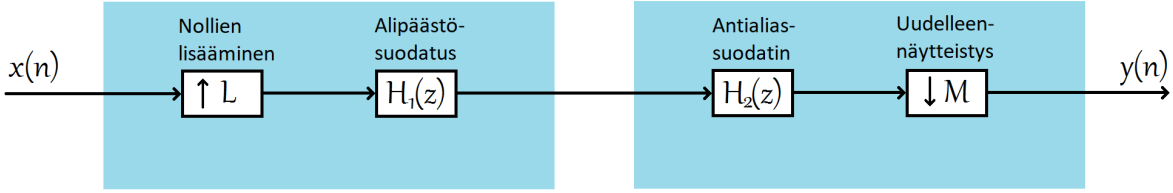
Näytteenottotaajuuden muunnos rationaalikertoimella

- Kuten aiemmin mainittiin, rationaalikertoiminen näytteenottotaajuuden muunnos saadaan yhdistämällä interpolointi ja desimointi, tässä järjestyksessä.
- Jos siis näytteenottotaajuus halutaan $\frac{\mathrm{L}}{\mathrm{M}}$-kertaiseksi, interpoloidaan signaali ensin L-kertaiseksi ja desimoidaan tämän jälkeen kertoimella M.
- Tässä kannattaa laskennan säästämiseksi supistaa murtoluku $\frac{\mathrm{L}}{\mathrm{M}}$ niin pitkälle kuin mahdollista.

Näytteenottotaajuuden muunnos rationaalikertoimella

- Järjestelmän lohkokaavio on alla olevan kuvan mukainen.



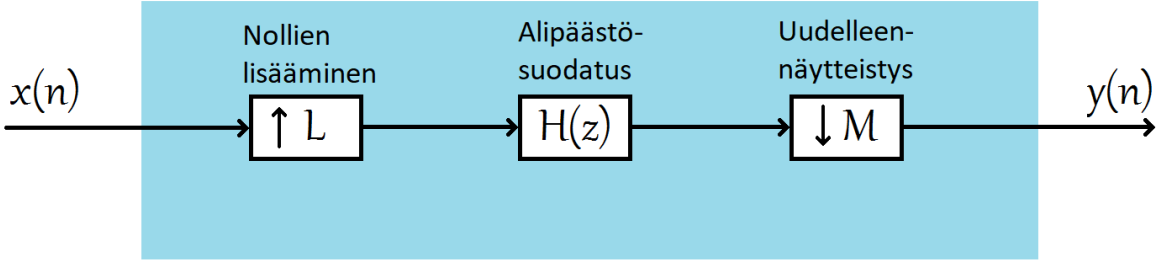
- Kaaviosta käy ilmi, että järjestelmässä on kaksi alipäästösuodatusta peräkkäin.
- Toinen näistä voidaan poistaa ja jättää jäljelle se, jonka suodatusvaatimukset ovat tiukemmat.

Näytteenottotaajuuden muunnos rationaalikertoimella

- Jos esimerkiksi kerroin $\frac{L}{M} = \frac{4}{3}$, tulee suotimen $H_1(z)$ päästökaistaksi $[0, \frac{1}{2L} - \Delta f] = [0, \frac{1}{8} - \Delta f]$ ja estokaistaksi $[\frac{1}{8}, \frac{1}{2}]$.
- Suotimen $H_2(z)$ vastaavat arvot ovat $[0, \frac{1}{2M} - \Delta f] = [0, \frac{1}{6} - \Delta f]$ ja $[\frac{1}{6}, \frac{1}{2}]$.
- Suodin $H_1(z)$ poistaa siis laajemman taajuusalueen kuin $H_2(z)$, joten pelkkä $H_1(z)$ riittää.
- Tässä yhteydessä täytyy luonnollisesti huomioida myös vaimennusvaatimusten toteutuminen.

Näytteenottotaajuuden muunnos rationaalikertoimella

- Alla oleva kaavio esittää näin saatavaa yksinkertaistettua järjestelmää.



Näytteenottotaajuuden pienentäminen useassa vaiheessa

- Desimaattoreiden toteutusta voidaan nopeuttaa pudottamalla näytteenottotaajuutta useammassa vaiheessa.
- Esimerkiksi näytteenottotaajuuden pudotus yhteen kymmenesosaan alkuperäisestä voidaan tehdä yhdessä vaiheessa kertoimella kymmenen tai kahdessa vaiheessa ensin kertoimella 5 ja sitten kertoimella 2.
- Kolmas mahdollisuus on pudottaa ensin kertoimella 2 ja sitten kertoimella 5.
- Kaikilla näillä menetelmillä tarvitaan eri määrä kertoimia, ja useimmiten suora pudotus kertoimella 10 tarvitsee enemmän kuin useammassa vaiheessa tehdyt operaatiot.

$$8 = 2 \times 4 = 4 \times 2 = 2 \times 2 \times 2$$

Näytteenottotaajuuden pienentäminen useassa vaiheessa

- Alla on esitetty kaaviot eri menetelmistä, kun lähtötaajuus on 20 kHz ja tavoite on 2 kHz.
- Oletetaan lisäksi, että signaalista täytyy säilyttää taajuudet 900 Hertsiin asti ja että suunnittelussa käytetään FIR-suodatinta ja Hamming-ikkunaa.
  - Yhdessä vaiheessa: $x(n) \rightarrow \boxed{H(z)} \rightarrow \boxed{\downarrow 10} \rightarrow y(n)$
    - päästökaista $[0, 0.9]$ kHz,
    - estokaista $[1, 10]$ kHz,
    - siirtymäkaista normalisoituna $[0.045, 0.05]$,
    - kertoimia tarvitaan $N = 3.3/\Delta f = 3.3/0.005 \approx \mathbf{661}$.

Näytteenottotaajuuden pienentäminen useassa vaiheessa

- Kahdessa vaiheessa: $x(n) \rightarrow \boxed{H_1(z)} \rightarrow \boxed{\downarrow 5} \rightarrow \boxed{H_2(z)} \rightarrow \boxed{\downarrow 2} \rightarrow y(n)$

  $H_1(z)$:
  - päästökaista $[0, 0.9]$ kHz,
  - estokaista $[2, 10]$ kHz,
  - siirtymäkaista normalisoituna $[0.045, 0.1]$,
  - kertoimia tarvitaan $N_1 = 3.3/0.055 \approx 61$.

  $H_2(z)$:
  - päästökaista $[0, 0.9]$ kHz,
  - estokaista $[1, 2]$ kHz,
  - siirtymäkaista normalisoituna (näytteenottotaajuudella 4 kHz) $[0.225, 0.25]$,
  - kertoimia tarvitaan $N_2 = 3.3/0.0250 \approx 133$.

  Yhteensä kertoimia tarvitaan siis **194**.

Näytteenottotaajuuden pienentäminen useassa vaiheessa

- Kahdessa vaiheessa: $x(n) \rightarrow \boxed{H_1(z)} \rightarrow \boxed{\downarrow 2} \rightarrow \boxed{H_2(z)} \rightarrow \boxed{\downarrow 5} \rightarrow y(n)$

  $H_1(z)$:
  - päästökaista $[0, 0.9]$ kHz,
  - estokaista $[5, 10]$ kHz,
  - siirtymäkaista normalisoituna $[0.045, 0.25]$,
  - kertoimia tarvitaan $N_1 = 3.3/0.205 \approx 17$.

  $H_2(z)$:
  - päästökaista $[0, 0.9]$ kHz,
  - estokaista $[1, 5]$ kHz,
  - siirtymäkaista normalisoituna (näytteenottotaajuudella 10 kHz) $[0.09, 0.1]$,
  - kertoimia tarvitaan $N_2 = 3.3/0.01 \approx 331$.

  Yhteensä kertoimia tarvitaan siis **348**.

Näytteenottotaajuuden pienentäminen useassa vaiheessa

- Näin ollen tehtävä on viisainta suorittaa kahdessa vaiheessa kertoimilla 5 ja 2 (tässä järjestyksessä).
- Yleisesti pitää paikkansa, että desimointikertoimet kannattaa sijoittaa laskevaan järjestykseen.
- Siksi järjestys $x(n) \rightarrow \boxed{H_1(z)} \rightarrow \boxed{\downarrow 2} \rightarrow \boxed{H_2(z)} \rightarrow \boxed{\downarrow 5} \rightarrow y(n)$ tuottaa aina enemmän kertoimia kuin $x(n) \rightarrow \boxed{H_1(z)} \rightarrow \boxed{\downarrow 5} \rightarrow \boxed{H_2(z)} \rightarrow \boxed{\downarrow 2} \rightarrow y(n)$, ja se olisi voitu alun perinkin jättää tarkastelematta.

Interpoloinnin käyttö D/A-muunnoksessa

- Seitsemänkymmentäluvun puolivälin jälkeen alettiin tutkia mahdollisuuksia musiikin ja muun äänimateriaalin digitaaliseen tallentamiseen.
- Lopputuloksena syntyi CD-soitin.
- Seuraavassa tarkastellaan sen joitakin teknisiä yksityiskohtia ja erityisesti sen D/A-muunnosvaihetta, jossa digitaalinen signaali muunnetaan analogiseksi.
- CD-levyn halkaisija on 12 cm ja raidat ovat 1.6 mikrometrin etäisyydellä toisistaan.
- Audio-CD:n sisältämä informaatio koostuu 16 bitin näytteistä, joita on otettu 44100 Hertzin näytteenottotaajuudella.
- Koska stereoääni tarvitsee kaksi kanavaa, saadaan bittimääräksi $2 \times 16 \times 44100 \approx 1.41$ miljoonaa bittiä sekunnissa.

Interpoloinnin käyttö D/A-muunnoksessa

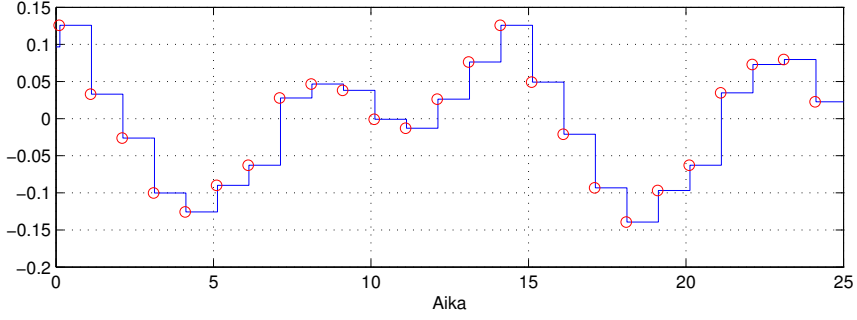
- Todellinen levyllä oleva bittimäärä on kuitenkin noin kolminkertainen.
- Ylimääräiset bitit ovat enimmäkseen virheenkorjausta varten.
- Pieni osa biteistä tarvitaan myös kappaleiden pituuksien ja mahdollisesti myös nimien tallentamiseen.
- Multirate-tekniikan yhteydessä mielenkiintoisin vaihe on CD-soittimen D/A-muunnos.

