **Metode s odgodom** - područje statističkog raspoznavanja uzoraka
- **Regresija** – način aproksimacije ciljne funkcije s realnim vrijednostima
- **Rezidual** (ostatak) – pogreška $h(x^{(i)}) - y^{(i)}$
- **Jezgrena funkcija** (*engl. kernel function*) – funkcija udaljenosti koja se koristi za određivanje težinskih faktora primjera za učenje, tj. jezgrena funkcija ***K*** je takva da je

$$w_i = K(d(x^{(i)}, x_q))$$

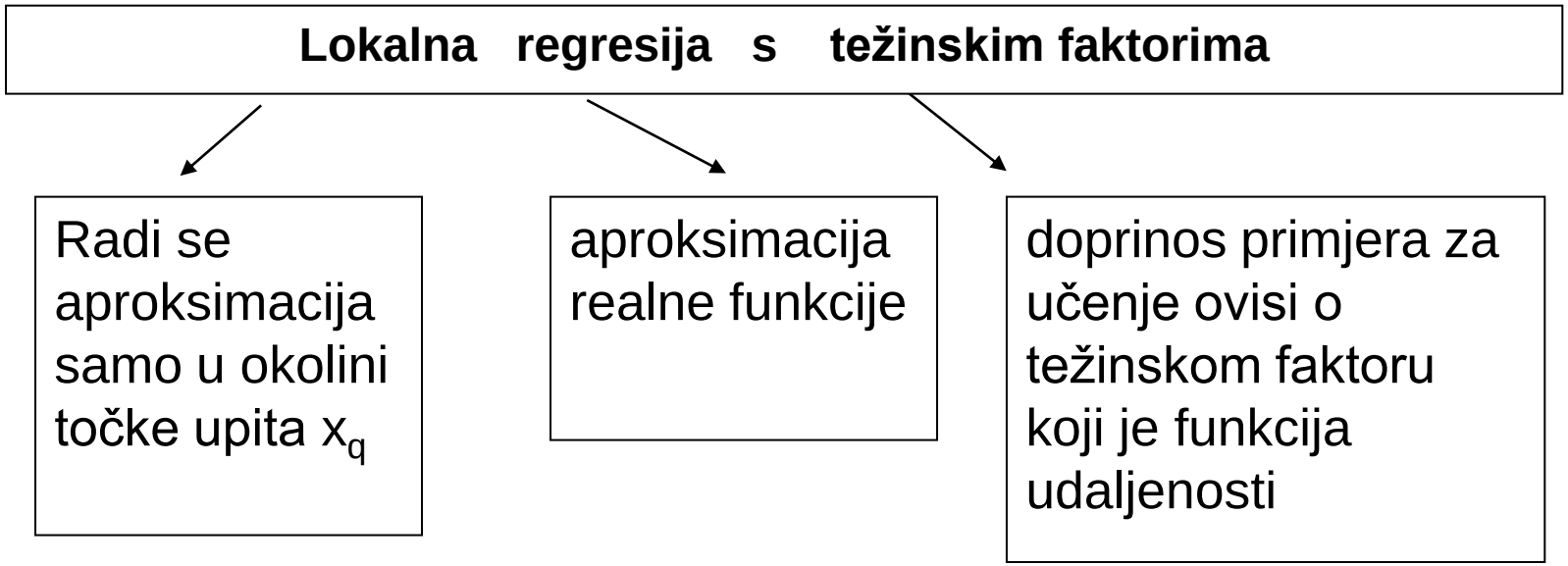
Lokalna regresija s težinskim faktorima

- *Engl. locally weighted regression*
- Algoritam *k*-nn se može interpretirati kao aproksimiranje ciljne funkcije u h(x) u točci $x=x_q$.

- Regresija s težinskim faktorima generalizacija je te metode jer **konstruira eksplicitnu aproksimaciju ciljne funkcije na cijelom lokalnom području** oko x_q.

