

# Kurkistus tilastollisesti oppivien koneiden eläintarhaan

Studia Varjomafia 4, Terrakoti

Aaro Salosensaari [aaro.salosensaari@utu.fi](mailto:aaro.salosensaari@utu.fi). CC-BY-SA 4.0, paitsi  
fair use.

- 1 Eläintarhan portti
- 2 Ensimmäinen eläin: Logistinen regressio
- 3 Toinen eläin: Yksinkertainen neuroverkko
- 4 Kuinka eläimet selviytyvät luonnossa (MNIST-numeroiden luokittelussa)

# Mikä on tilastollisesti oppiva kone?



# Tilastollisesti oppiva kone on ... ?

Eräs epätarkasti yksinkertaistettu määritelmä:

## Tilastollinen koneoppimisalgoritmi

Algoritmi, joka oppii aineistosta ("data") antamaan vastauksia jotka todennäköisesti (= tilastollisesti keskimäärin) ovat oikein.

# Tilastollisesti oppiva kone on ... ?

Eräs epätarkasti yksinkertaistettu määritelmä:

## Tilastollinen koneoppimisalgoritmi

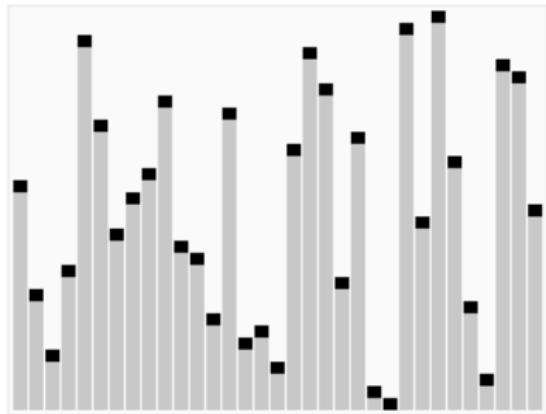
Algoritmi, joka oppii aineistosta ("data") antamaan vastauksia jotka todennäköisesti (= tilastollisesti keskimäärin) ovat oikein.

Seuraava kysymys: Mitä tarkoittaa oppiminen?

# Algoritmi joka ei opi

```
algorithm quicksort(A, lo, hi) is
    if lo < hi then
        p := partition(A, lo, hi)
        quicksort(A, lo, p - 1)
        quicksort(A, p + 1, hi)

algorithm partition(A, lo, hi) is
    pivot := A[hi]
    i := lo
    for j := lo to hi do
        if A[j] < pivot then
            swap A[i] with A[j]
            i := i + 1
    swap A[i] with A[hi]
    return i
```



Quicksort-algoritmi järjestää lukuja askel askeleelta, ja antaa todistettavasti oikean ratkaisun jokaiselle lukujonolle.

Kuvalähde: Wikipedia  
User:RolandH CC-BY-SA 3.0 [1]

# Algoritmit jotka oppivat

Algoritmi (ihmisen koodaama ohjelmakoodi) ei ratkaise ongelmaa itsessään (vrt. edel. kalvon quicksort).

# Algoritmit jotka oppivat

Algoritmi (ihmisen koodaama ohjelmakoodi) ei ratkaise ongelmaa itsessään (vrt. edel. kalvon quicksort).

Sen sijaan ns. oppiva algoritmi koostuu usein oikeastaan kahdesta osasta:

# Algoritmit jotka oppivat

Algoritmi (ihmisen koodaama ohjelmakoodi) ei ratkaise ongelmaa itsessään (vrt. edel. kalvon quicksort).

Sen sijaan ns. oppiva algoritmi koostuu usein oikeastaan kahdesta osasta:

- ① Malli (jossa on parametreja).
- ② Algoritmi, joka kertoo kuinka valitaan eli **opitaan** parametrit joilla malli antaa (tilastollisesti keskimäärin) ratkaisuja johonkin tiettyyn ongelmaan.

# Algoritmit jotka oppivat

Algoritmi (ihmisen koodaama ohjelmakoodi) ei ratkaise ongelmaa itsessään (vrt. edel. kalvon quicksort).

Sen sijaan ns. oppiva algoritmi koostuu usein oikeastaan kahdesta osasta:

- ① Malli (jossa on parametreja).
- ② Algoritmi, joka kertoo kuinka valitaan eli **opitaan** parametrit joilla malli antaa (tilastollisesti keskimäärin) ratkaisuja johonkin tiettyyn ongelmaan.



vs.