Nollannen asteen pitopiiri

- Yksinkertaisimmillaan muunnos digitaalisesta signaalista analogiseksi tapahtuu nollannen kertaluvun pitopiirillä (engl. zero-order hold; ZOH tai sample-and-hold; S/H).
- Tällöin analogisen signaalin arvoksi asetetaan viimeksi tullut digitaalisen signaalin arvo.
- Oheisessa kuvassa esitetään erään digitaalisen signaalin muunnos analogiseksi nollannen kertaluvun pitopiirillä.

Nollannen asteen pitopiiri

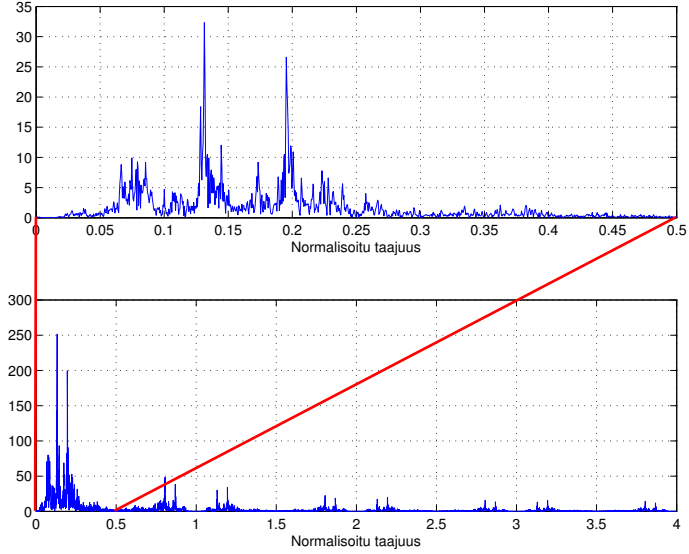
- Ympyrät kuvaavat digitaalisen signaalin arvoja ja viiva on analoginen approksimaatio.



Nollannen asteen pitopiiri

- Äänisignaalin ollessa kyseessä mielenkiintoisinta on kuinka analogisen signaalin spektri vastaa alkuperäisen digitaalisen signaalin spektriä.
- Vastaus löytyy seuraavan kuvan simuloiduista spektreistä.
- Ylemmässä kuvassa on digitaalisen signaalin spektri.
- Vaaka-akseli esittää taajuuksia näytteenottotaajuuden suhteen normalisoituina.
- Alempi kuvaaja esittää analogisen signaalin spektriä nollannen asteen pitopiirin jälkeen.

Nollannen asteen pitopiiri



Nollannen asteen pitopiiri

- Nyt signaalin energia jakautuu paljon laajemmalle alueelle siten, että alkuperäinen spektri (välillä $[0, 0.5]$) monistuu suuremmille taajuuksille.
- Välillä $[0.5, 1]$ on vaimentunut peilikuva alkuperäisestä spektristä, välillä $[1, 1.5]$ on alkuperäisen spektrin vaimennettu kopio, välillä $[1.5, 2]$ taas peilikuva vaimennettuna, jne.
- Kuvassa on esitetty ainoastaan alkuperäinen spektri ja sen seitsemän kopiota, mutta itse asiassa kopioita on äärettömän monta (pienemmillä ja pienemmillä energioilla).
- Nollannen asteen pitopiirin käyttäytyminen taajuustasossa voidaan esittää analyyttisesti.
- Koska kyseessä on jatkuva-aikainen suodin, analyysimenetelmä poikkeaa hieman johdatuskursseilla tarkastelluista.

Nollannen asteen pitopiiri

- Pitopiiri voidaan ajatella suotimeksi, jonka impulssivaste on jatkuva funktio

$$h(t) = \begin{cases} 1, & \text{kun } 0 \leqslant t < T, \\ 0, & \text{muulloin,} \end{cases}$$

  missä T on kahden näytteenottohetken välinen aikaero (CD-soittimella $1/44100$ s).
- Analogisen suotimen tapauksessa konvoluutio määritellään kaavalla

$$y(t) = \int_{-\infty}^{\infty} h(u)x(t-u)\,du$$

  eli tässä tapauksessa

$$y(t) = \int_{0}^{T} x(t-u)\,du.$$

Nollannen asteen pitopiiri

- Tällaisen suotimen taajuusvaste on

$$\begin{aligned} H(e^{i\omega}) &= \int_{-\infty}^{\infty} h(t) e^{-i\omega t}\, dt \\ &= \int_{0}^{T} e^{-i\omega t}\, dt \\ &= \Big/_{0}^{T} \frac{1}{-i\omega} e^{-i\omega t} \\ &= \frac{e^{-i\omega T} - 1}{-i\omega}. \end{aligned}$$

Nollannen asteen pitopiiri

- Viimeisin lauseke voidaan sieventää muotoon

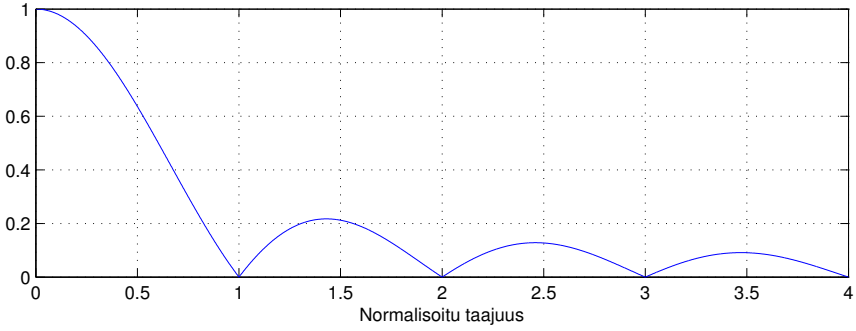
$$H(e^{i\omega}) = Te^{-i\omega T/2}\,\mathrm{sinc}(\omega T/2),$$

jolloin amplitudivasteeksi tulee

$$|H(e^{i\omega})| = T|\,\mathrm{sinc}(\omega T/2)|.$$

Nollannen asteen pitopiiri

- Tämän funktion kuvaaja on alla (nyt $T = 1$).

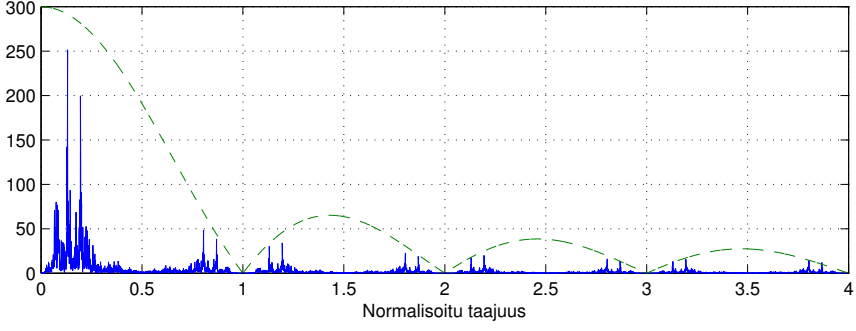


Nollannen asteen pitopiiri

- Digitaalisen signaalin näytteenottotaajuus on yo. asteikolla yksi ja Nyquistin rajataajuus näinollen kohdassa $1/2$.
- Jokainen kuvaajan huippukohta tuottaa yhden vaimennetun kopion digitaalisen signaalin spektristä.
- Amplitudivasteen kuvaaja sovitettuna signaalin spektriin on alla.

Nollannen asteen pitopiiri

- Visualisointisyistä amplitudivaste on nyt kerrottu kolmellasadalla.



Nollannen asteen pitopiiri

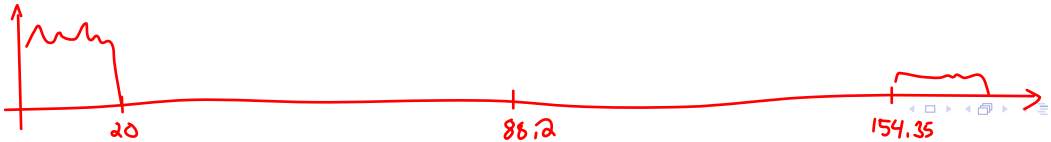
- Näin havaittiin, että pelkkä pitopiiri tuottaa ylimääräisiä taajuuksia analogiseen signaaliin.
- CD-soittimen tapauksessa ylimääräinen energia esiintyy yli 22.05 kilohertsin taajuuksilla, joita ihmiskorva ei kuule.
- Ylimääräinen energia korkeilla taajuuksilla saattaa kuitenkin rasittaa vahvistinlaitteistoa, joten se on hyvä poistaa.
- Korkeat taajuudet voitaisiin poistaa analogisella alipäästösuotimella, jonka päästökaista on esimerkiksi väli $[0, 20]$ kHz ja estokaista $[22.05, \infty)$ kHz (analogisella suotimilla käsiteltävillä taajuuksilla ei ole ylärajaa).
- Tällöin siirtymäkaistasta tulee kuitenkin verraten kapea, jolloin analogisesta suotimesta tulee hankala suunniteltava ja helposti kallis.

Nollannen asteen pitopiiri

- Multirate-menetelmiä käyttäen suodatus voidaan tehdä kahdessa vaiheessa, jolloin suotimista tulee yksinkertaisempia.
- Ensimmäisessä vaiheessa nostetaan digitaalisen signaalin näytteenottotaajuus nelinkertaiseksi (176.4 kHz).
- Tämä tapahtuu lisäämällä kahden näytearvon väliin kolme nollaa ja suodattamalla tulos alipäästösuotimella, jonka päästökaista on väli $[0, 20]$ kHz ja estokaista $[22.05, 88.2]$ kHz.
- Tämän suotimen toteutus riippuu halutusta tarkkuudesta, mutta esimerkiksi Philipsin kuvauksessa käytetään FIR-suodinta, jossa on 96 kerrointa.
- Kukin kerroin esitetään kahdentoista bitin tarkkuudella.
- Päästökaistalla suotimen amplitudivaste on nouseva niin, että nollataajuuden lähellä se on hieman alle nollan desibelin ja 20 kHz:n lähellä hieman yli nollan desibelin.

