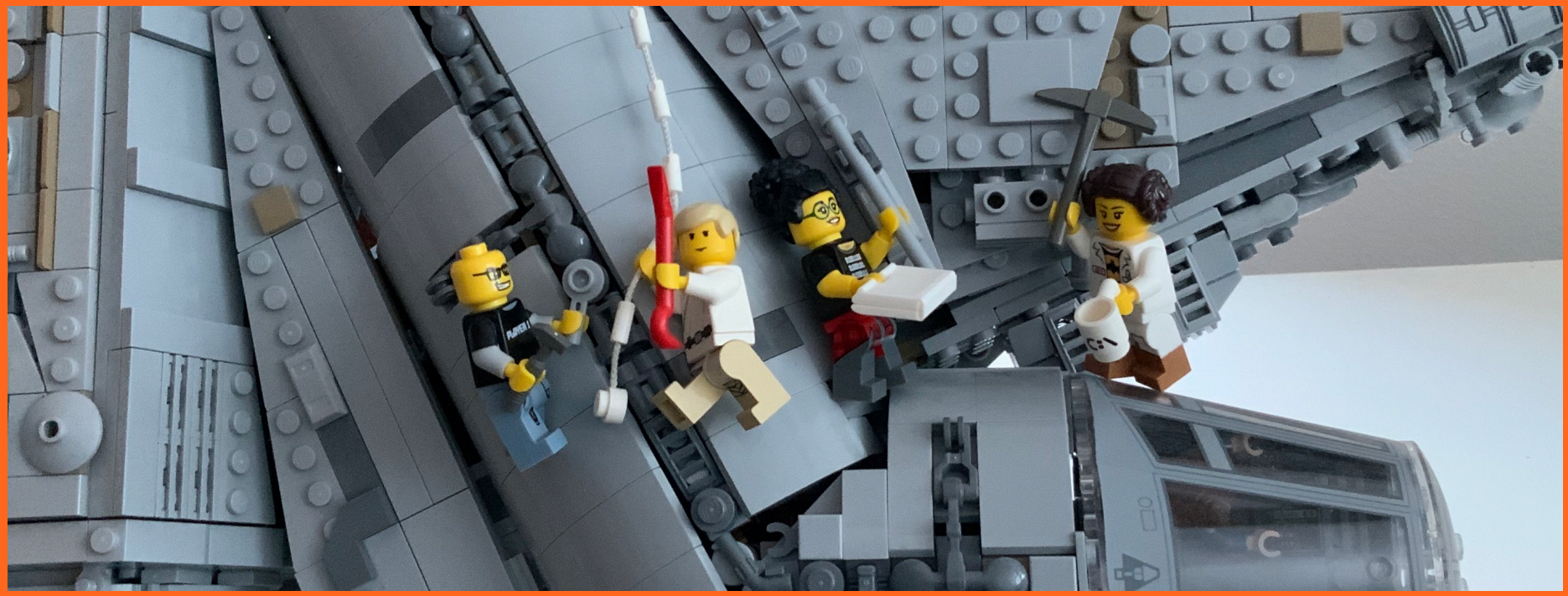
# CS-A1120 Ohjelmointi 2



Aalto-yliopisto
Perustieteiden
korkeakoulu

## Luento 10
## kevät 2026

**Sanna Suoranta (suomeksi, T1 klo 14) ja**

**Vesa Hirvisalo / Lukas Ahrenberg (englanniksi, T1 klo 10)**

**11.5.2026 https://presemo.aalto.fi/o2fi2026**

# Tänään O2:ssa: oppivat koneet

https://presemo.aalto.fi/o2fi2026


Aalto-yliopisto
Perustieteiden
korkeakoulu

# Tämä on vain lyhyt esittely

- **Tällä kurssilla raapaistaan vain vähän pintaa**
- **Aallossa on kokonaisia kursseja tästä:**
  - CS-C3240 Machine Learning
  - CS-E4800 Artificial Intelligence
  - …
- **… aina maisteristasoiseen pääaineeseen:**
  - Machine Learning, Data Science and Artificial Intelligence (Macadamia)
  - https://www.aalto.fi/en/programmes/masters-programme-in-computer-communication-and-information-sciences/curriculum-2024-2026

# Laskenta (computing)

**Alhaalta–ylös-näkökulman mukaisesti tällä kurssilla:**

- **Fysikaalinen ilmiö**
- **Toteuttaa sekvenssilogiikkaa**
- **Tietokoneen arkkitehtuuri (ALU, muisti, konekieli)**
- **Ohjelmat**
- **Abstraktiot, jotka mahdollistavat ohjelmoijalle enemmän tekemisen vähemmässä ajassa**

- **Nyt sitten miten koneet oppii**

# Abstraktiot

- **Ohjelmointi on kovaa työtä.**
- **Ohjelmoijat ovat ‘laiskoja’ ja haluavat koneen tekevän suurimman osan työstä**

- **Mitä jos opettaisimme koneen ohjelmoimaan?**

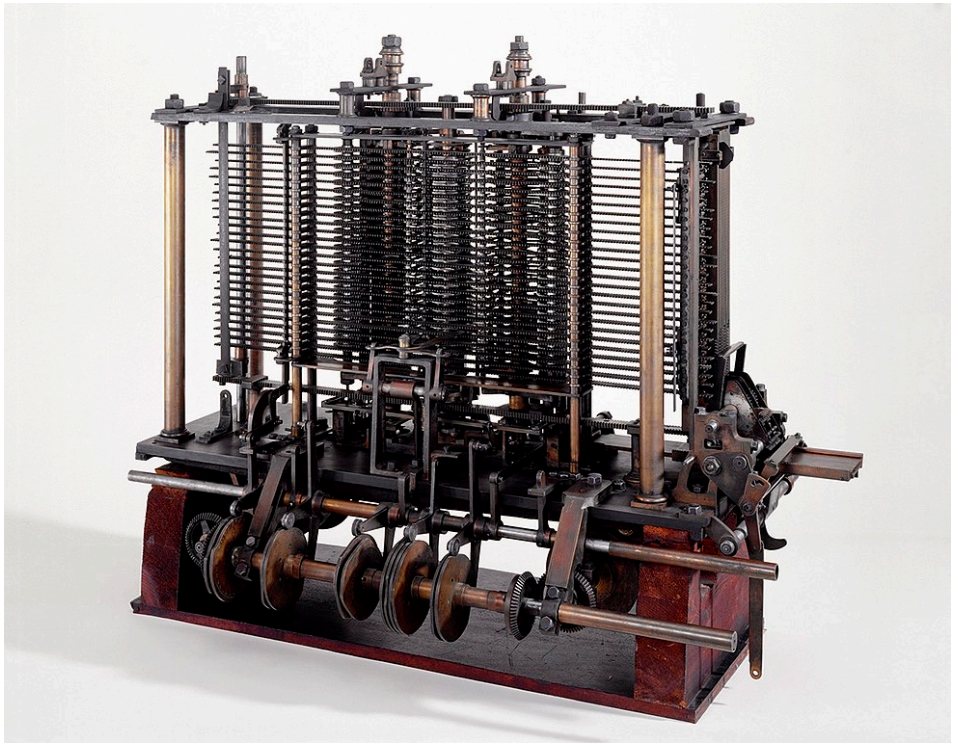
# Kun ratkomme ongelmaa ohjelmoimalla,

https://presemo.aalto.fi/o2fi2026

- **Mitä ohjelmoija tekee?**

Ada Lovelace, ensimmäinen ohjelmoija

- **Mitä (tieto)kone tekee?**



Charles Babbagen analyyttinen kone (osa)



https://en.wikipedia.org/wiki/Ada_Lovelace
https://en.wikipedia.org/wiki/Analytical_engine
Kuvat: Wikimedia