# Mitä tarkoittaa malli?

## Parametrit?

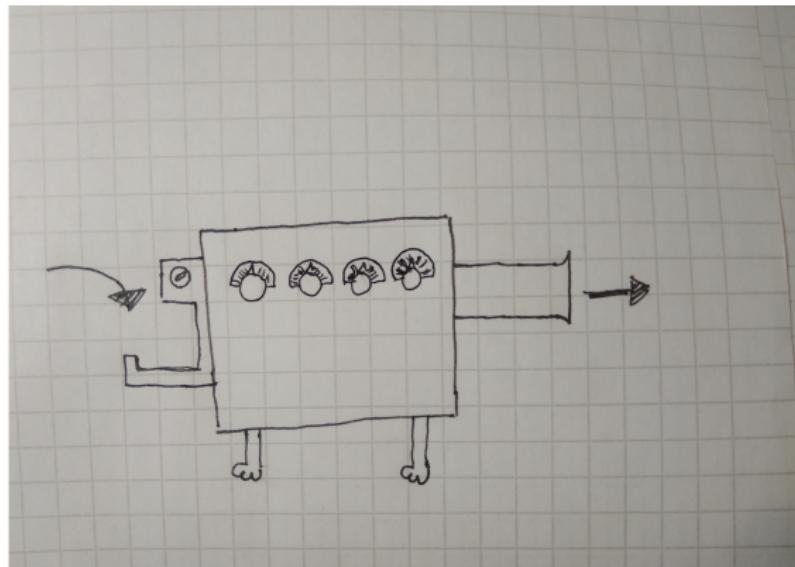
Apua!?



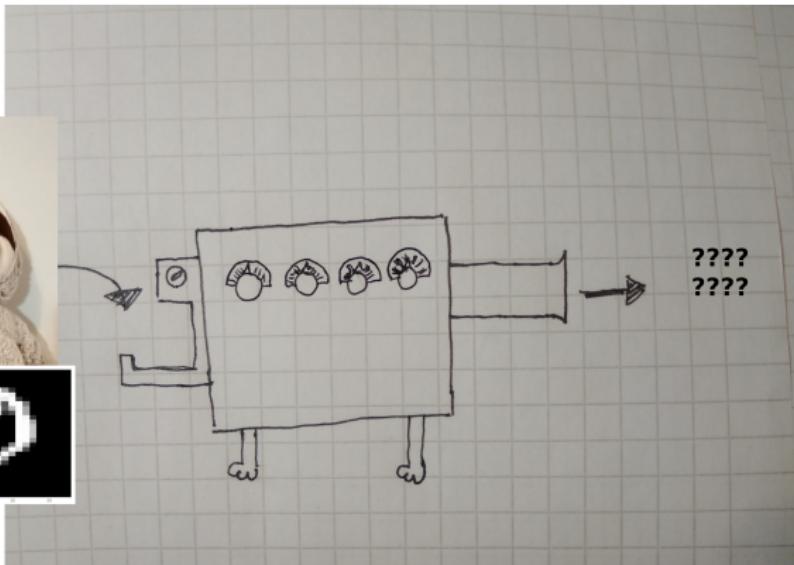
# Mitä tarkoittaa malli?

## Parametrit?

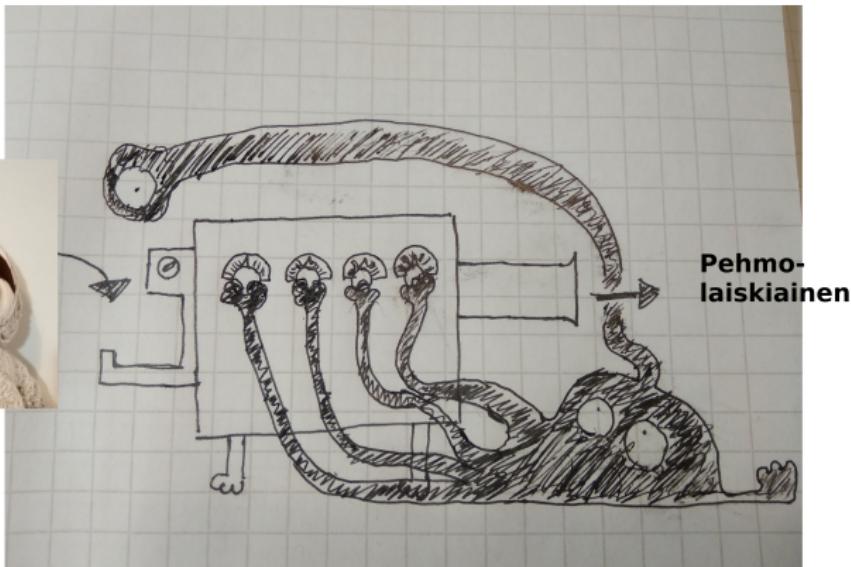
Apua!?