Nollannen asteen pitopiiri

- Näin pyritään kompensoimaan korkeampien taajuuksien pientä vaimenemista nollannen asteen pitopiirissä.
- Toisessa vaiheessa digitaalinen signaali muunnetaan analogiseksi nollannen asteen pitopiirillä ja suodatetaan lopuksi analogisella suotimella, joka poistaa suurille taajuuksille tulevan ylimääräisen energian.
- Nyt analogisen suotimen vaatimukset on helppo toteuttaa: päästökaista $[0, 20]$ kHz, estokaista $[88.2, \infty)$ kHz.
- Siirtymäkaistan leveys on nyt yli 30-kertainen aikaisempaan verrattuna.
- Itse asiassa siirtokaista voitaisiin venyttää yli 150:een kilohertsiin ilman ongelmia (miksi?).

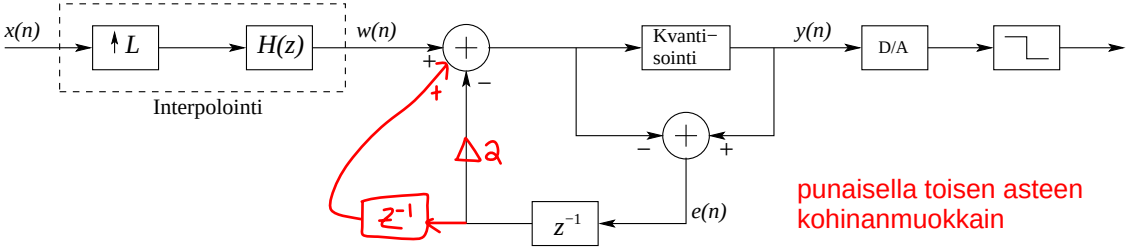


Kohinanmuokkaus

- Digitaalilaitteita käytettäessä informaation kaikkein luonnollisin esitysmuoto on binäärinen.
- Binääridatalla tehtävät operaatiot ovat nopeita ja niitä käyttävät laitteet yksinkertaisia suunnitella.
- Myös CD-soittimen alunperin 16-bittinen data muunnetaan nykyisissä soittimissa ensin binääriseksi ja vasta tämän jälkeen analogiseksi.
- Tällöin D/A-muunnin tuottaa vain kahta eri jännitetasoa ja se saadaan rakenteeltaan yksinkertaiseksi.
- Normaalisti signaalin muunnos binääriseksi aiheuttaisi voimakkaan kvantisointikohinan signaaliin, mutta siitä päästään eroon nostamalla signaalin näytteenottotaajuutta sekä siirtämällä kvantisointikohinaa korkeammille taajuuksille.

Kohinanmuokkaus

- Tällaisista menetelmistä käytetään nimeä **kohinanmuokkaus** (**engl. noise shaping**), ja ideana on nimenomaan siirtää kohinan energiaa suuremmille taajuuksille, josta se on helppo poistaa alipäästösuodatuksella.



- Yksinkertaisin kohinanmuokkausmenettely on yllä olevan lohkokaavion mukainen.

Kohinanmuokkaus

- Ensimmäisessä vaiheessa signaali $x(n)$ interpoloidaan näytteenottotaajuudeltaan moninkertaiseksi.
- Tulos $w(n)$ kvantisoidaan ja järjestelmä laittaa muistiin syntyneen kvantisointivirheen $e(n)$.
- Kvantisointivirheen tallentaminen auttaa tasapainottamaan virhettä seuraavilla kierroksilla.
- Yleensä kvantisointia mallinnetaan lisäämällä kvantisoitavaan signaaliin virhesignaali $e(n)$.

Kohinanmuokkaus

- Yllä olevan kuvan tapauksessa lohkokaaviosta saadaan yhtälö

$$y(n) = w(n) + e(n) - e(n-1).$$

Ottamalla z-muunnokset puolittain saadaan yhtälö

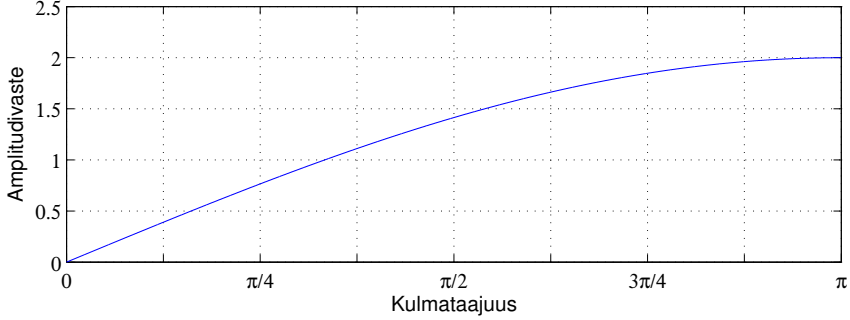
$$Y(z) = W(z) + E(z)(1 - z^{-1}).$$

- Näin ollen signaali $w(n)$ menee sellaisenaan järjestelmän läpi (ei suodatusta).
- Sen sijaan virhesignaali $e(n)$ kulkee lineaarisen järjestelmän $H(z) = 1 - z^{-1}$ läpi.
- Käytetään tästä ulostulosignaalista jatkossa merkintää $d(n)$:

$$d(n) = e(n) - e(n-1).$$

Kohinanmuokkaus

- Järjestelmän $H(z)$ taajuusvaste on $H(e^{i\omega}) = 1 - e^{-i\omega}$, jonka itseisarvon kuvaaja on alla.



- Melko helposti voidaan osoittaa, että $|1 - e^{-i\omega}| = 2\sin(\frac{\omega}{2})$.

Kohinanmuokkaus

- Kuviosta nähdään, että kohinan pienet taajuudet vaimenevat ja suuret taajuudet vahvistuvat.
- Koska signaali $w(n)$ on interpoloitu versio alkuperäisestä signaalista, sijaitsevat tärkeät taajuudet nimenomaan pienillä taajuuksilla, jossa kohina on vaimentunut.
- Signaalin $y(n)$ SNR riippuu nyt interpolointikertoimesta L ja siitä, kuinka moneen bittiin signaali kvantisoidaan.

Kohinanmuokkaus

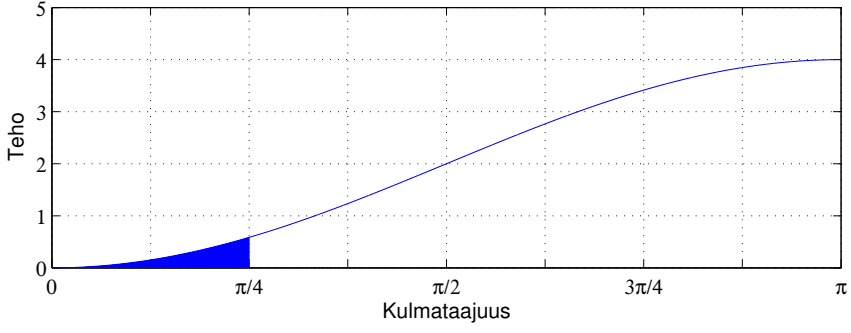
- Kohinan määrä järjestelmän $H(z)$ jälkeen voidaan laskea analyyttisesti käyttämällä hyväksi tietoa, että sen tehospektri riippuu valkoisen kohinan tapauksessa suoraan järjestelmän $H(z)$ taajuusvasteesta:

$$\begin{aligned} P_{dd}(e^{i\omega}) &= |H(e^{i\omega})|^2 \sigma_e^2 \\ &= \left(2 \sin\left(\frac{\omega}{2}\right)\right)^2 \sigma_e^2 \\ &= 4\sigma_e^2 \sin^2\left(\frac{\omega}{2}\right), \end{aligned}$$

  missä $\sigma_e^2 = 2^{-2b}/12$ ja b on bittien määrä kvantisoinnin jälkeen.
- Koska signaali sijaitsee interpoloinnin jälkeen taajuuksilla $\omega \in [0, \pi/L]$, meidän tarvitsee tietää tällä taajuusalueella olevan kohinan teho.

Kohinanmuokkaus

- Esimerkiksi tapauksessa $\mathrm{L} = 4$ tilanne olisi alla olevan kuvan mukainen.
- Käyrän alle jäävä merkitty alue vastaa signaalin kanssa samalla taajuusalueella olevan kohinan tehoa.



Kohinanmuokkaus

- Se saadaan integroimalla,

$$\begin{aligned}\frac{1}{\pi}\int_0^{\pi/L} 4\sigma_e^2 \sin^2(\frac{\omega}{2})\,d\omega &= 4\cdot\frac{2^{-2b}}{12\pi}\int_0^{\pi/L}\sin^2(\frac{\omega}{2})\,d\omega \\ &= \frac{2^{-2b}}{3\pi}\left(\frac{\pi}{2L}-\frac{1}{2}\sin(\frac{\pi}{L})\right) \\ &= \frac{2^{-2b}}{6}\left(\frac{1}{L}-\frac{\sin(\pi/L)}{\pi}\right).\end{aligned}$$

- Nyt voidaan kysyä, montako bittiä säästetään kun interpoloidaan tietyllä kertoimella L.

Kohinanmuokkaus

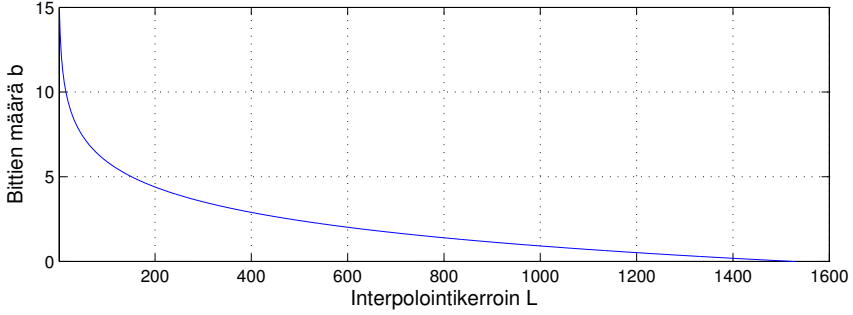
- Esimerkiksi CD-soittimen tapauksessa bittimäärä on $15 + 1$, ja kvantisointikohinan teho näin ollen $\frac{2^{-30}}{12}$.
- Samaan kvantisointikohinaan päästään interpoloinnin ja kohinanmuokkauksen avulla interpolointikertoimella L ratkaisemalla tarvittava bittimäärä b yhtälöstä

$$\frac{2^{-2\mathrm{b}}}{6}\left(\frac{1}{\mathrm{L}} - \frac{\sin(\pi/\mathrm{L})}{\pi}\right) = \frac{2^{-30}}{12}.$$

- Bittien tarve eri kertoimilla on esitetty alla olevassa kuvassa.