# Ohjelmoija ja (tieto)kone

**Ohjelmoija**

- **Kehittelee parhaan tavan ratkaista ongelma (”algoritmi”)**
- **Toteuttaa algoritmin tietokoneohjelmana jollakin kielellä**
- **Testaa ohjelman jollain syötteillä**
- **Mittaa suorituskyvyn**

**(Tieto)kone**

- **Suorittaa ohjelman**

# Opettaja ja oppija

## Opettaja

- Valitsee aiheet
- Valmistelee materiaalin
- Valmistelee esimerkit
- Vastaanottaa palautetta (feedback) oppijalta
- Antaa palautetta oppijalle

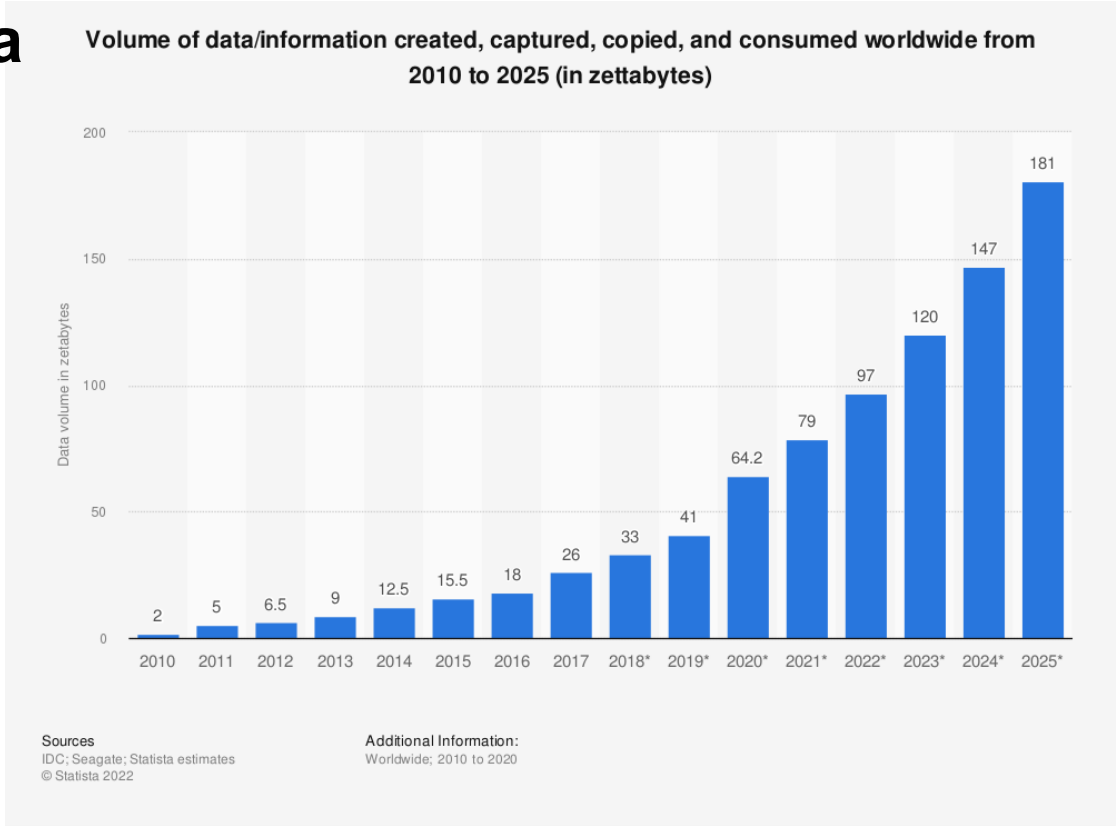
- (Arvostelee opiskelijan suorituksen tulokset)

## Oppija

- Tutkii materiaalin
- Työskentelee materiaalin (esim. tehtävien) kanssa
  - Näkee vaivaa (effort) oppiakseen
- Yrittää käyttää uusia taitojaan (skill)
- Antaa palautetta opettajalle
- Saa palautetta opettajalta

# Dataa on kaikkialla nykyään

- ⇒ tarjolla on paljon materiaalia ja esimerkkejä (jostakin)
- Voisiko ohjelman kirjoittaa niin, että se oppii (learn) datasta?
- Voisiko sen saada tuottamaan lisää ’samanlaista’ data?
- Kyllä – tämä on koneoppimisen (machine learning) perusta



# Koneoppimisen tekniikoita

- **Algoritmi / matemaattinen malli, joka pystyy yleistämään datasta (generalise from data) tai tekemään päätöksiä sen perusteella**
- **Periaate: datalla on rakenne, joka tekee siitä ennustettavan**
- **Useita eri tekniikoita**
  - Ohjattu oppiminen (supervised training) merkityllä datalla (labeled data) tai etsittyjen säännöllisyyksien esimerkkejä käyttäen
  - Ohjaamaton oppiminen (unsupervised training) datalla sellaisenaan; tavoite on tunnistaa ja oppia hyödyllisiä piirteitä
  - Vahvistusoppiminen (reinforcement training) annetussa ‘ympäristössä’ (sis esim, säännöt), jota malli tutkii ja kokeilee

# Koulutus (training)

- **Koulutus: kaappaa rakenteen säätämällä mallin parametreja**
- **Useita (osin erottamattomia) lähestymistapoja:**
  - Datan prosessointi niin, että uudet arvot voidaan interpoloida tai etsiä (esim. Polynomin sovittaminen)
  - Datan ryhmittely sen selvittämiseksi, mitkä datapisteet ovat lähekkäin (esim. k-means –menetelmä)
  - Arvioidaan todennäköisyys nähdä tietty lopputulos, kun jokin syöte on annettu (erotteleva malli, discriminative model)
  - Arvioidaan todennäköisyys nähdä tietty syöte ja lopputulos yhdessä (generatiivinen malli, generative model)
  - Arvioidaan neuroverkon (neural network) painokertoimet
  - …

# Koulutus (training) jatkuu

- **Koulutusvaihe voi myös käsitellä dataa,**
  - esim piirteiden erotus (feature extraction)
- **Huolia:**
  - Ylisovitus (overfitting): enemmän parametreja kuin dataa, havaitsee kohinaa rakenteen sijaan
  - Alisovitus (underfitting): liian vähän parametreja, ei havaitse rakennetta tarpeeksi hyvin
  - Vinoutumat (bias): koulutusdata ei ole kyllin edustava otos ’tuotantodatasta’
- **Validointia (validation) käytetään valvontaan**