# Mallit ovat yleisiä



# Mitä tarkoittaa parametrien oppiminen: optimointialgoritmi



# Mitä tarkoittaa "tilastollisesti keskimäärin oikea vastaus"?

Tähän on tarkka matemaattinen vastaus → ei sillä väliä.

# Mitä tarkoittaa "tilastollisesti keskimäärin oikea vastaus"?

Tähän on tarkka matemaattinen vastaus → ei sillä väliä.

Eräs käytännöllinen esimerkki:

- 1 Valitaan sääntö jolla mitata virheellisen vastauksen virheellisyyttä numerolla.

# Mitä tarkoittaa "tilastollisesti keskimäärin oikea vastaus"?

Tähän on tarkka matemaattinen vastaus → ei sillä väliä.

Eräs käytännöllinen esimerkki:

- ① Valitaan sääntö jolla mitata virheellisen vastauksen virheellisyyttä numerolla.
- ② Jätetään sivuun osa datasta jota ei käytetty oppimiseen.  
(Testijoukko.)

# Mitä tarkoittaa "tilastollisesti keskimäärin oikea vastaus"?

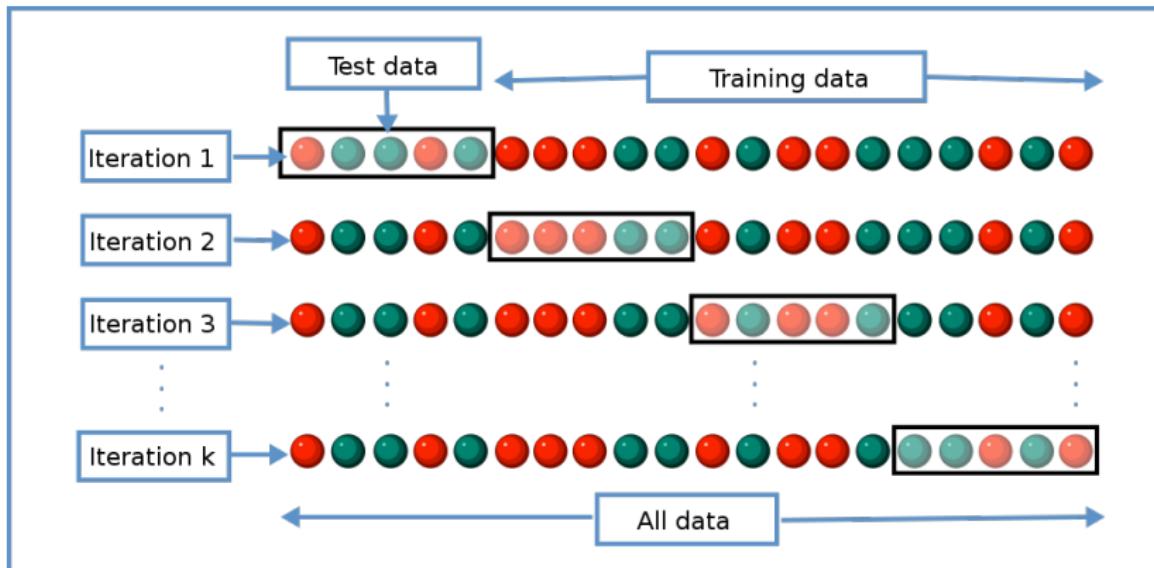
Tähän on tarkka matemaattinen vastaus → ei sillä väliä.