Kohinanmuokkaus

- Arvoissa ei ole mukana merkkibittiä.



- Kuviosta nähdään interpolointikertoimien olevan edelleenkin varsin suuria.

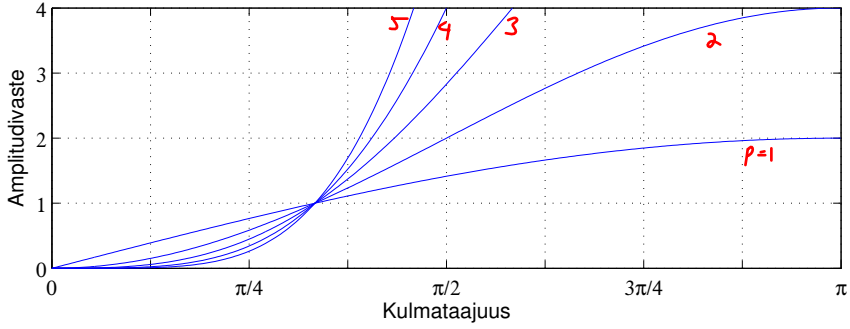
Kohinanmuokkaus

- Täysin binääriseen tilanteeseen ($b = 0$) tarvitaan nyt 1523-kertainen interpolointi, mikä on liikaa.
- Järjestelmää on siis kehitettävä edelleen.
- Kohinanmuokkaus tapahtuu siis yksinkertaisimmillaan viemällä kvantisointikohina järjestelmän $H(z) = 1 - z^{-1}$ läpi.
- Tulosta voidaan parantaa korottamalla siirtofunktio toiseen tai suurempaan potenssiin.
- Toisen asteen kohinanmuokkaus käyttää siis järjestelmää $H(z) = (1 - z^{-1})^2 = 1 - 2z^{-1} + z^{-2}$, joka on melko helppo liittää aiemmin esillä olleeseen lohkokaavioon.

Kohinanmuokkaus

- Yleisemmin p:nnen asteen kohinanmuokkaus vie kohinan järjestelmän $H(z) = (1 - z^{-1})^p$ läpi.
- Tällöin amplitudivasteeksi tulee $|H(e^{i\omega})| = (2\sin(\frac{\omega}{2}))^p$ ja amplitudivasteen kuvaajat arvoilla $p = 1, 2, \ldots, 5$ on esitetty alla.
- Käyrät kohtaavat pisteessä ω, jossa on voimassa yhtälö $2\sin(\omega/2) = 1$, eli pisteessä $\omega = \pi/3$, joka on normalisoituina taajuuksina $f = \frac{\omega}{2\pi} = \frac{1}{6}$.

Kohinanmuokkaus



Kohinanmuokkaus

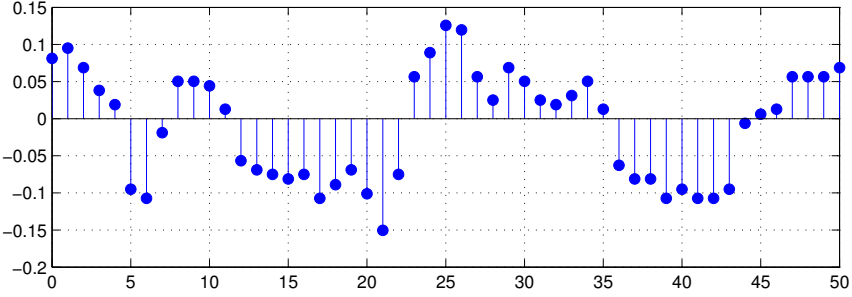
- Näin ollen p:nnen asteen kohinanmuokkaimen kvantisointikohinan varianssi on

$$\frac{2^{2(\mathrm{p}-\mathrm{b})}}{12\pi}\int_0^{\pi/\mathrm{L}} \sin^{2\mathrm{p}}\left(\frac{\omega}{2}\right)\,\mathrm{d}\omega.$$

- Tutkitaan lopuksi toisen asteen kohinanmuokkaajaa, kun testisignaalin alkuperäinen näytteenottotaajuus on 8192 Hz, $\mathrm{L} = 16$ ja $\mathrm{b} = 0$ eli kvantisoinnin tulos on binäärinen.
- Interpolointikerroin L on visualisointisyistä melko pieni.

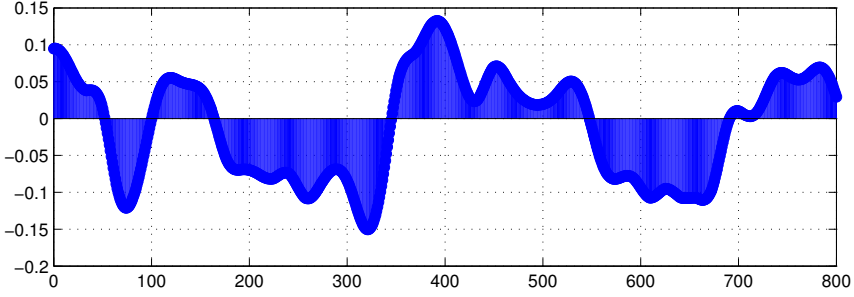
Kohinanmuokkaus

- Alla olevassa kuvassa on pätkä testisignaalia.



Kohinanmuokkaus

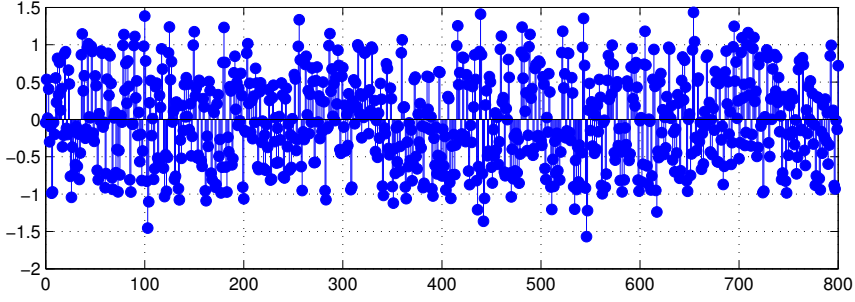
- Ensimmäisessä vaiheessa signaali interpoloidaan; tulos on alla.



- Interpoloinnin jälkeen signaali kvantisoidaan yhteen bittiin, eli käytännössä jäljelle jää vain näytteen merkki.

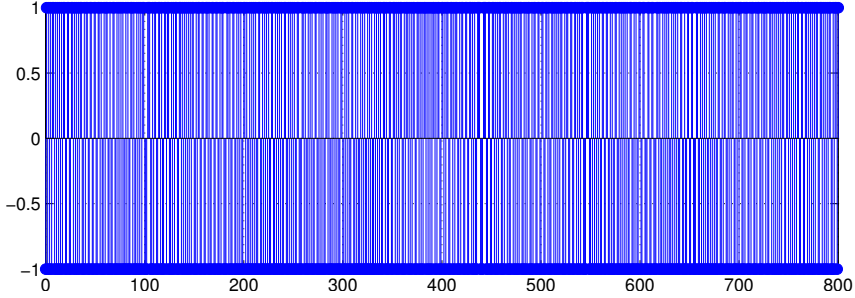
Kohinanmuokkaus

- Huomaa, että kyseessä ei ole alkuperäisen signaalin merkki vaan mukana on myös edellisen askeleen virhesignaali.



Kohinanmuokkaus

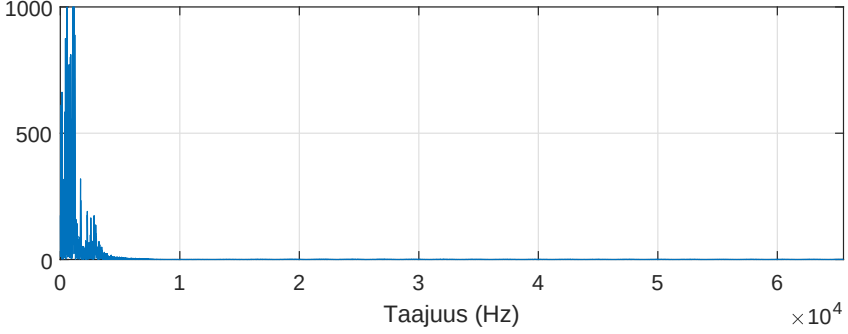
- Järjestelmän ulostulosignaali näyttää seuraavalta.



- Signaali ei näytä aikatasossa juurikaan samalta kuin alkuperäinen, mutta taajuustasossa yhtäläisyydet silti löytyvät.

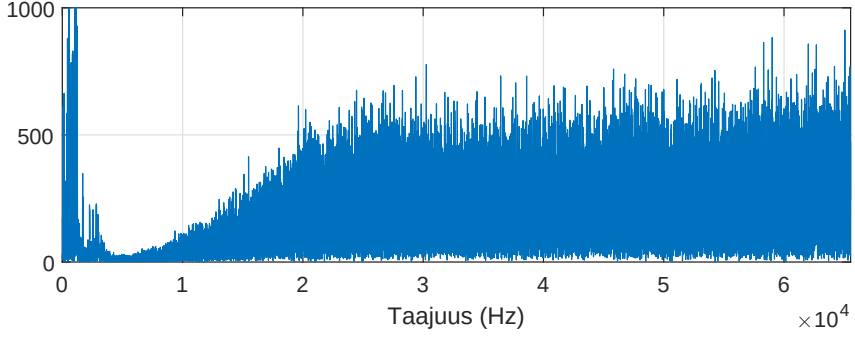
Kohinanmuokkaus

- Ensin alkuperäisen interpoloidun signaalin spektri.



Kohinanmuokkaus

- Kun tätä verrataan alla olevaan tulossignaalin spektriin, nähdään, että pienillä taajuuksilla spektrit ovat kokolailla samanlaisia.



Kohinanmuokkaus

- Lisäksi havaitaan suuremmilla taajuuksilla spektrin noudattavan amplitudivasteen $|\mathrm{H}(e^{\mathrm{i}\omega})| = 4\sin^2(\frac{\omega}{2})$ muotoa.
- Nyt interpolointikerroin L ei ollut riittävän suuri, joten signaalin ja kohinan spektrit ovat osittain samalla taajuusalueella.
- Seuraavaksi kvantisoitu signaali muunnetaan analogiseksi yksinkertaisella yksibittisellä D/A-muuntimella, jonka jälkeen analoginen signaali suodatetaan analogisella alipäästösuotimella.
- Analoginen suodin poistaa signaalista kohinan taajuudet.
- Jos interpolointikerroin ja järjestelmän aste olisivat olleet riittävän suuria, signaalin ja kohinan spektrit olisivat melko selkeästi erillään.
- Tällöin analogisen alipäästösuotimen suunnittelu olisi melko helppoa, koska siirtymäkaista saataisiin riittävän leveäksi.