# Piirteet (features) ja datan prosessointi

- **Piirre (feature) on diskreetti, mitattava ominaisuus jossain datasetissä**
  - Esim. Pikselit tai reunat kuvassa
- **Piirteet ovat ideaalitapauksessa toisistaan riippumattomia**
- **ulottuen koko piirreavaruuteen**
- **Koosta (size) ja tarkkuudesta (fidelity) johtuen nykykään datan piirteiden valinta on erittäin tärkeä vaihe**
  - (jos käytät jokaista pikseliä HD-kuvassa, piirreavaraisuudella on miljoonia ulottuvuuksia)
- **Piirre-erotus (feature exraction) vähäulotteisen piirrevekotorin (feature vector) valitsemiseksi voi olla tärkeä askel oppimisessa**

# Validointi (validation)

- **Mikä on mallin tekemän yleistyksen (generalisation) laatu?**
- **Malli pitää validoida harjoitusdatan ulkopuolisella materiaalilla**
- **Perusstrategia:**
  - Satunnaisesti jaetaan data kahteen erilliseen joukkoon:
  - Koulutusdata (training data)
  - Testidata (test data)
- **Validointi ei ole helposti ratkaistavissa sovelluksissa:**
  - Vinoumat
  - Vihamielinen syöte (adversarial input) eli hyökkäykset syötteen/mallin koulutuksen kautta

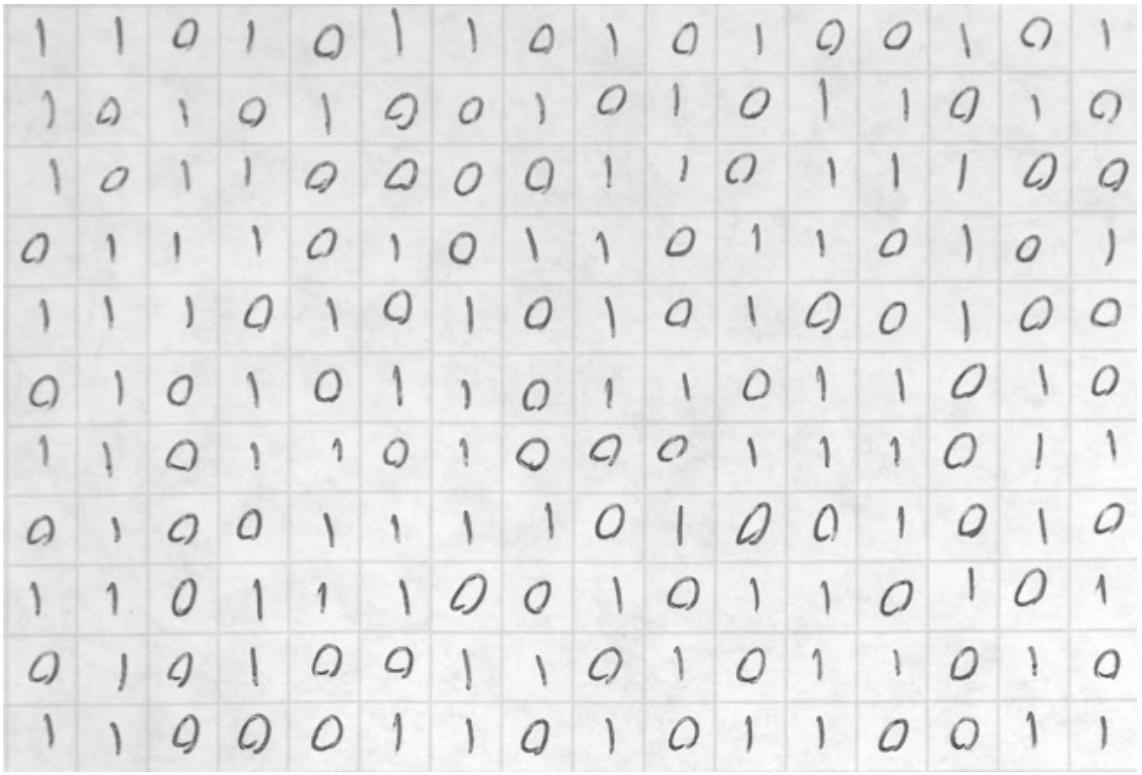
# Esimerkki: 0 vai 1?

https://presemo.aalto.fi/o2fi2026

Aalto-yliopisto
Perustieteiden
korkeakoulu

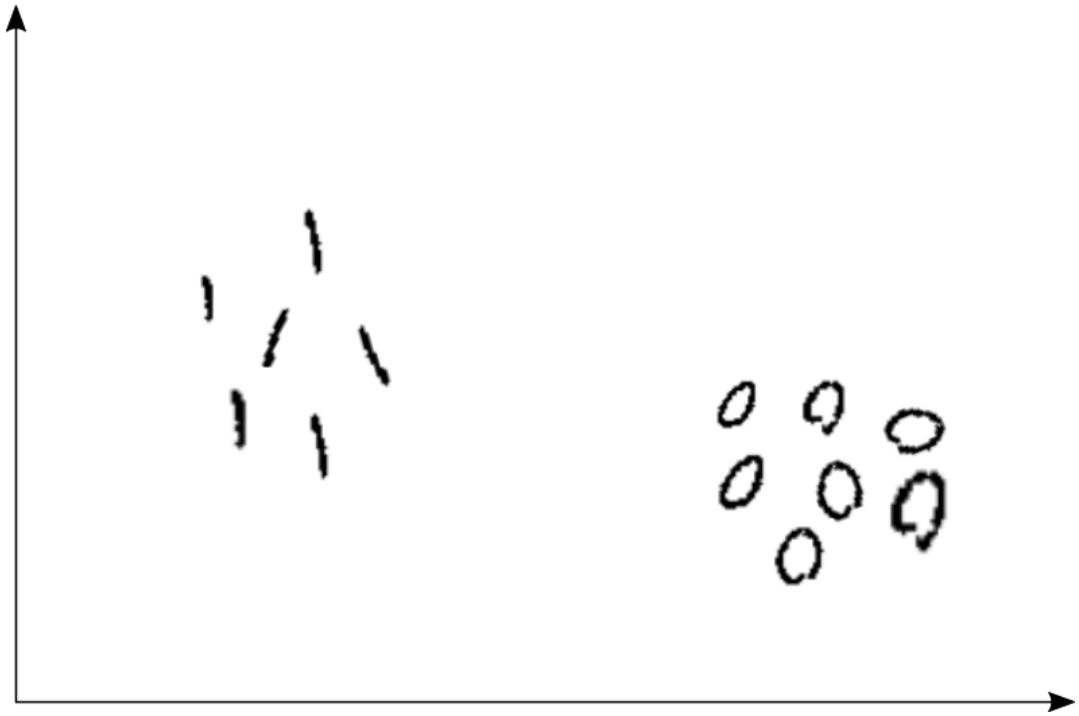
# Esimerkki – käsin kirjoitetut ykköset ja nollat

- **Luokitteluongelma (classification problem):**
  - Annettuna 160 × 160-taulukko 32-bittisiä intensiteettiarvoja (tarkoittaa $160^2 \times 2^{32}$ mahdollista diskreettiä arvoa)
  - Kuvaa nämä nimikkeille joukossa {0, 1}