Eräs käytännöllinen esimerkki:

- ① Valitaan sääntö jolla mitata virheellisen vastauksen virheellisyyttä numerolla.
- ② Jätetään sivuun osa datasta jota ei käytetty oppimiseen.  
(Testijoukko.)
- ③ Mitataan opetetun koneen vastausten virhe testijoukossa.

# Tilastollisesti keskimäärin oikea vastaus: Ristiinvaliointi

Pätkitään data palasiin monta kertaa ja mitataan virhe monta kertaa!



# Tiivistelmä johdannosta:

Tilastollisesti oppiva kone:

# Tiivistelmä johdannosta:

Tilastollisesti oppiva kone:

- ① Algoritmi joka ei opи (esimerkki quicksort): olemme kirjoittaneet sen niin etт se antaa taatusti oikeita vastauksia.
- ② Algoritmi joka oppii (esimerkiksi tm esitelm:) emme osaa / jaksa kirjoittaa algoritmia joka antaisi taatusti oikeita vastauksia. Sen sijaan valitsemme yleisen mallin, jonka parametrit voimme stt (opettaa) niin etт se antaa keskimarin oikeita vastauksia
- ③ Keskimarin oikean vastauksen arvointi:

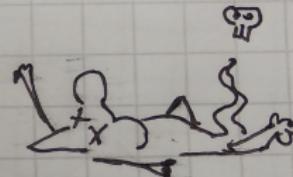
- 1 Eläintarhan portti
- 2 Ensimmäinen eläin: Logistinen regressio
- 3 Toinen eläin: Yksinkertainen neuroverkko
- 4 Kuinka eläimet selviytyvät luonnossa (MNIST-numeroiden luokittelussa)

# Eläinosion idea

MALLI



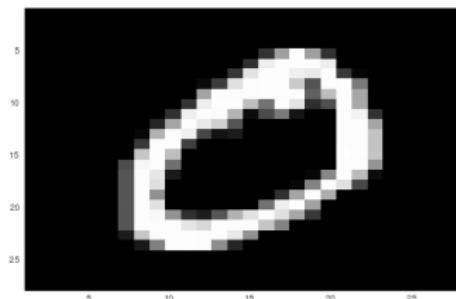
SEN TUOTOS



# Mistä eläinosiossa ei puhuta



# Eläinosion esimerkkidata: MNIST

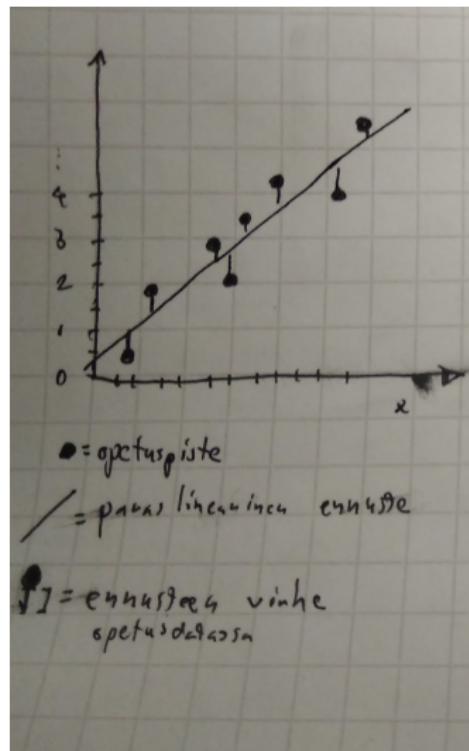


60000 käsinkirjoitettua numeroa jotka käsitelty kivasti.