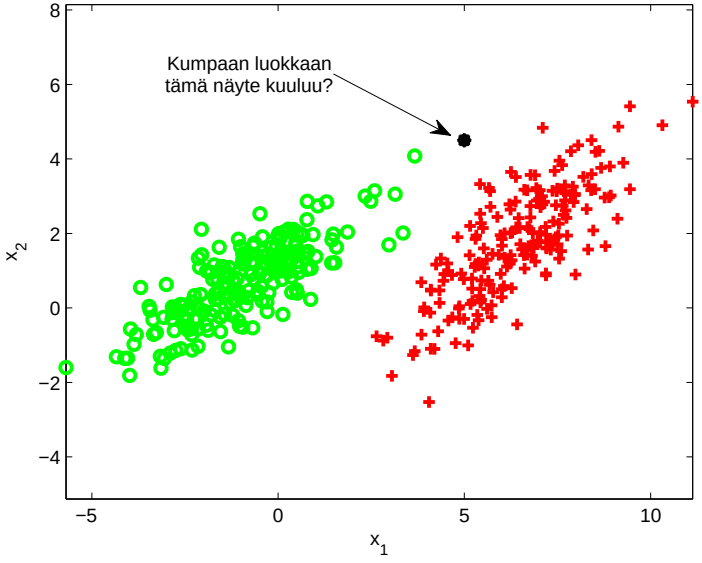
Oppivat järjestelmät

- Tässä kappaleessa tutustutaan ns. **oppiviin järjestelmiin**, joille opetetaan laskennallinen ongelma esimerkkien avulla.
- Yksi merkittävä osa tähän luokkaan kuuluvia järjestelmiä ovat ns. **luokittelijat**, joiden täytyy esimerkkien perusteella oppia luokittelemaan moniulotteinen data $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in \mathbb{R}^n)$ äärelliseen määrään luokkia: $y_1, y_2, \ldots, y_N \in \{1, 2, \ldots, M\}$.
- Oppivien järjestelmien sovelluskohteisiin signaalinkäsittelyssä kuuluvat mm. tekstintunnistus ja puheentunnistus.
- Näiden lisäksi menetelmiä käytetään esimerkiksi hakukoneissa, lääketieteellisessä diagnostiikassa sekä mm. sähköpostisuodattimissa.

Oppivat järjestelmät

- Kaikille näille järjestelmille on yhteistä, että ne oppivat luokittelun esimerkkien avulla: esimerkiksi tekstintunnistuksessa luokittelijalle näytetään kustakin kirjaimesta satoja tai tuhansia esimerkkejä, joiden perusteella opetusalgoritmi valitsee luokittelijalle sopivat parametrit.
- Esimerkiksi käsinkirjoitettujen merkkien tunnistuksessa yleisesti käytetty MNIST-tietokanta koostuu yhteensä noin 70000 käsinkirjoitettua kirjainta esittävästä bittikartasta.
- Yleisesti käytettyihin luokittelumenetelmiin kuuluvat mm. **Fisherin lineaarinen erottelija**, **tukivektorikone** sekä **hermoverkko**.
- Menetelmiä tarkastellaan jatkossa esimerkin avulla, jossa alla olevan kuvan mukaisten kaksiulotteisten joukkojen (punaiset ristit ja vihreät ympyrät) avulla pitäisi oppia päättelysääntö uusien näytteiden luokittelemiseksi (kuten kuvan musta pallo).

Oppivat järjestelmät



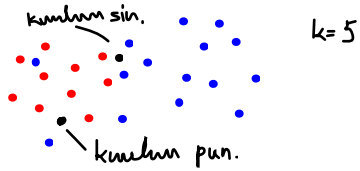
Oppivat järjestelmät

- Ensinäkemältä ongelma saattaa vaikuttaa helpolta, koska tässä kaksiulotteisessa tapauksessa kuka tahansa osaa piirtää kohtuullisen hyvän rajan joukkojen välille.
- Ongelmasta tulee kuitenkin haasteellisempi kun joukkojen dimensio kasvaa ja ne menevät osin päällekkäin.
- Esimerkiksi edellä mainitun MNIST-tietokannan näytteet ovat 20×20 pikselin kokoisia kuvia, joten avaruus on 400-ulotteinen.
- Tärkeitä käsitteitä luokittelussa sekä yleensäkin koneoppimisessa ovat **opetusjoukko** ja **testijoukko**.
- Opetusjoukkoa käytetään nimensä mukaisesti järjestelmän opetuksessa, mutta lopputuloksen toimivuutta arvioidaan erillisen testijoukon avulla.

Oppivat järjestelmät

- Todellisessa tilanteessahan luokittelija luokittelee enimmäkseen opetusjoukkoon kuulumattomia näytteitä, joten opetusjoukon luokitteluvirhe ei ole luotettava mittari.
- Esimerkiksi edellä mainittu MNIST-tietokannan 70000 merkkiä on jaettu 60000:n opetusmerkkiin ja 10000:n testikuvaan.

Lähimmän naapurin luokittelija



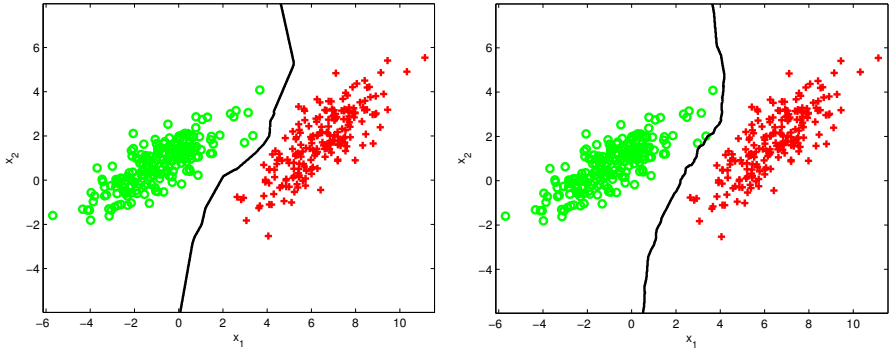
- Yksinkertainen ja helposti toteutettava ratkaisu luokitteluongelmaan on niin sanottu lähimmän naapurin luokittelija (engl. Nearest Neighbor Classifier; NN-classifier).
- Nimensä mukaisesti menetelmä etsii opetusaineistosta samankaltaisimman näytteen ja valitsee luokiteltavan pisteen luokan sen perusteella.
- Yleisempi versio tästä varsin intuitiivisesta luokittelijasta on ns. k-NN-luokittelija, joka etsii opetusdatasta k samankaltaisinta näytettä ja valitsee näiden joukosta yleisimmän luokan.
- Menetelmä ei siis vaadi erillistä opetusvaihetta, vaan se muistaa koko opetusmateriaalin sellaisenaan.
- Tämä on myös menetelmän suurin ongelma: uuden näytteen luokkaa pääteltäessä täytyy koko aineisto käydä läpi ja etsiä sieltä lähimpänä olevat.

Lähimmän naapurin luokittelija

- Lisäksi koko aineiston täytyy olla muistissa, mikä voi tulla ongelmaksi suurilla aineistoilla.
- Pienille opetusdatan määrille k-NN sen sijaan toimii yleensä yhtä hyvin kuin mikä tahansa muukin luokittelija, ja onkin usein käytetty helpohkoissa luokitteluongelmissa, joissa luokat erottuvat toisistaan selvästi.
- Alla olevassa kuvassa on piirretty k-NN-luokittelijan luokkarajat aiemmin esillä olleen kahden luokan testidatan tapauksessa.

Lähimmän naapurin luokittelija

- Vasemmanpuoleisessa kuvassa on 1-NN-luokittelijan tulos ja oikeanpuoleisessa 9-NN-luokittelijan tulos.

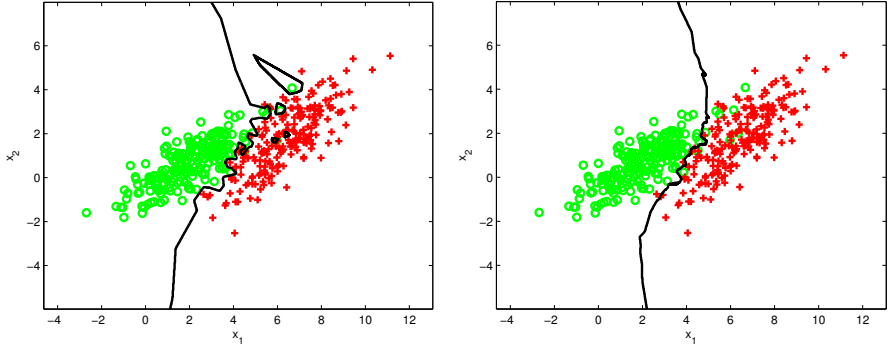


Lähimmän naapurin luokittelija

- Kuvasta nähdään, että suuremmilla parametrin k arvoilla menetelmästä tulee epäherkempi yksittäisille poikkeaville näytteille.
- Oikeanpuoleisessa kuvassa oikeanpuolimmaiset vihreät ympyrät eivät vaikuta luokkarajaan yhtä paljon, koska ne ovat melko irrallisia muista näytteistä ja näin ollen sattuman tulosta.
- Jos luokat ovat enemmän päällekkäin, tulee 1-NN-luokittelijan luokkarajasta melko rikkonainen.

Lähimmän naapurin luokittelija

- Alla on esimerkki tällaisesta tilanteesta.



Lähimmän naapurin luokittelija

- Vasemmalla olevassa 1-NN-luokittelijan tuloksessa jokainen näytepiste vaikuttaa lopputulokseen, ja luokkaraja koostuu monesta irrallisesta osasta.
- Oikealla olevassa 9-NN-luokittelijan kuvassa luokkaraja on yhtenäinen ja luultavasti paremmin toimiva.
- Seuraavaksi tutustutaan monimutkaisempiin luokittelijoihin, joiden esitysmuoto on k-NN-luokittelijaa kompaktimpi.
- Kompakti esitys voi olla esimerkiksi funktio $\mathrm{f}(\mathbf{x})$, jonka arvo (esim. positiivinen vai negatiivinen) määrää näytteen $\mathbf{x}$ luokan.
- Tällöin luokan valinta on nopeampaa, mutta hyvän funktion f päättely vie enemmän aikaa.
- Tämä on yleensä toivottu toimintatapa, koska opetusvaihe ei ole reaaliaikainen eikä opetuksen kesto ole niin kriittinen tekijä kuin itse luokittelun.