# Ykköset ja nollat – perusidea

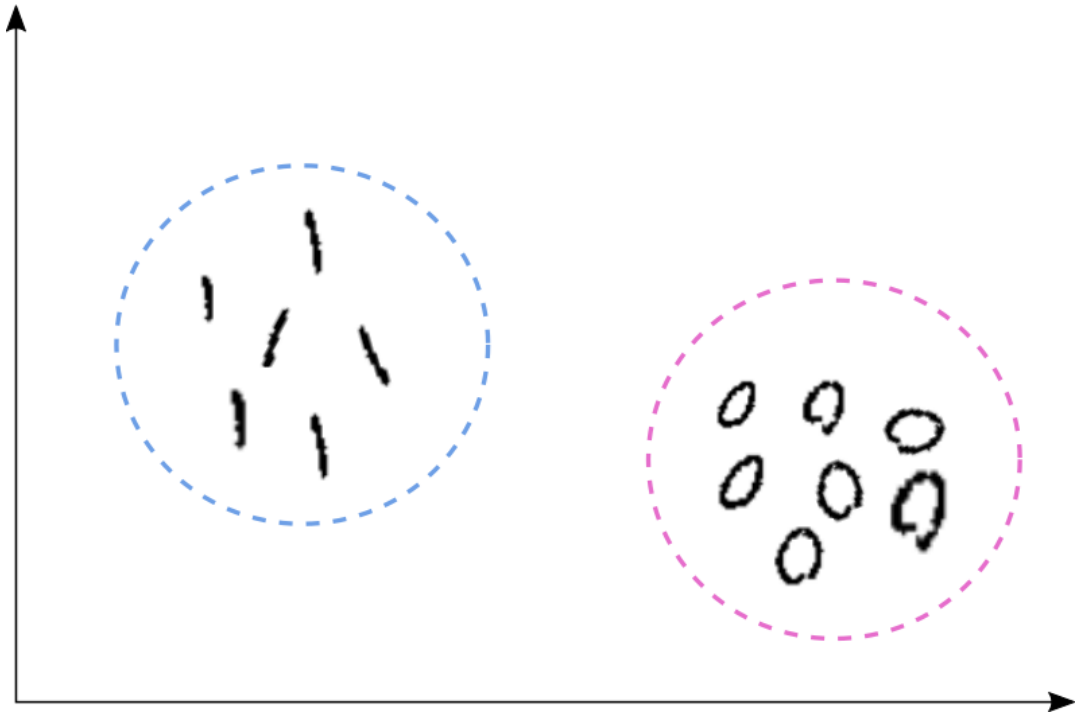
- **Yleistä annetusta koulutusdatasta ennestään tuntemattomiin tapauksiin tarkistamalla, mitkä piirteet uudesta tapauksessa ovat (jollain tavalla) lähimpänä.**



Huom. Piirreavaruus on oikeasti useampiulotteinen kuin tässä piirretty kaksiulotteinen

# Ykköset ja nollat – perusidea

- **Yleistä annetusta koulutusdatasta ennestään tuntemattomiin tapauksiin tarkistamalla, mitkä piirteet uudesta tapauksessa ovat (jollain tavalla) lähimpänä.**

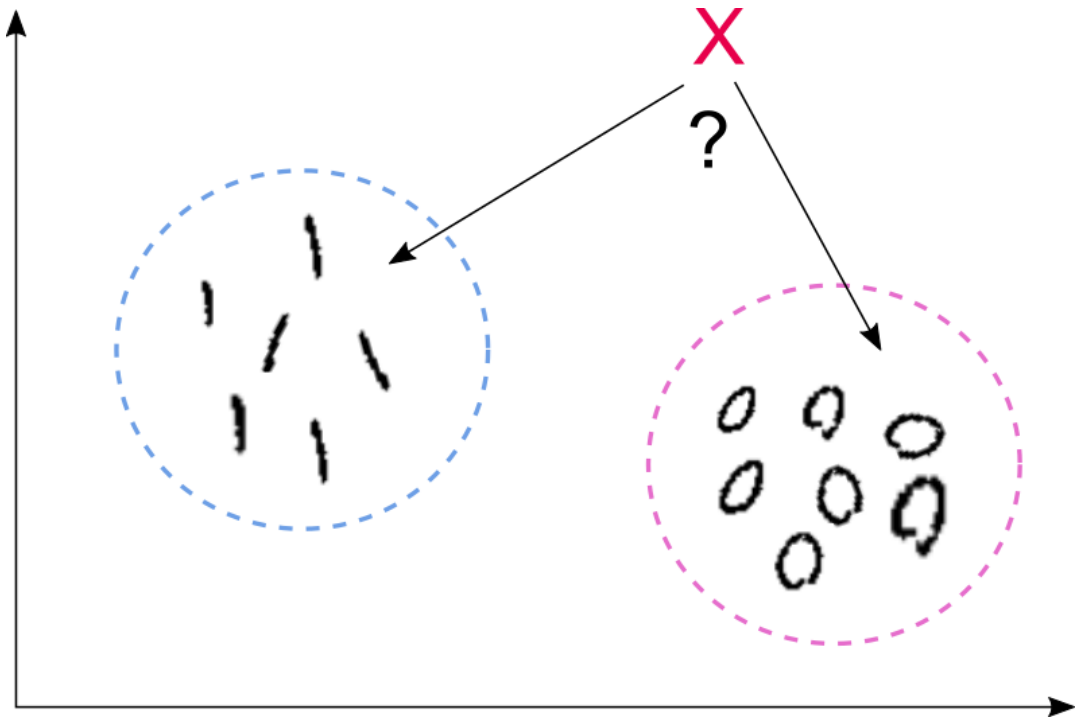


Huom. Piirreavaruus on oikeasti useampiulotteinen kuin tässä piirretty kaksiulotteinen

# Ykköset ja nollat – perusidea

- **Yleistä annetusta koulutusdatasta ennestään tuntemattomiin tapauksiin tarkistamalla, mitkä piirteet uudesta tapauksessa ovat (jollain tavalla) lähimpänä.**