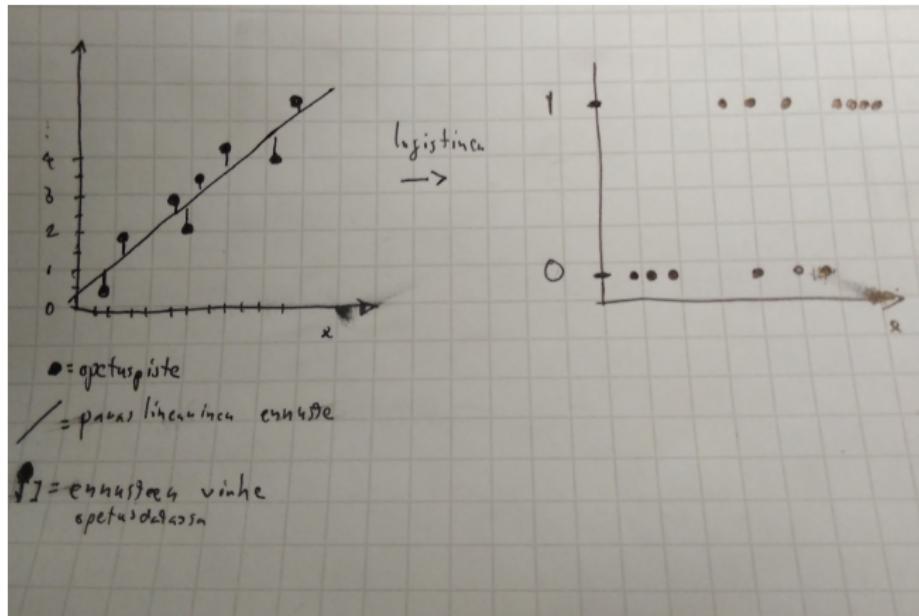
# Logistinen regressio: Visuaalinen johdanto



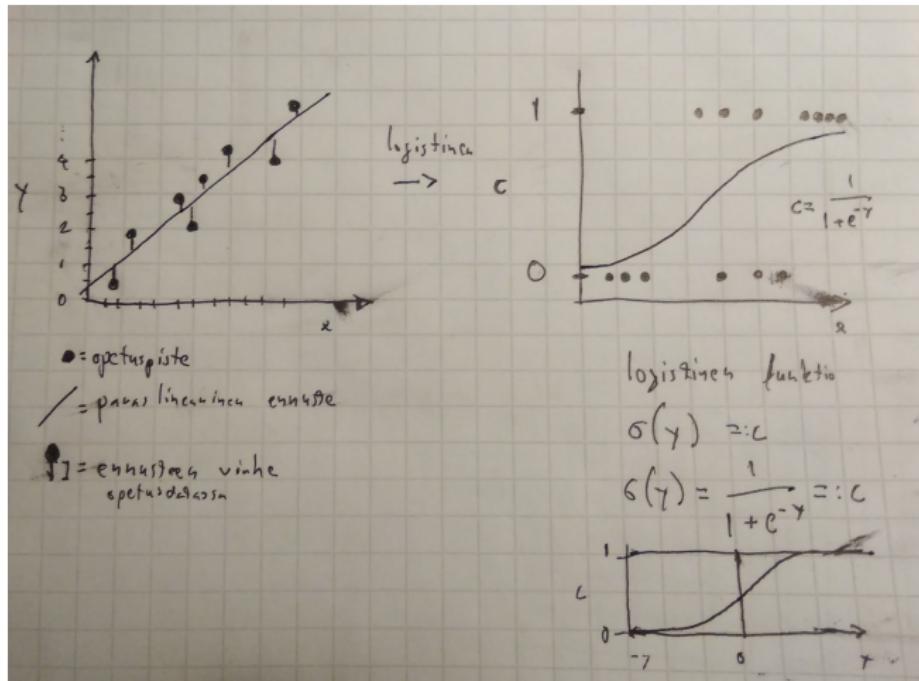
# Lineaarin regressio yhdellä kuvalla



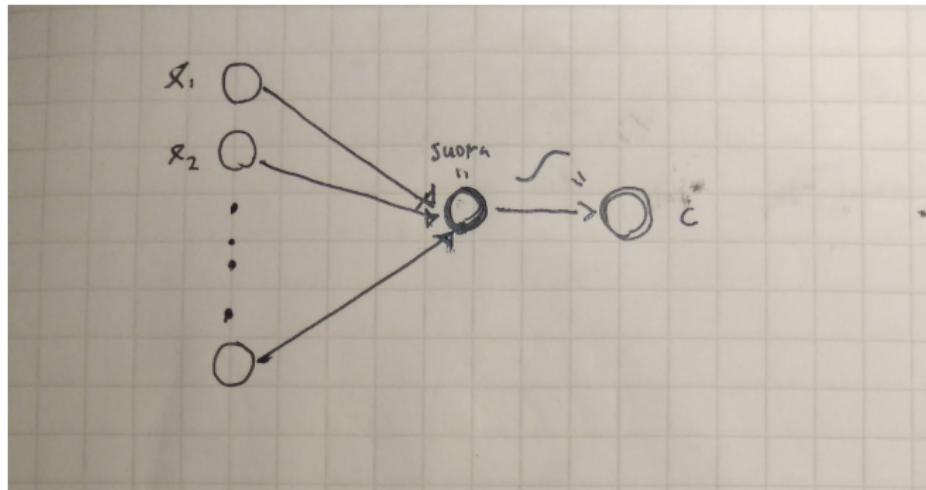
# Logistinen regressio kahdella kuvalla: 1