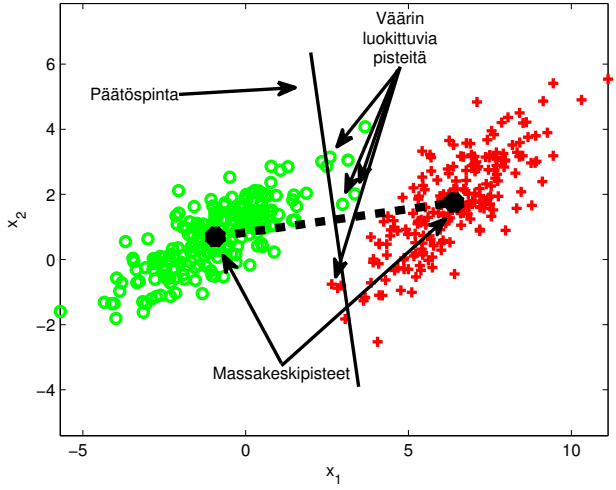
Lineaarinen erottelija

- Fisherin lineaarinen erottelija eli lineaarinen diskriminantti (**Linear Discriminant Analysis; LDA**) on julkaistu jo vuonna 1936, mutta on edelleen yleisesti käytetty ratkaisu pääasiassa yksinkertaisuutensa vuoksi.
- Perusideana on ratkaista se suora, jolle projisoitu moniulotteinen data maksimoi luokkien välisen varianssin (eli vie luokat mahdollisimman kauas toisistaan).
- Ensimmäinen mieleen tuleva ratkaisu on vetää raja suunnilleen joukkojen puoliväliin.
- Kaksiulotteiselle datalle tämä voidaan määrittää käsinkin, mutta esim. sataulotteiselle datalle tarvitaan automaattinen menetelmä rajan (eli ns. päätöspinnan) määräämiseksi.

Lineaarinen erottelija

- Yksinkertaisin ratkaisu on piirtää jana joukkojen massakeskipisteiden välille, ja vetää (ortogonaalinen) päätöspinta niiden puoliväliin.
- Tämä ratkaisu on esitetty alla olevassa kuvassa.

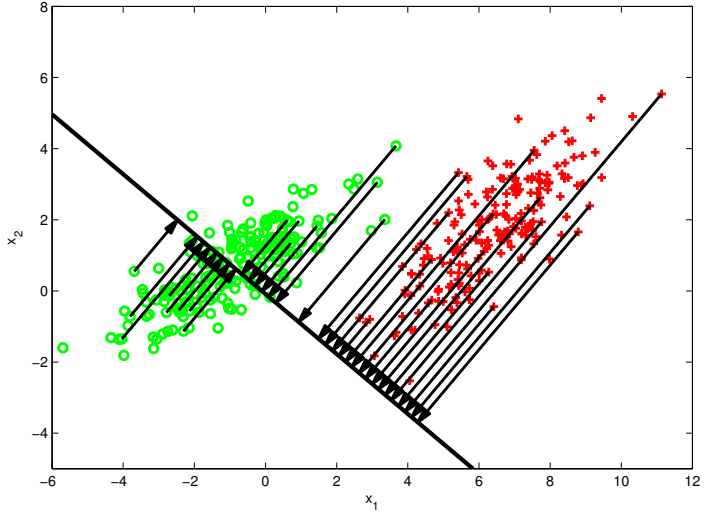
Lineaarinen erottelija



Lineaarinen erottelija

- Tuloksesta nähdään, että ratkaisu ei ole paras mahdollinen, jos luokat ovat vähänkään limittäin.
- Nyt nimittäin vääriä päätöksiä jää valittiinpa rajan sijainti miten hyvänsä, jos suunta on kuvan mukainen.
- Parempi ratkaisu olisi suora, jolle projisoitaessa pistejoukot tulevat toisistaan erillisiksi yksiulotteisiksi joukoiksi.
- Esimerkiksi alla olevassa kuvassa on edellä olleet kaksi pisteparvea projisoitu eräälle tällaiselle suoralle (pisteen projektio kuvaa sen lähimmäksi suoralla olevaksi pisteeksi).

Lineaarinen erottelija



Lineaarinen erottelija

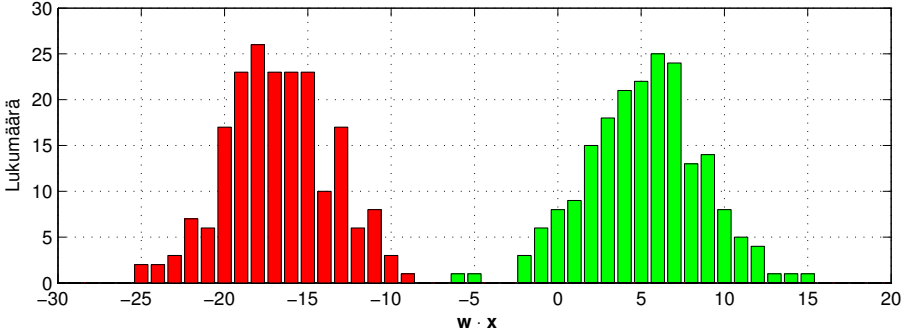
- Kuvion perusteella havaitaan, ettei erottelun kannalta ole väliä missä suora sijaitsee, ainoastaan sen suunnalla on merkitystä.
- Tämän vuoksi riittää tarkastella pelkästään suoran suunnan määrävää vektoria $\mathbf{w}$.
- Pisteen $\mathbf{x}$ projektio vektorille $\mathbf{w}$ määritellään kaavalla

$$\text{proj}_{\mathbf{w}}(\mathbf{x}) = \frac{\mathbf{w} \cdot \mathbf{x}}{|\mathbf{w}|^2}\mathbf{w}.$$

- Projisoidun pisteen sijainti suoralla määräytyy pistetulon $\mathbf{w} \cdot \mathbf{x}$ perusteella, joten yksinkertaisuuden vuoksi riittää tarkastella pelkästään sitä.

Lineaarinen erottelija

- Suureen $\mathbf{w} \cdot \mathbf{x}$ jakauma on esitetty alla olevassa kuvassa.



Lineaarinen erottelija

- Näyttäisi siis, että käytetty suora on hyvä valinta, ja esimerkiksi seuraava yksinkertainen algoritmi vaikuttaisi olevan hyvä luokittelija kyseiselle datalle:

$$\text{Pisteen } \mathbf{x} \text{ luokka} = \begin{cases} \text{vihreä ympyrä, jos} & \mathbf{w} \cdot \mathbf{x} \geqslant -7.5, \\ \text{punainen risti, jos} & \mathbf{w} \cdot \mathbf{x} < -7.5. \end{cases}$$

- Kuinka vektori $\mathbf{w}$ sitten valitaan?
- Tässä yksinkertaisessa tapauksessa se onnistuu käsinkin, mutta 100-ulotteisten osin päällekkäin olevien luokkien tapauksessa automaattinen menetelmä on tarpeen.

Lineaarinen erottelija

- Fisher muotoili kysymyksen optimointiongelman muotoon: mikä on se vektori $\mathbf{w}$, jolle projisoitaessa luokkien välinen varianssi maksimoituu suhteessa luokkien sisäiseen varianssiin.
- Kyseinen vektori saadaan ratkaistua kirjoittamalla Fisherin määritelmän mukaiset lausekkeet auki ja sieventämällä tulos.
- Melko suoraviivaisen matriisilaskennan avulla voidaan osoittaa että tämä suure saa muodon (Huomaa, että kaavassa käytetään pistetulon kanssa yhtäpitävää matriisituloa $\mathbf{w} \cdot \mathbf{x} = \mathbf{w}^{\mathrm{T}}\mathbf{x}$.)

$$J(\mathbf{w}) = \frac{\text{luokkien välinen varianssi}}{\text{luokkien sisäinen varianssi}} = \frac{\mathbf{w}^{\mathrm{T}}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^{\mathrm{T}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}(\mathbf{C}_1 + \mathbf{C}_2)\mathbf{w}},$$

missä $\boldsymbol{\mu}_1$ ja $\boldsymbol{\mu}_2$ ovat luokkien keskipisteet ja $\mathbf{C}_1$ ja $\mathbf{C}_2$ niiden otoskovarianssimatriisit.

Lineaarinen erottelija

- Etsimällä gradientin nollakohta voidaan osoittaa, että tämän lausekkeen maksimoiva vektori $\mathbf{w}$ on

$$\mathbf{w} = (\mathbf{C}_1 + \mathbf{C}_2)^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2).$$

- Edellä olleen esimerkin tapauksessa kovarianssimatriiseilla ja keskipisteillä on seuraavat arvot

$$\mathbf{C}_1 = \begin{pmatrix} 2.4775 & 1.3271 \\ 1.3271 & 1.1239 \end{pmatrix}, \mathbf{C}_2 = \begin{pmatrix} 2.2666 & 1.7515 \\ 1.7515 & 2.1657 \end{pmatrix}, \boldsymbol{\mu}_1 = \begin{pmatrix} -0.9330 \\ 0.6992 \end{pmatrix}, \boldsymbol{\mu}_2 = \begin{pmatrix} 6.3992 \\ 1.7508 \end{pmatrix},$$

jolloin $\mathbf{w} = [-3.4077, 2.8695]^\mathsf{T}$, joka on aiemmassa kuvassa olleen projektiosuoran suuntavektori.

Lineaarinen erottelija

- Huomaa, että kaikki samansuuntaiset (ja vastakkaissuuntaiset) vektorit kelpaavat yhtä lailla.
- Usein onkin tapana normalisoida vektori ykkösen mittaiseksi: $\mathbf{w}' = \mathbf{w}/\|\mathbf{w}\|$.
- Näin saadun luokittelijan ohjelmallinen toteutus on äärimmäisen yksinkertainen.
- Uuden näytteen $\mathbf{x} = (\mathrm{x}(1), \mathrm{x}(2))$ luokka päätellään laskemalla pistetulo

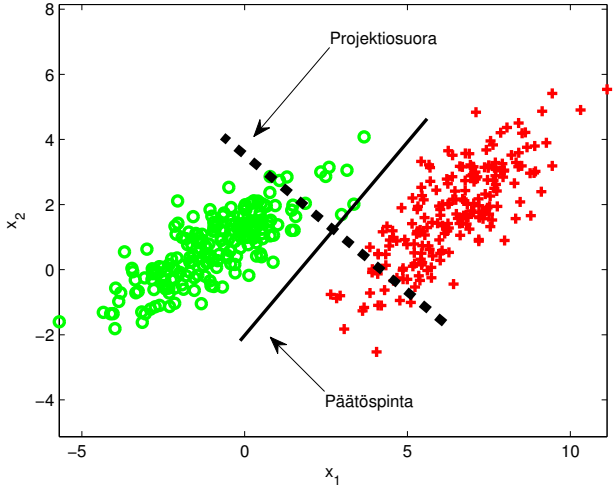
$$\mathbf{x} \cdot \mathbf{w} = -3.4077\mathrm{x}(1) + 2.8695\mathrm{x}(2)$$

  ja vertaamalla tulosta kynnysarvoon c (esim. edellä ollut raja $\mathrm{c} = -7.5$).
- Todellisessa tilanteessa kynnyksen valinta ”kuvasta katsomalla”on liian epätarkkaa, joten tähänkin tarvitaan automaattinen menetelmä.