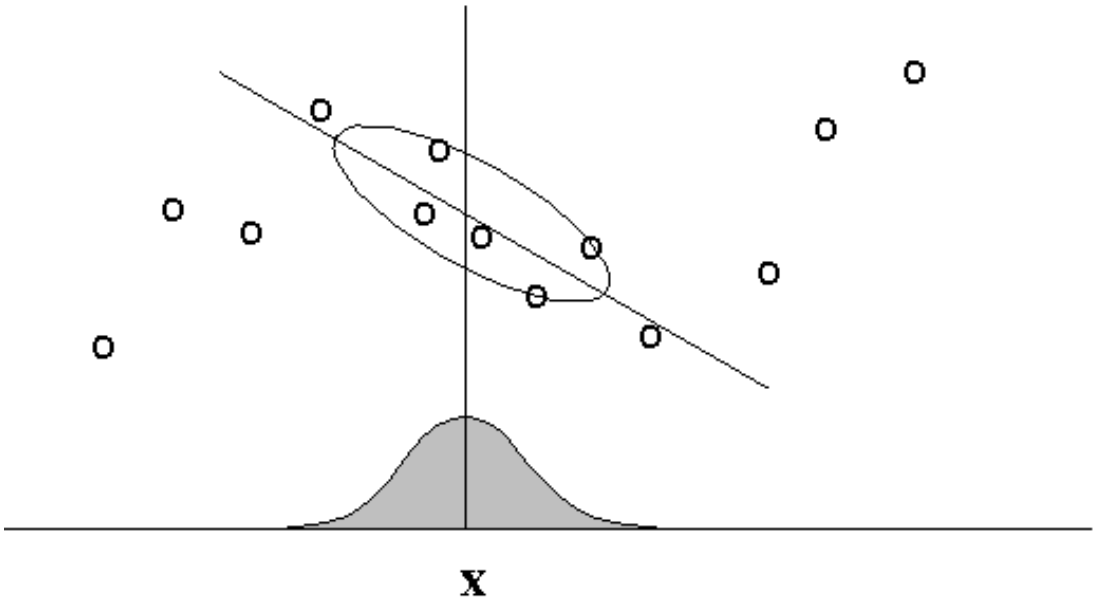
- Aproksimacija ciljne funkcije, može biti ostvarena:
  - linearnom funkcijom
  - kvadratnom funkcijom
  - višeslojnom neuronskom mrežom

Lokalna regresija s težinskim faktorima



- Neka je dan je upit x_q
  - konstruira se aproksimacija h(x) ciljne funkcije koja odgovara primjerima za učenje u okolini x_q
  - aproksimacija se koristi za izračun vrijednosti $h(x_q)$.

Lokalna regresija s težinskim faktorima



- *h* aproksimiramo linearnom funkcijom

$$h(x) = w_0 + w_1 x_1 + \ldots + w_n x_n$$

x_i je vrijednost *i*-te značajke primjera x.

Lokalna regresija s težinskim faktorima

- Metoda globalne aproksimacije:

$$E(h \mid D) = \frac{1}{2} \sum_{i=1}^{N} (h(x^{(i)}) - y^{(i)})^2$$

- Tri moguća kriterija prilagodbe ove metode za lokalnu aproksimaciju:
  1. Minimizacija kvadrata pogreške samo nad *k* najbližih susjeda.

$$E_1(h \mid D) = \frac{1}{2} \sum_{\substack{x^{(i)} \in k \text{ najbližih} \\ \text{susjeda od } x_q}} (h(x^{(i)}) - y^{(i)})^2$$

Lokalna regresija s težinskim faktorima

2. Minimizacija kvadrata pogreške nad skupom primjera za učenje *D* uz umnožak s težinskim faktorima

$$E_2(h \mid D) = \frac{1}{2}\sum_{i=1}^{N}(h(x^{(i)}) - y^{(i)})^2 K(d(x_q, x^{(i)}))$$

3. Kombinacija 1. i 2.

$$E_1(h \mid D) = \frac{1}{2}\sum_{\substack{x^{(i)} \in k \text{ najbližih} \\ \text{susjeda od } x_q}}^{N}(h(x^{(i)}) - y^{(i)})^2 K(d(x_q, x^{(i)}))$$

Lokalna regresija s težinskim faktorima

- Model pod 2 je računski najzahtjevniji.
- Ako usvojimo 3. model, pravilo učenja je:

$$\Delta w_j = \eta \sum_{\substack{x^{(i)} \in k \text{ najbližih} \\ \text{susjeda od } x_q}}^{N} K(d(x_q, x^{(i)}))(h(x^{(i)}) - y^{(i)})\, x^{(i)}$$

- Napomena:
  - Postoji niz varijanti metode linearne regresije s težinskim faktorima. Funkcija f(x) je u našem slučaju linearna, no koriste se još i kvadratna aproksimacijska funkcija, ali ne i složenije zbog cijene koja bi se platila za izračunavanje takve funkcije za svaki pojedini upit.