# Logistinen regressio kahdella kuvalla: 2



# Huijasin: Kolmas kuva



- 1 Eläintarhan portti
- 2 Ensimmäinen eläin: Logistinen regressio
- 3 Toinen eläin: Yksinkertainen neuroverkko
- 4 Kuinka eläimet selviytyvät luonnossa (MNIST-numeroiden luokittelussa)

Neuroverkot **hyvin** ytimekkäästi



# Pari sanaa neuroverkoista ja oikeista hermosoluista



- Ihmisten, nisäkkäiden, muiden **hermoverkot**: yksittäiset hermosolut ovat monimutkaisia koneita.

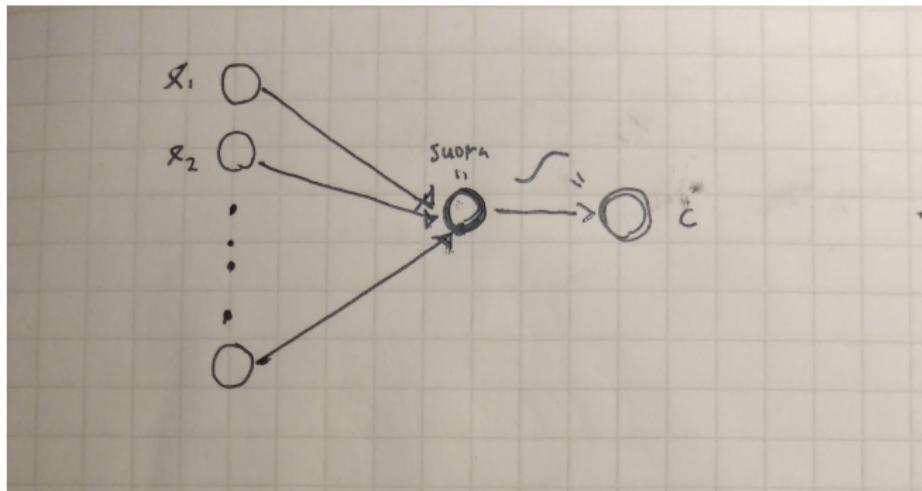
# Pari sanaa neuroverkoista ja oikeista hermosoluista

- Ihmisten, nisäkkäiden, muiden **hermoverkot**: yksittäiset hermosolut ovat monimutkaisia koneita.
- Koneoppimisen neuroverkot: koostuvat hyvin yksinkertaisista "neuroneista", muutama matemaattinen operaatio.

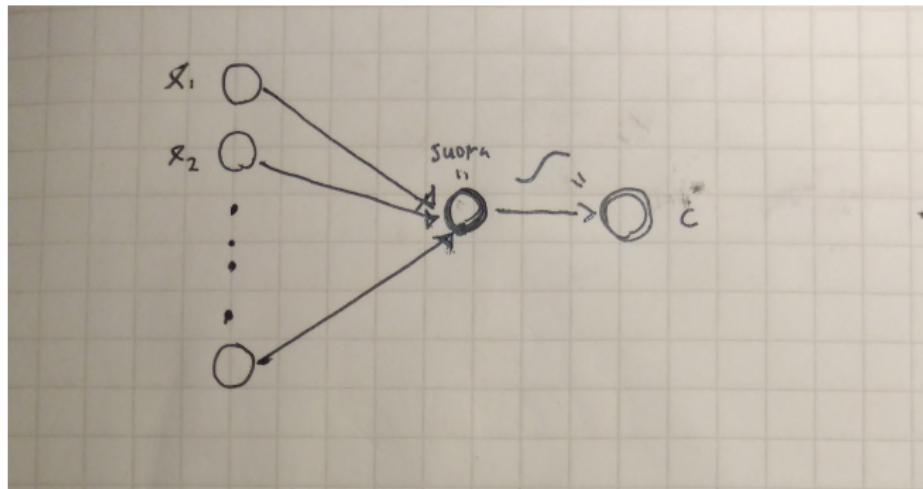
# Pari sanaa neuroverkoista ja oikeista hermosoluista

- Ihmisten, nisäkkäiden, muiden **hermoverkot**: yksittäiset hermosolut ovat monimutkaisia koneita.
- Koneoppimisen neuroverkot: koostuvat hyvin yksinkertaisista "neuroneista", muutama matemaattinen operaatio.
- Yhteisyys: Yksinkertaiset palaset voivat tehdä monimutkaista laskentaa kun ne verkotetaan oikein.