- **Jos meillä on uusi X, kumpi se on?**

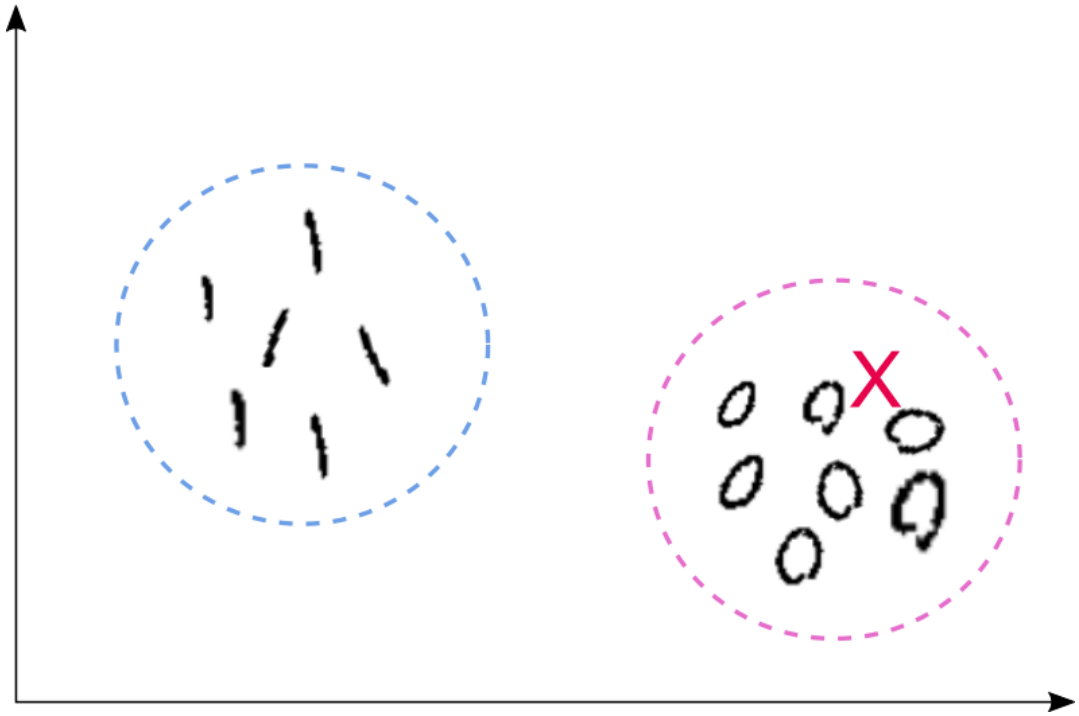



Huom. Piirreavaruus on oikeasti useampiulotteinen kuin tässä piirretty kaksiulotteinen

# Ykköset ja nollat – perusidea

- **Yleistä annetusta koulutusdatasta ennestään tuntemattomiin tapauksiin tarkistamalla, mitkä piirteet uudesta tapauksessa ovat (jollain tavalla) lähimpänä.**

- **Lähellä muita nollia -> 0**

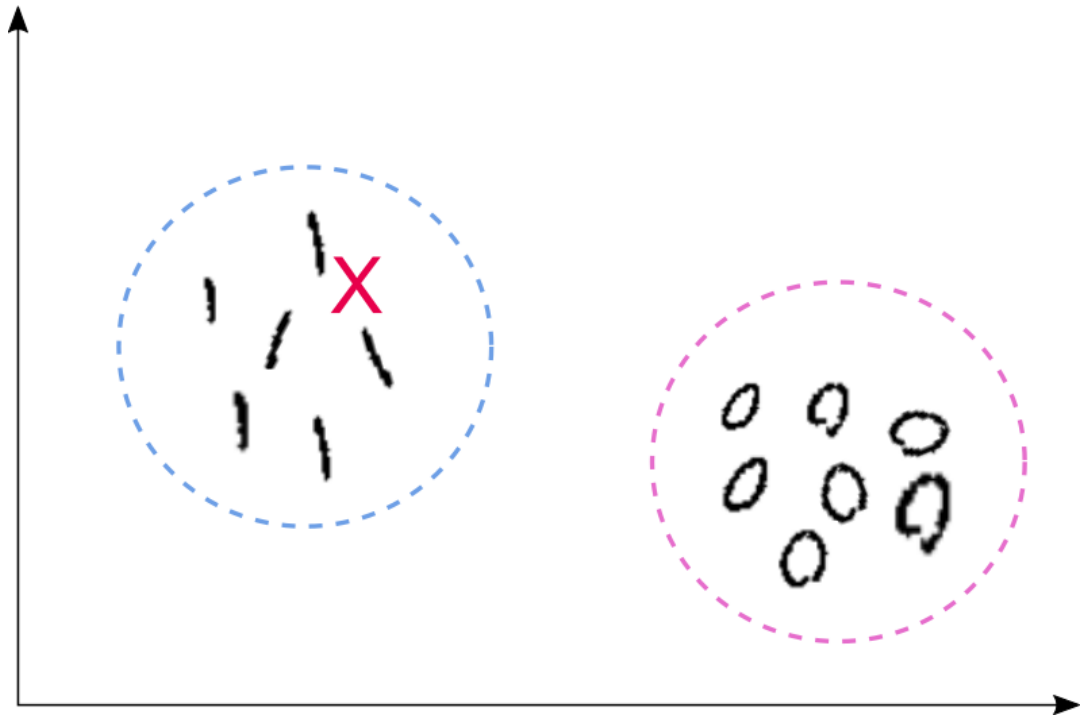



Huom. Piirreavaruus on oikeasti useampiulotteinen kuin tässä piirretty kaksiulotteinen

# Ykköset ja nollat – perusidea

- **Yleistä annetusta koulutusdatasta ennestään tuntemattomiin tapauksiin tarkistamalla, mitkä piirteet uudesta tapauksessa ovat (jollain tavalla) lähimpänä.**