Lineaarinen erottelija

- Paras tapa on valita c niin, että väärin menneiden opetusnäytteiden osuus on mahdollisimman pieni (esimerkin tapauksessa kaikki kynnykset väliltä $c \in [-8.9267, -5.6639]$ tuottavat nollavirheen, joten ne kaikki kelpaisivat).
- Yksinkertaisempi tapa on valita kynnys joukkojen keskipisteiden puolivälistä (kuten teemme harjoituksissa), jolloin saadaan arvo $c = -5.7984$.
- Tällä kynnysarvolla saatava päätöspinta on alla olevassa kuvassa. Kuvasta nähdään menetelmän jakavan opetusjoukot kahtia täysin oikein.

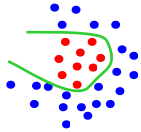
Lineaarinen erottelija



Lineaarinen erottelija

- Päätöspinnan lähellä on kuitenkin nyt kaksi vihreää pistettä, jotka luokittuvat juuri ja juuri oikein.
- Tätä ratkaisua turvallisemmalta tuntuisikin valinta, jossa marginaali päätöspinnan molemmin puolin olisi mahdollisimman suuri.
- Tähän ajatukseen perustuvat tukivektorikoneet, joita käsitellään seuraavaksi.

Tukivektorikoneet

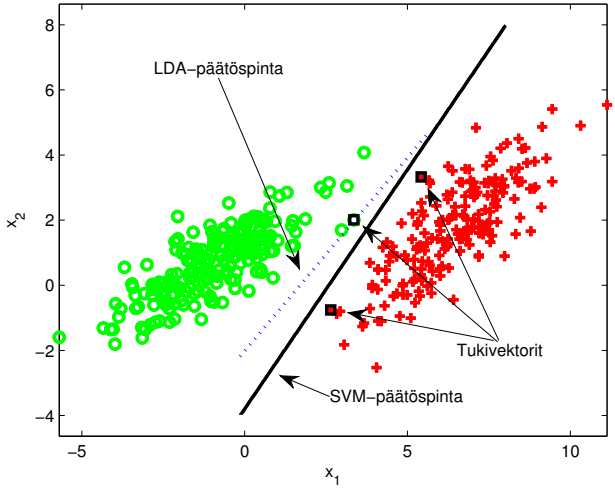


- Tukivektorikone (**engl. support vector machine (SVM)**) kehitettiin alunperin 1960-luvulla Moskovassa, mutta laajempi suosio syntyi vasta 1990-luvun lopulla tehokkaiden laskentamenetelmien sekä kehittyneemmän teorian myötä.
- Menetelmän ideana on valita se päätöspinta, joka maksimoi luokkien väliin jäävän marginaalin (eli on mahdollisimman keskellä pisteparvien välissä).
- Esimerkiksi yllä ollut LDA:n päätöspinta on piirretty luokkien keskipisteiden puoliväliin, mutta se sattuu olemaan lähempänä vihreitä ympyröitä kuin punaisia ristejä.
- SVM piirtää päätöspinnan niin, että etäisyys molempien luokkien lähimpiin alkioihin on mahdollisimman suuri.

Tukivektorikoneet

- Tukivektoreiden ja päätöspinnan ratkaisu on varsin matemaattinen ja tulee vastaan myöhemmillä kursseilla.
- SVM:stä on kuitenkin useita toteutuksia, joita voi käyttää vaikka algoritmin toiminta ei olisikaan täysin selvä.
- Suosittuja toteutuksia ovat mm. LibSVM sekä Matlabin Bioinformatics-toolboxin toteutus.
- Optimiratkaisu on kuvattu alla olevassa kuvassa.

Tukivektorikoneet



Tukivektorikoneet

- Kuvan sininen katkoviiva kuvaa LDA:n päätöspintaa ja musta yhtenäinen viiva SVM:n päätöspintaa, joka lepää ns. **tukivektoreiden** varassa.
- Kaikkiin kolmeen tukivektoriin on nyt sama etäisyys (eli marginaali) ja tukivektorit on merkitty mustilla neliöillä.
- Tukivektorikoneet siis maksimoivat päätöspinnan ympärille jäävän marginaalin ja valitsevat tukivektorit sen mukaan.
- Tämä ominaisuus ei pääse täysin oikeuksiinsa lineaarisen päätöspinnan tapauksessa, koska LDA:n päätöspinta on usein melko lähellä SVM:n pintaa.
- Marginaalin maksimointi kuitenkin korostuu kun data on korkeampiulotteista, ja marginaalit tyypillisesti suuremmat.

Tukivektorikoneet

- Tähän havaintoon perustuu SVM:n suosio ns. **kernelitempun** (**engl. kernel trick**) yhteydessä, joka esiteltiin jo vuonna 1964 käytettäväksi LDA:n kanssa.
- Se ei kuitenkaan saavuttanut merkittävää suosiota, koska se ei toimi LDA:n kanssa erityisen hyvin.
- Kernelitempun idea on kuvata data korkeampiulotteiseen avaruuteen ns. kernelin eli ytimen avulla.
- Ytimeksi sopii mikä tahansa tietyt ehdot toteuttava funktio, ja tarkoituksena on että data olisi luokiteltavissa korkeammassa ulottuvuudessa lineaarisen päätöspinnan avulla.

Tukivektorikoneet

- Ratkaisua kutsutaan tempuksi sen vuoksi, että sen toteuttamiseksi dataa ei tarvitse eksplisiittisesti siirtää korkeampaan ulottuvuuteen, vaan riittää pelkästään korvata kaikki algoritmin pistetulot sopivalla funktiolla.
- Menetelmää voidaan näin ollen käyttää kaikilla menetelmillä, jotka perustuvat vektoreiden pistetuloon (kuten mm. LDA ja SVM).
- Ytimiä ja sitä kautta erilaisia kuvauksia on olemassa lukuisia.
- Yksinkertaisimmat ytimet ovat ns. polynomiytimiä, jotka muodostetaan nostamalla pistetulo johonkin kokonaislukupotenssiin.
- Esimerkiksi toisen asteen polynomiydin vastaa normaalin pistetulon korvaamista sen toisella potenssilla:

$$\mathrm{k}(\mathbf{x}_\mathrm{i}, \mathbf{x}_\mathrm{j}) = \left(\mathbf{x}_\mathrm{i} \cdot \mathbf{x}_\mathrm{j}\right)^2 .$$

Tukivektorikoneet

- Esimerkin tapauksessa data oli kaksiulotteista, joten vektoreiden $\mathbf{x}_i = (\mathbf{x}_i(1), \mathbf{x}_i(2))$ ja $\mathbf{x}_j = (\mathbf{x}_j(1), \mathbf{x}_j(2))$ pistetulo määritellään kaavalla

$$\mathbf{x}_i \cdot \mathbf{x}_j = \mathbf{x}_i(1)\mathbf{x}_j(1) + \mathbf{x}_i(2)\mathbf{x}_j(2).$$

- Pistetulon toinen potenssi on näin ollen lauseke

$$\begin{aligned}
k(\mathbf{x}_i, \mathbf{x}_j) &= \left[\mathbf{x}_i(1)\mathbf{x}_j(1) + \mathbf{x}_i(2)\mathbf{x}_j(2)\right]^2 \\
&= \left[\mathbf{x}_i(1)\mathbf{x}_j(1) + \mathbf{x}_i(2)\mathbf{x}_j(2)\right] \cdot \left[\mathbf{x}_i(1)\mathbf{x}_j(1) + \mathbf{x}_i(2)\mathbf{x}_j(2)\right] \\
&= \mathbf{x}_i^2(1)\mathbf{x}_j^2(1) + \mathbf{x}_i^2(2)\mathbf{x}_j^2(2) + 2\mathbf{x}_i(1)\mathbf{x}_j(1)\mathbf{x}_i(2)\mathbf{x}_j(2) \\
&= \mathbf{x}_i^2(1)\mathbf{x}_j^2(1) + \mathbf{x}_i^2(2)\mathbf{x}_j^2(2) + (\sqrt{2}\mathbf{x}_i(1)\mathbf{x}_i(2)) \cdot (\sqrt{2}\mathbf{x}_j(1)\mathbf{x}_j(2)) \\
&= (\mathbf{x}_i^2(1), \mathbf{x}_i^2(2), \sqrt{2}\mathbf{x}_i(1)\mathbf{x}_i(2))^T \cdot (\mathbf{x}_j^2(1), \mathbf{x}_j^2(2), \sqrt{2}\mathbf{x}_j(1)\mathbf{x}_j(2))^T.
\end{aligned}$$

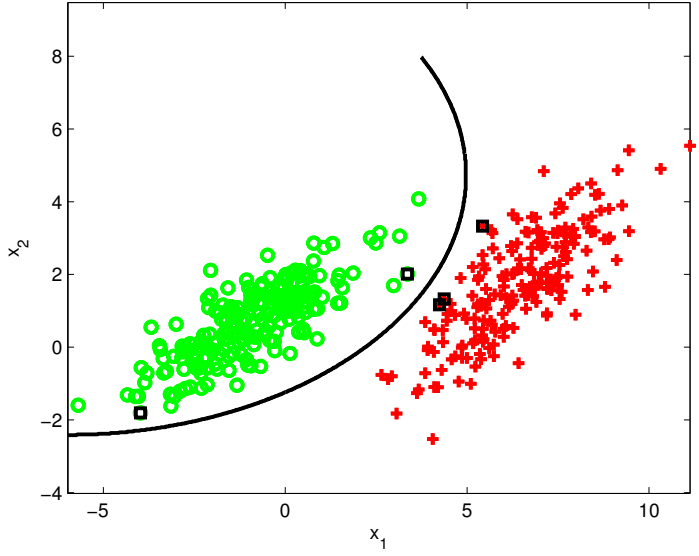
Tukivektorikoneet

- Kernelitempun idea on siis hoksata, että tämäkin lauseke on eräs pistetulo kolmiulotteisessa avaruudessa.
- Kuten viimeisestä lausekkeesta nähdään, niin samaan tulokseen päästään kuvaamalla alkuperäinen 2-ulotteinen vektori $\mathbf{x} = (\mathbf{x}(1), \mathbf{x}(2))$ kolmiulotteiseen avaruuteen kuvauksella

$$\begin{pmatrix} \mathbf{x}(1) \\ \mathbf{x}(2) \end{pmatrix} \mapsto \begin{pmatrix} \mathbf{x}(1)^2 \\ \mathbf{x}(2)^2 \\ \sqrt{2}\mathbf{x}(1)\mathbf{x}(2) \end{pmatrix}. \tag{9}$$

- Johtopäätöksenä voidaan siis todeta, että pistetulojen nostaminen toiseen potenssiin vastaa normaalia SVM:ää kolmiulotteisessa tapauksessa, jossa näytteet on saatu kaavalla (9).
- Alla oleva kuva esittää luokittelun tulosta.