# Yksinkertaisin neuroverkko

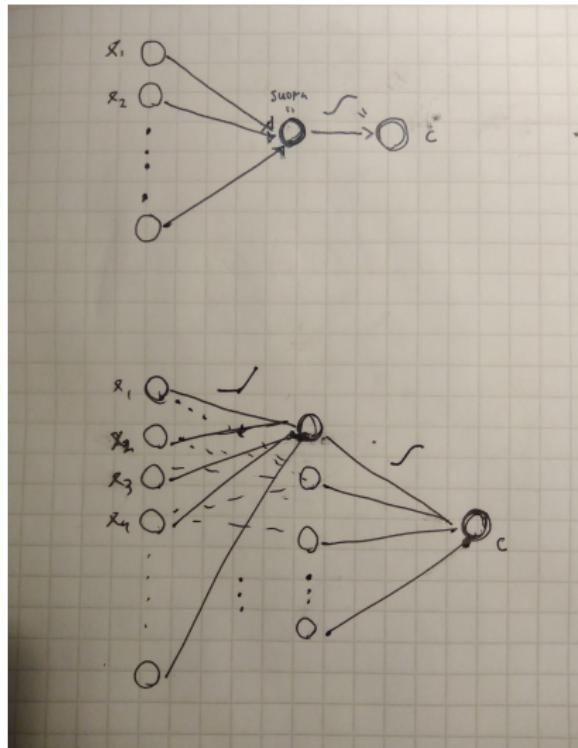


# Yksinkertaisin neuroverkko



Logistinen regressio on neuroverkko! (Perceptron.)

# Hieman monimutkaisempi neuroverkko (MLP)



- 1 Eläintarhan portti
- 2 Ensimmäinen eläin: Logistinen regressio
- 3 Toinen eläin: Yksinkertainen neuroverkko
- 4 Kuinka eläimet selviytyvät luonnossa (MNIST-numeroiden luokittelussa)

# Miltä eläimemme näyttää Julia-kielellä

```
79 for i in 1:5
80     Xtmp = X[:, folds[i][1]]
81     Ytmp = Y[:, folds[i][1]]
82     tX = X[:, folds[i][2]]
83     tY = Y[:, folds[i][2]]
84     dataset = repeated((Xtmp, Ytmp), 80)
85
86     # logistinen regressio-eläin
87     m_logres = Chain(
88         Dense(28^2, 10),
89         softmax
90     )
91
92     loss_logres(x, y) = crossentropy(m_logres(x),y)
93
94     accuracy_logres(x,y) = mean(onecold(m_logres(x)) .== one
95
96     evalcb_logres = () -> @show(loss_logres(Xtmp, Ytmp))
97
98     opt_logres = Descent(0.5)
99
100    Flux.train!((x, y) -> loss(x,y,m_logres),
101                params(m_logres),
102                dataset,
103                opt_logres,
104                cb = throttle(evalcb_logres, 10)
105            )
```

```
for i in 1:5
    i=1
    Xtmp = X[:, getobs(folds,i)[1]]
    Ytmp = Y[:, getobs(folds,i)[1]]
    tX = X[:, getobs(folds,i)[2]]
    tY = Y[:, getobs(folds,i)[2]]
    dataset = repeated((Xtmp, Ytmp), 80)

    # eläin:
    m_mlp = Chain(
        Dense(28^2, 32, relu),
        Dense(32, 10),
        softmax
    )
    # eläin päättyy

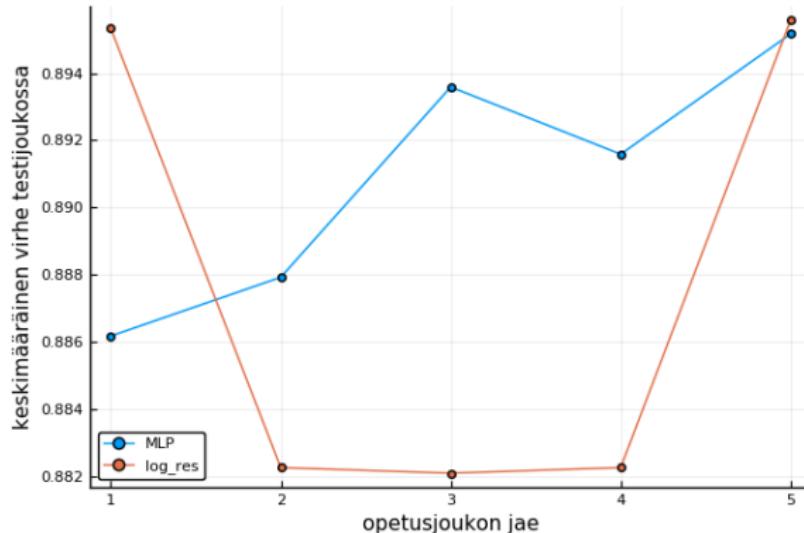
    loss_mlp(x, y) = crossentropy(m_mlp(x),y)

    accuracy_mlp(x,y) = mean(onecold(m_mlp(x)) .== onecold(y))

    evalcb_mlp = () -> @show(loss_mlp(Xtmp, Ytmp))
    opt = ADAM()

    Flux.train!(loss_mlp, params(m_mlp), dataset, opt, cb = t
    accuracy_mlp(tX, tY)
```