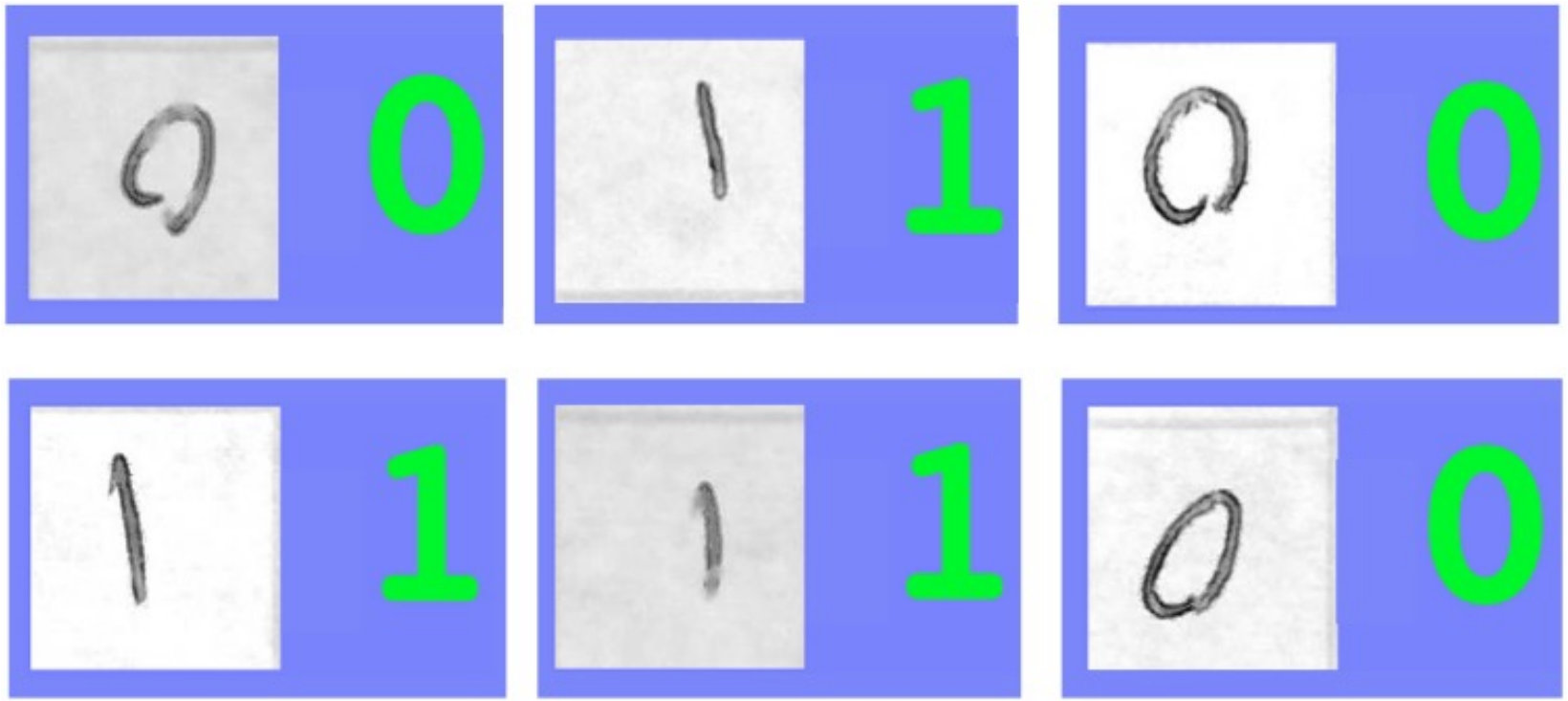
- **Lähellä muita ykkösiä -> 1**



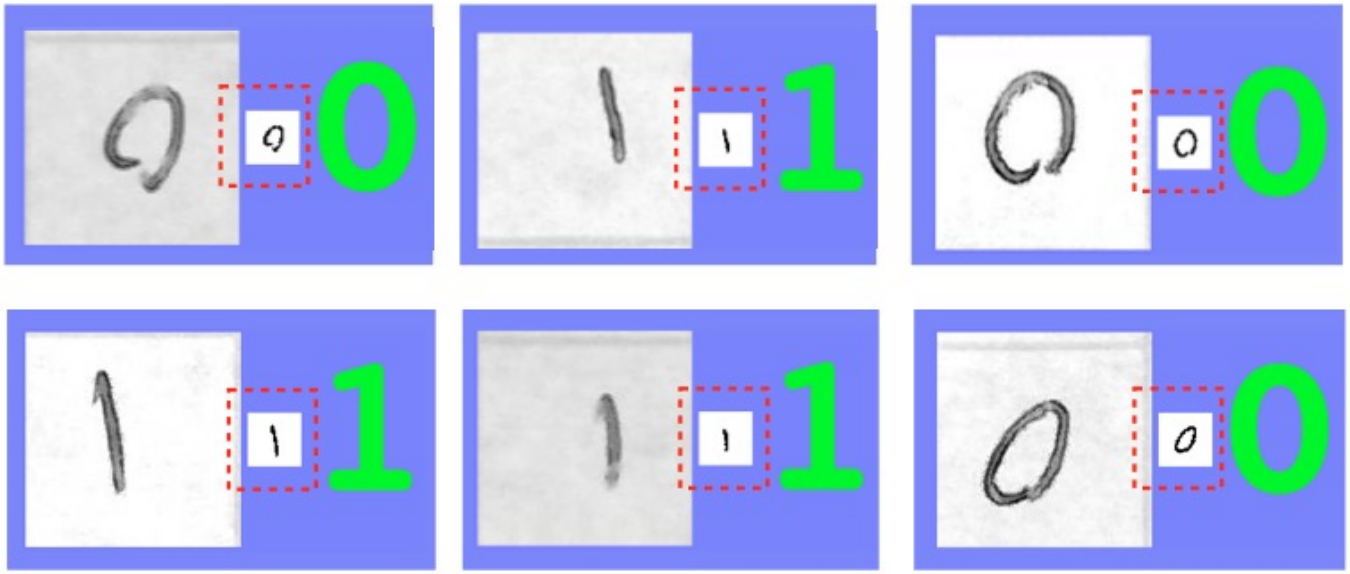

Huom. Piirreavaruus on oikeasti useampiulotteinen kuin tässä piirretty kaksiulotteinen

# Datan merkitseminen (labeling)

- **Kun kyseessä on ohjattu oppiminen, merkitään dataa:**



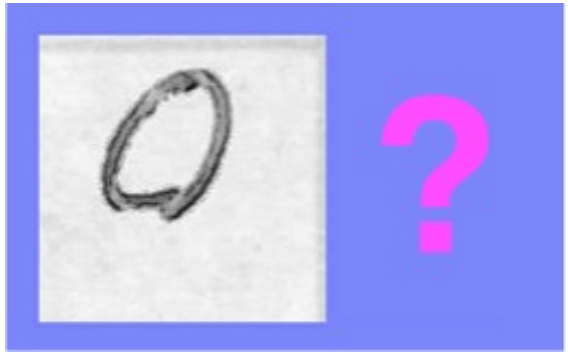
# Syötteen vähentäminen piirteitä muokkaamalla

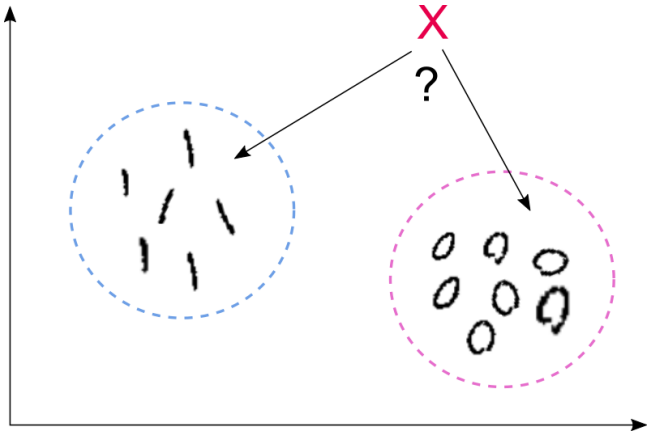


- **Piirteiden erotus, tässä esimerkissä:**
  - Muokataan näytteen kokoa: 160 × 160 -kuvasta 40 × 40 –kuvaksi
  - Harmaan sävyistä mustavalkoiseksi (binääri)kuvaksi
- **Muitakin (ehkä parempia) piirteitä voisi harkita**

# Koulutus – luokittelijan (classifier) luominen

- **Koulutusvaihe luo luokittelijan (classifier)**
  - Funktio, joka ottaa syötteenä kuvan ja palauttaa nimilapun/nimikkeen (label)
- **Se vastaa kysymykseen: mitkä nimikeet ovat lähinnä uutta kuvaa**
- **Tässä tapauksessa, oleta että samanlaiset kuvat ovat lähekkäin ominaisuusavaruudessa**
- **Alkeellinen luokittelija voisi tarkistaa**
  - Mikä on vektorin nimike koulutusdatan joukossa, joka on lähimpänä syötevektoria?
  - Esim käyttäen euklidista etäisyyttä






https://en.wikipedia.org/wiki/Euclidean_distance

# Ykköset ja nollat – yhteenveto

1. **Nimikkeiden (nimilappujen, label) lisääminen dataan**
2. **Jaetaan data kahtia koulutus- ja testijoukkoihin**
3. **Luodaan piirrevektorit**
4. **Oletus: ‘0’-kuvat ovat lähempänä toisiaan kuin ’1’-kuvia piirreavaruudessa (ja toistepäin)**
   - Sanomme, että ne ovat klusteroituneet (clustered)
5. **Uuden vektorin luokittelua varten tarkista onko se lähempänä koulutusdatan ‘0’-vektoreita kuin ‘1’-vektoreita**