Radijalne bazne funkcije **(RBF)**

- Metoda aproksimacije funkcije (povezana sa *k*-nn i lokalnom regresijom).
- Hipoteza je funkcija oblika:

$$h(x) = w_0 + \sum_{u=0}^{k} w_u K_u(d(x_u, x)) \qquad (1)$$

gdje su:

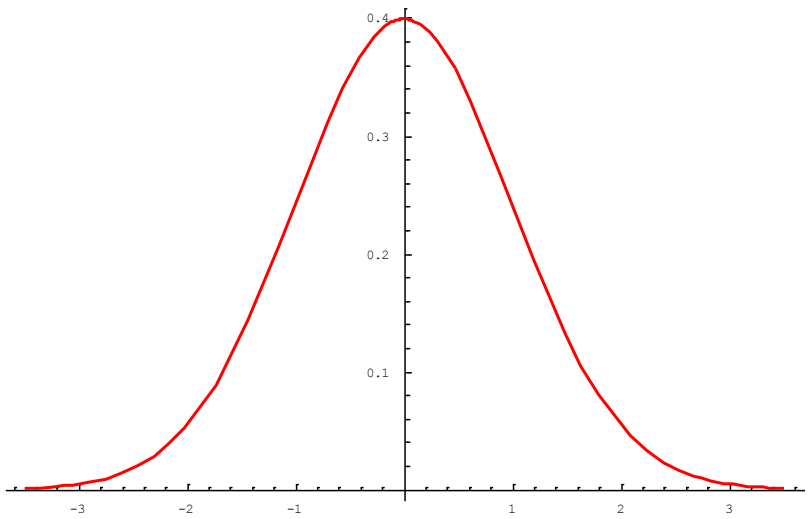
- x_u su primjeri iz *X*
- $K(d(x_u,x))$ je jezgrena funkcija koja se smanjuje kada udaljenost raste
- *k* je proizvoljan broj jezgrenih funkcija

- Iako je h(x) globalna aproksimacija f(x), doprinos svake $K_u(d(x_u,x))$ je lokalan – samo u okolini x_u

Radijalne bazne funkcije

- Uobičajen izbor za $K_u(d(x_u,x))$ su Gaussove* funkcije s centrom u x_u i varijancom σ^2.