# Kuinka eläimemme suoriutuvat numeroiden luokittelussa



# Miltä näyttää MLP-eläimen luokittelukyky



```
julia> tY_true[:, 1]
10-element Flux.OneHotVector:
 0
 0
 0
 0
 0
 0
 0
 1
 0
 0
```

```
julia> m_mlp(tX_true[:,1])
10-element Array{Float32,1}:
 0.0006360026
 5.829799e-5
 0.0005085462
 0.004486961
 0.0012833853
 0.000531616
 0.00016474871
 0.9821462
 0.0019820372
 0.008262223
```

# Miltä näyttää MLP-eläimen luokittelukyky

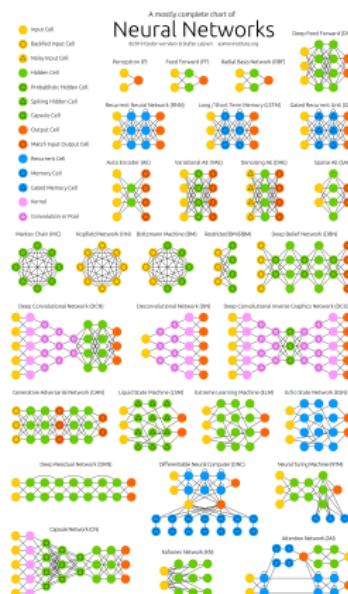


```
julia> tY_true[:, 1]
10-element Flux.OneHotVector:
 0
 0
 0
 0
 0
 0
 0
 1
 0
 0
```

```
julia> m_mlp(tX_true[:,1])
10-element Array{Float32,1}:
 0.0006360026
 5.829799e-5
 0.0005085462
 0.004486961
 0.0012833853
 0.000531616
 0.00016474871
 0.9821462
 0.0019820372
 0.008262223
```

```
julia> accuracy_mlp(tX_true[:,1], tY_true[:,1])
1.0
```

# Lopuksi: Neuroverkkoarkkitehtuureja



Arkkitehtuureja on valtavasti ja kaikki tekevät erilaisia asioita hyvin, joskus. Lähde: Fjodor Van Veen, [3]

# Viitteitä, linkkejä ja kirjallisuutta



[1]

[https://commons.wikimedia.org/wiki/File:Sorting\\_quicksort\\_anim.gif](https://commons.wikimedia.org/wiki/File:Sorting_quicksort_anim.gif)

[2]

[https://en.wikipedia.org/wiki/File:K-fold\\_cross-validation\\_EN.svg](https://en.wikipedia.org/wiki/File:K-fold_cross-validation_EN.svg)

[3] <https://www.asimovinstitute.org/neural-network-zoo/>