- **Lopuksi tarkistetaan tulos testijoukon avulla**

# Muita tapoja mallin koulutuk-seen

A”

Aalto-yliopisto
Perustieteiden
korkeakoulu

# Tilastolliset mallit (statistical models)

- **Toinen vaihtoehto ratkaista ongelma on kysyä jotakuinkin:**
  - “Annettuna tämä syöte, millä tuloksella on korkein todennäköisyys olla oikea?”
- **Koulutus ~ valitun todennäköisyyksiin pohjautuvan mallin parametrien estimointi harjoitusdatajoukosta**
- **Esim. Luokitteluongelma voidaan ajatella olevan seuraavan kysymyksen vastaamista:**
  - “Mikä on todennäköisyys, että tulokselle tullaan antamaan nimilappu *c*, kun syöte on *x*?”

# Data jakautumina (distributions)

| Observation # | $X_1$ | $X_2$ | … | $X_k$ | Label ($C$) |
|---|---|---|---|---|---|
| 1 | … | … | … | … | … |
| 2 | … | … | … | … | … |
| ⋮ | … | … | … | … | … |
| m | … | … | … | … | … |

| Student | Favourite Lab (T7,T8,Maari A) | Language (Fi,Sv,Eng) | Free cake on Fridays? (Y/N) | … | Study Program |
|---|---|---|---|---|---|
| 1 | T8 | Eng | N | … | CS |
| 2 | T8 | Fi | Y | … | Data |
| … | … | … | … | … | … |

- **Oletetaan, että meillä on *k* havaittavaa asiaa (jotka voi mitata)**
  - Esim. sensoreita jossain instrumentissa, kyselyn kysymyksiä
- **Ja datamme sisältää m havaintoa (datapistettä, data points)**
  - Ja niihin liittyviä tunnisteita (joita joskus kutsutaan luokiksi)
- **Olettaen että koulutusdata on edustava otos (representative sample) ilmiön taustalla olevasta jakaumasta,**
  - jokainen havainto on tulos satunnaismuuttujasta $(X) = (X_1, X_2, \ldots, X_k)$
  - ja luokkamuuttujaa (tuloskategoriaa) tunnisteiden yli merkitään C:llä

# Bayesiläiset mallit (Bayesian models)

- **Bayesin kaava** $$\frac{P(C = c \mid \mathbf{X} = \mathbf{x})}{P(C = c)} = \frac{P(\mathbf{X} = \mathbf{x} \mid C = c)}{P(\mathbf{X} = \mathbf{x})}$$
- **"Mikä on todennäköisyys, että tulos on merkitty *c*:ksi, kun syöte on *x*"?** $$P(C = c \mid \mathbf{X} = \mathbf{x}) = \frac{P(C = c) \times P(\mathbf{X} = \mathbf{x} \mid C = c)}{P(\mathbf{X} = \mathbf{x})}$$
- **Posteriori eli jälkikäteinen (posterior) todennäköisyys, tapahtuman C todennäköisyys ehdolla X: P(C = c | X = x)**
- **Priori eli etukäteinen todennäköisyys (prior): P(C = c)**
- **Todennäköisyys (uskottavuus, likelihood): P(X = x | C = c)**
- **Todiste (evidence): P(X = x) eli tapahtuman X todennäköisyys**
- **Etsitään: mikä on suurin posterioritodennäköisyys tapahtumalle?**

# Naiivin bayesiläisen luokittelijan koulutus

- **Arvioidaan priori ja todennäköisyys (likelihood) dataan perustuen**
  - Todiste tarvittaisiin myös kaavassa, mutta sitä ei tarvita verratessa suhteellisia

$$P(C = c \mid \mathbf{X} = \mathbf{x}) \propto P(C = c) \times P(\mathbf{X} = \mathbf{x} \mid C = c)$$

  - Käyttäen suurimman uskottavuuden menetelmää (maximum likelihood estimation)
- **Oletetaan että muuttujat Xi ovat toisistaan riippumattomia (tämä on se naivi osa)**

- **Priori**

$$P(c) = \frac{\text{havaintojen lukumäärä luokassa c}}{\text{kaikkien havaintojen määrä}}$$

- **Todennäköisyys (voimassa vain naiivin oletuksen kanssa)**

$$P(\mathbf{X} = \mathbf{x} \mid C = c) = P(X_1 = x_1 \mid C = c) \times P(X_2 = x_2 \mid C = c) \times \ldots \times P(X_k = x_k \mid C = c)$$

- **Termit estimoituina:**

$$P(X_i = x_i \mid C = c) = \frac{\text{luokan c havaintojen määrä, joissa vastaus } X_i = x_i}{\text{luokan c havaintojen määrä}}$$



https://en.wikipedia.org/wiki/Maximum_likelihood_estimation

# Prosessista oppiminen

- **Datan ei tarviste olla nimenomaista (explicit)**
- **Voidaan generoida ‘prosessilla’, johon pääsee käsiksi oppimisen aikana**

COMPUTER SCIENCE

*Science* 07 Dec 2018:
Vol. 362, Issue 6419, pp. 1140-1144

## A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play

David Silver[1,2*†], Thomas Hubert[1*], Julian Schrittwieser[1*], Ioannis Antonoglou[1], Matthew Lai[1], Arthur Guez[1], Marc Lanctot[1], Laurent Sifre[1], Dharshan Kumaran[1], Thore Graepel[1], Timothy Lillicrap[1], Karen Simonyan[1], Demis Hassabis[1†]

The game of chess is the longest-studied domain in the history of artificial intelligence. The strongest programs are based on a combination of sophisticated search techniques, domain-specific adaptations, and handcrafted evaluation functions that have been refined by human experts over several decades. By contrast, the AlphaGo Zero program recently achieved superhuman performance in the game of Go by reinforcement learning from self-play. In this paper, we generalize this approach into a single AlphaZero algorithm that can achieve superhuman performance in many challenging games. Starting from random play and given no domain knowledge except the game rules, AlphaZero convincingly defeated a world champion program in the games of chess and shogi (Japanese chess), as well as Go.