Tukivektorikoneet



Tukivektorikoneet

- Edellisessä tapauksessa sama luokittelija olisi voitu toteuttaa ilman kernelitemppuakin, yksinkertaisesti suunnittelemalla kolmiulotteinen SVM-luokittelija kuvauksen (9) jälkeiselle datalle.
- Eksplisiittinen kuvaus korkeampaan dimensioon ei kuitenkaan ole mahdollista, jos dimensio on kovin suuri.
- Tällainen tilanne syntyy esimerkiksi käytettäessä epähomogeenista polynomiydintä

$$k(\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i \cdot \mathbf{x}_j + 1\right)^d,$$

joka tuo mukaan kaikki enintään astetta d olevat monomit.

Tukivektorikoneet

- Kuvaus (9) muuttuisi tässä tapauksessa muotoon

$$\begin{pmatrix} \mathbf{x}(1) \\ \mathbf{x}(2) \end{pmatrix} \mapsto \begin{pmatrix} \mathbf{x}(1)^2 \\ \mathbf{x}(2)^2 \\ \sqrt{2}\mathbf{x}(1)\mathbf{x}(2) \\ \sqrt{2}\mathbf{x}(1) \\ \sqrt{2}\mathbf{x}(2) \\ 1 \end{pmatrix}. \tag{10}$$

- Jos nyt esimerkiksi alkuperäinen data olisikin viisiulotteista ja esim. $\mathrm{d} = 8$, suoraan kuvaukseen tulisi kymmeniä termejä ja kernelitemppu alkaa vaikuttaa entistä houkuttelevammalta.

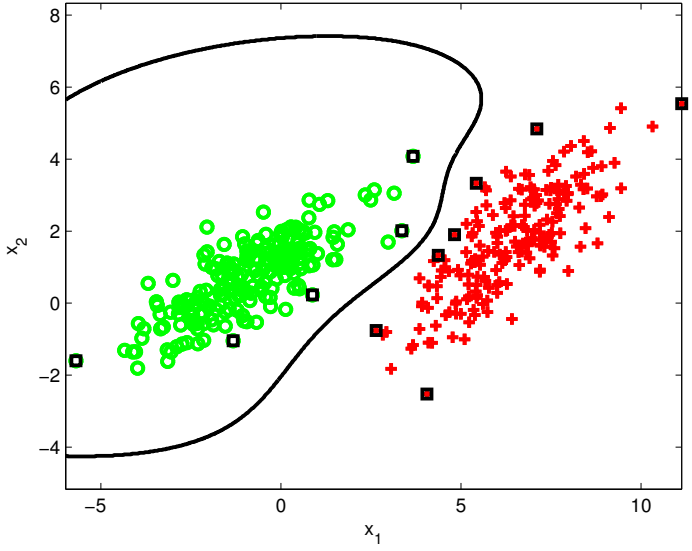
Tukivektorikoneet

- Äärimmäisenä esimerkkinä kernelitempun mahdollisuuksista on yleisesti käytetty ns. RBF-ydin (radial basis function), joka vastaa kuvausta **ääretönulotteiseen** avaruuteen.
- RBF-ydin korvaa vektoreiden pistetulon $\mathbf{x}_i \cdot \mathbf{x}_j$ funktiolla

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma ||\mathbf{x}_i - \mathbf{x}_j||^2),$$

  missä parametri $\gamma > 0$.
- Voidaan osoittaa, että tämä temppu vastaa lineaarisen päätöspinnan hakua ääretönulotteisessa avaruudessa.
- Alla olevassa kuvassa on esimerkki päätöspinnasta RBF-ydintä käytettäessä tapauksessa $\gamma = 1$.

Tukivektorikoneet



Tukivektorikoneet

- Vaikka edellä oletettiinkin, että luokat ovat eroteltavissa toisistaan lineaarisella päätöspinnalla (jotta marginaali olisi ylipäätään olemassa), ei käytännössä näin useinkaan ole.
- Tämän vuoksi SVM-algoritmi on yleistetty myös tilanteisiin joissa joukot menevät päällekkäin, ja kaikki järkevät toteutukset tukevat tätä.
- SVM on saavuttanut huomattavan suosion, mikä johtuu ensisijaisesti sen tavasta maksimoida luokittelumarginaali.
- Tämä nimittäin helpottaa opetusta ja saa luokittelijan toimimaan hyvin jo pienilläkin opetusjoukoilla.
- Muilla luokittelijoilla on nimittäin vaarana ns. **ylioppiminen**, josta on esimerkki hermoverkkojen yhteydessä.
- Yleisen käsityksen mukaan SVM ei ole yhtä herkkä opetusjoukon koolle kuin muut luokittelijat.

Logistinen Regressio

- Logistinen regressio on kolmas lineaarinen luokittelumenetelmä.
- Kaikilla kolmella on siis lineaarinen luokkaraja (eli esitettävissä muodossa $\mathbf{w}^{\mathrm{T}}\mathbf{x} > \mathrm{b}$), mutta niiden opetusalgoritmit (ja siis tuloksena saatavat luokkarajat) ovat erilaiset.
- Logistisen regression erikoispiirre on sen tilastollinen luonne, eli tuloksena saadaan itse luokan lisäksi myös kunkin luokan todennäköisyys.
- Logistiseen regressioon saadaan liitettyä myös piirteenvalintaominaisuuksia, joiden avulla luokittelu osaa valita ennustuksen kannalta parhaan osajoukon kaikista piirteistä.

Logistinen Regressio

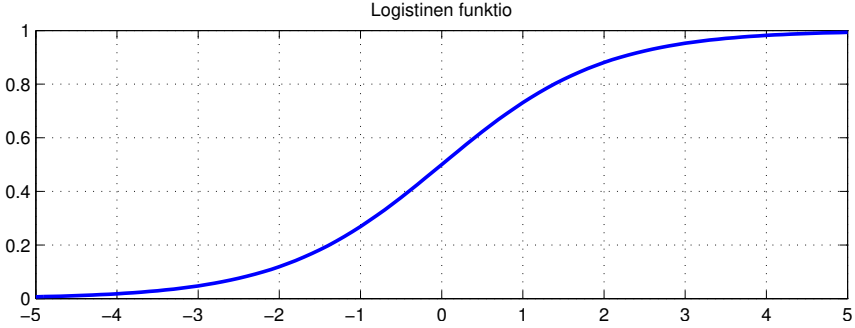
- Kahden luokan tapauksessa logistinen regressiomalli mallintaa toisen luokan todennäköisyyttä kaavalla:

$$f(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{x} - b)},$$

  missä $\mathbf{w}$ on aineistosta opittu kerroinvektori, $\mathbf{x}$ luokiteltava näyte, b reaaliluku.
- Tuloksena saadaan siis yhden luokan todennäköisyys, ja kahden luokan tapauksessa toisen luokan todennäköisyys on tämän komplementti $(1 - f(\mathbf{x}))$.
- Todennäköisyys saadaan siis pistetulosta $\mathbf{w}^T\mathbf{x}$ kuvaamalla se niin sanotun logistisen funktion $\sigma(x) = 1/(1 + \exp(-x))$ läpi.

Logistinen Regressio

- Funktion kuvaaja on alla, ja sen roolina voidaan ajatella olevan rajata ulostulo todennäköisyydeksi nollan ja yhden välille.



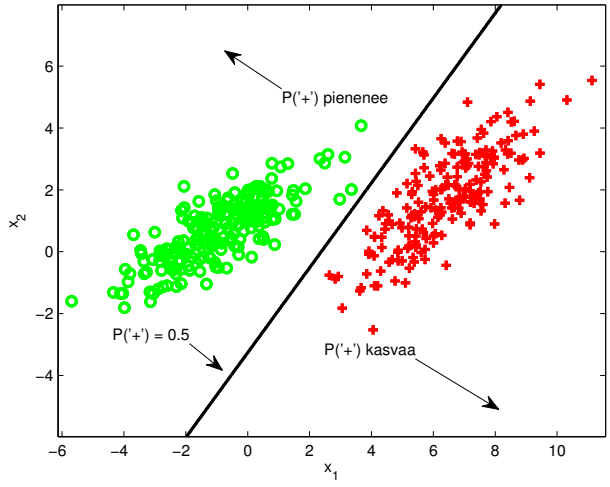
Logistinen Regressio

- Useamman kuin kahden luokan tapaus mallintaa luokan $c \in \{1, 2, \ldots, C\}$ todennäköisyyttä kaavalla

$$P(c \mid \mathbf{x}) = \frac{\exp(\mathbf{w}_c^T \mathbf{x} + b_c)}{\sum_{k=1}^{C} \exp(\mathbf{w}_k^T \mathbf{x} + b_k)},$$

  missä mallin parametrit $\mathbf{w}_k$ and b_k $(k = 1, 2, \ldots, C)$ opitaan opetusaineistosta.
- Luokittelutulos aiemmin esillä olleelle testidatalle on kuvattu alla.

Logistinen Regressio



Logistinen Regressio

- Kuten nähdään luokkaraja on hyvin samanlainen kuin kahdella aiemmalla lineaarisella luokittelijalla.
- Erona on nyt se, että jokaiselle pisteelle arvioidaan nyt myös todennäköisyys kuulua punaisiin plussiin. Luokkaraja sijaitsee kohdassa, jossa molemmat luokat ovat yhtä todennäköisiä (eli $\mathrm{P}(\prime+\prime) = \mathrm{P}(\prime\circ\prime) = 0.5$).