$$K_u\left(d(x_u,x)\right) = e^{-\frac{1}{2\sigma_u{}^2}d^2(x_u,x)}$$

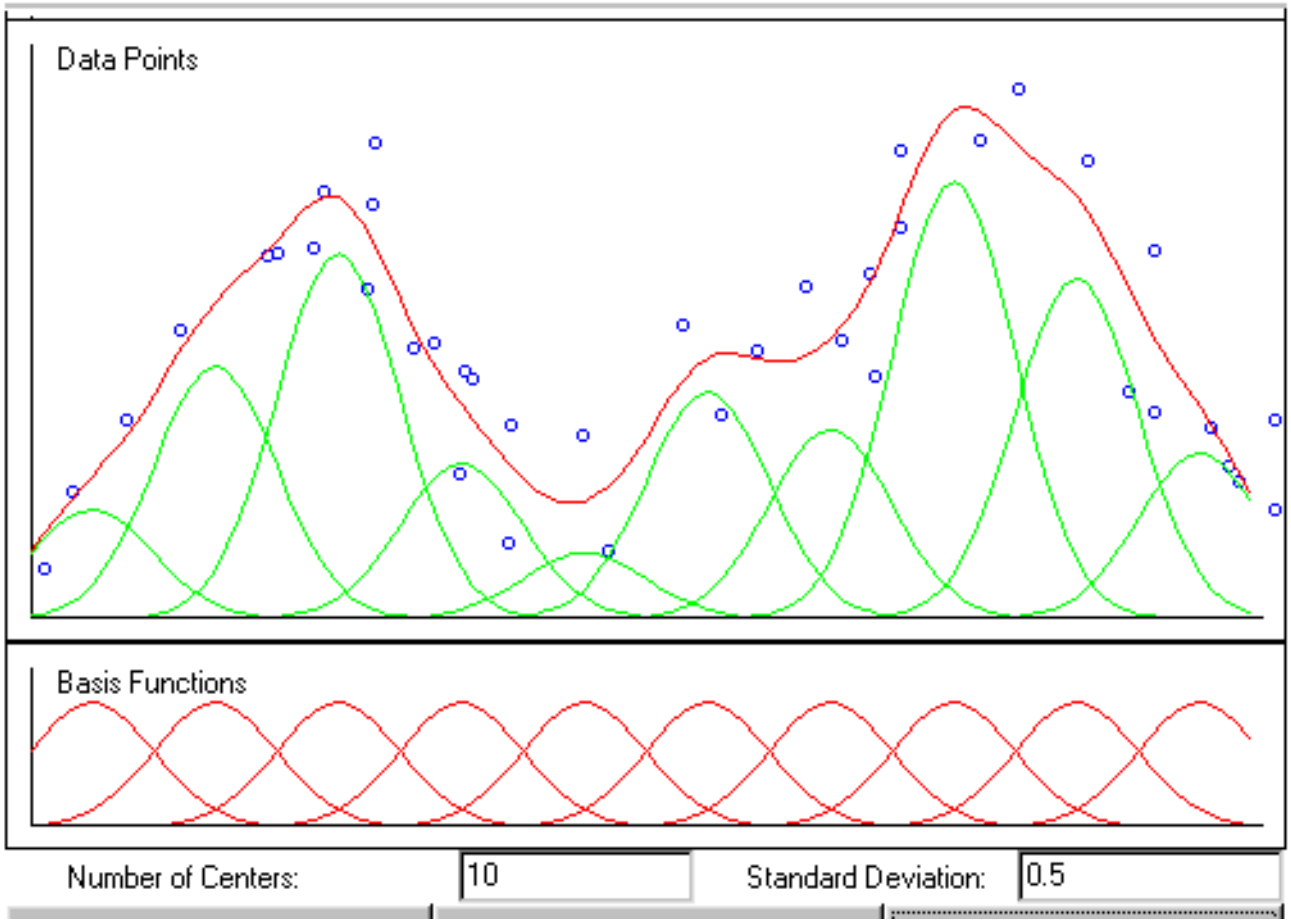


* Zapravo nije prava Gaussova gustoća vjerojatnosti (nedostaje normalizacijski faktor)

Radijalne bazne funkcije

- Prema (Hartman *et al.*, 1990), izraz (1) može aproksimirati bilo koju funkciju proizvoljno točno za dovoljno veliki broj Gaussovih jezgri uz uvjet da se varijance mogu nezavisno odrediti.
  - "Univerzalni aproksimator"

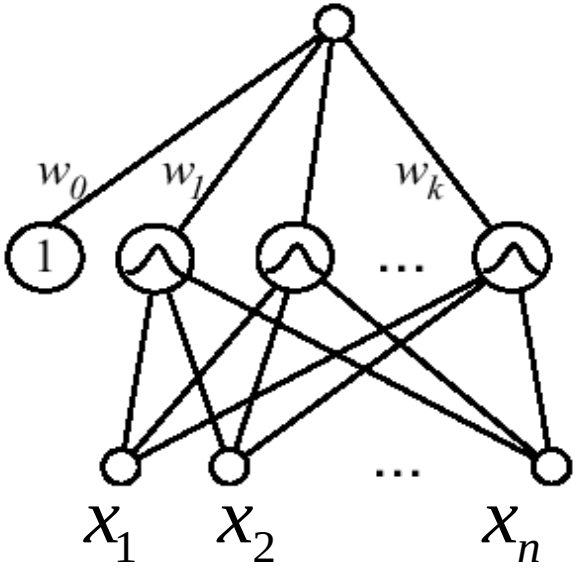
Radijalne bazne funkcije



http://diwww.epfl.ch/mantra/tutorial/english/rbf/html/index.html

Radijalne bazne funkcije

- Funkcija (1) se može interpretirati kao dvoslojna neuronska mreža:
- Prvi sloj računa $K_u(d(x_u,x))$
- Drugi sloj je linearna kombinacija vrijednosti prvog sloja



Radijalne bazne funkcije

- Parametri mreže RBF uče se u dva koraka:
  1. Određuje se broj skrivenih jedinica k, određuje se x_u i σ^2 koji određuju jezgrenu funkciju.
  2. Na temelju minimizacije zbroja kvadrata pogreške određuju se težinski faktori w_i tako da mreža odgovara podacima za učenje. Za vrijeme te faze jezgrene funkcije se ne mijenjaju pa je učenje efikasno.
- Nekoliko metoda za izbor broja k:
  1. Za svaki primjer za učenje $(x^{(i)},y^{(i)})$ po jedna jezgrena funkcija s centrom u $x^{(i)}$ i s dijeljenom varijancom.
     Na ovaj način RBF-u potpunosti odgovara primjerima za učenje (raspored RBF-ova je gušći tamo gdje ima više primjera).

Radijalne bazne funkcije

2. Broj jezgrenih funkcija < N (efikasniji način)

   Centri x_u mogu biti smješteni:

   - uniformno po X
   - neuniformno,
   - slučajnim izborom, izvlačeći primjere iz skupa za učenje u skladu s njihovom distribucijom (što nam za to treba?)
   - prototipovima grupa primjera za učenje (uz uporabu algoritma grupiranja)

- Zaključak:
  - RBF daju globalnu aproksimaciju ciljne funkcije kao **linearnu kombinaciju više lokalnih jezgrenih funkcija**.
  - mogu biti **trenirane efikasnije** od unaprijednih neuronskih mreža s algoritmom *backpropagation* (taj algoritam radi u dva koraka)