Article | Published: 30 October 2019

## Grandmaster level in StarCraft II using multi-agent reinforcement learning

Oriol Vinyals, Igor Babuschkin, [...] David Silver

*Nature* **575**, 350–354(2019) | Cite this article

**53k** Accesses | **17** Citations | **963** Altmetric | Metrics

### Abstract

Many real-world applications require artificial agents to compete and coordinate with other agents in complex environments. As a stepping stone to this goal, the domain of StarCraft has emerged as an important challenge for artificial intelligence research, owing to its iconic and enduring status among the most difficult professional esports and its relevance to the real world in terms of its raw complexity and multi-agent challenges. Over the course of a decade and numerous competitions[1,2,3], the strongest agents have simplified important aspects of the game, utilized superhuman capabilities, or employed hand-crafted sub-systems[4]. Despite these advantages, no previous agent has come close to matching the overall skill of top StarCraft players. We chose to address the challenge of StarCraft using general-purpose learning methods that are in principle applicable to other complex domains: a multi-agent reinforcement learning algorithm that uses data from both human and agent games within a diverse league of continually adapting strategies and counter-strategies, each represented by deep neural networks[5,6]. We evaluated our agent, AlphaStar, in the full game of StarCraft II, through a series of online games against human players. AlphaStar was rated at Grandmaster level for all three StarCraft races and above 99.8% of officially ranked human players.

# Sovelluskehyksiä (frameworks)

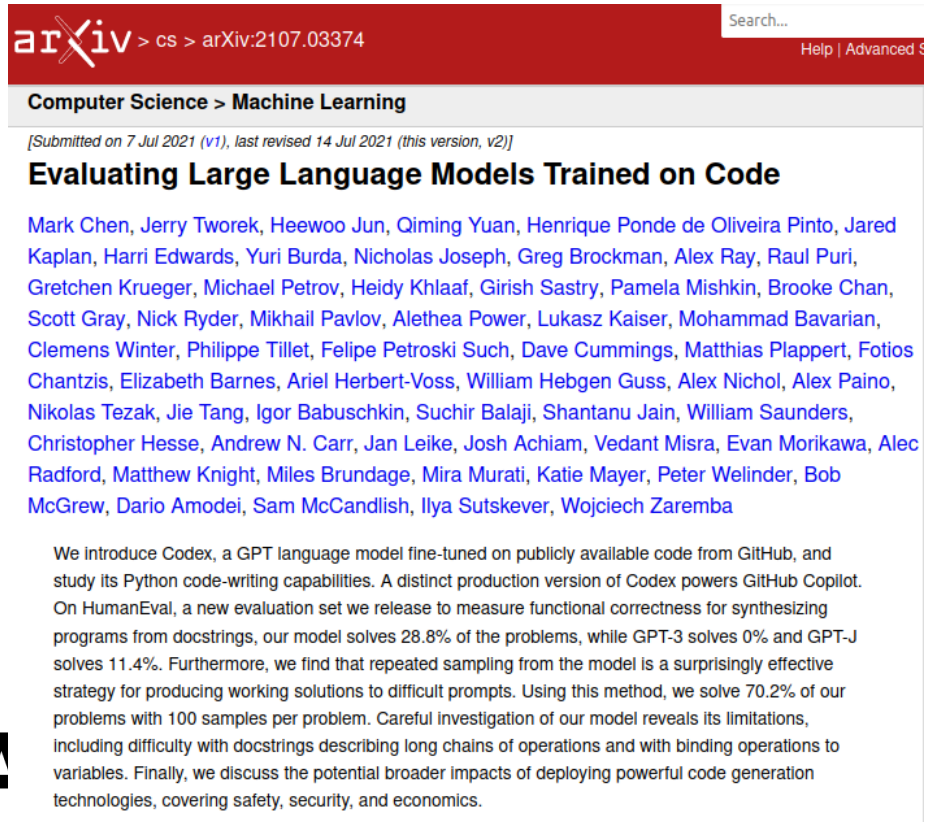


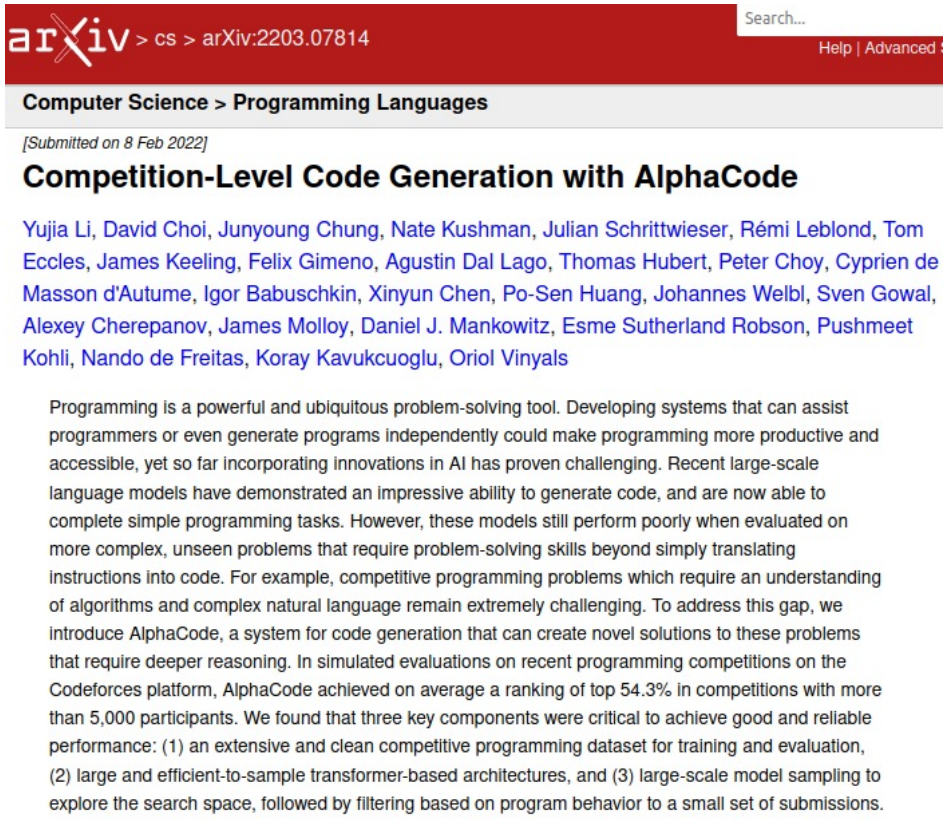
- **Näitä on monia**
- **Me käytämme pelkkää Scalaa harjoituksissa, joissa perusteita**

- **Jatkokursseilla varmasti lisää**

# Ohjelmoimaan koulutettuja koneita?

- **Voisiko kone kirjoittaa ohjelmia (esim. Scalalla)?**
- **Kyllä, kuten monet teistä tietävät: esim. Claude, ChatGPT, GitHub Copilot, CodeGen, AlphaCode, Bard,…**



# Miten nämä toimivat?

- **(Suuret) kielimallit (Large Language Model)**
  - Yleensä toteutettuna syvinä neuroverkkoina (deep neural network)
  - (Sorry, yksityiskohdat myöhemmillä kursseilla)
  - Alunperin luonnollisen kielen kääntämiseen / tiivistämiseen
  - Mutta osoittautuvat olevan aika hyviä ohjelmakoodin tekemisellekin…

# Harjoitukset

1. **Binäärinäkö**
2. **Matriisin käänteismuunnos (matrix inversion) ja sen sovellukset**
   - Tämän totutus ja
   - Pienimmän neliösumman menetelmä (least squares method)
3. **Haaste: singulaariarvon hajoittaminen**

- **Katso kalvojen esimerkit**
- **Binäärinäössä on pääohjelma**
- **Matriisin käänteismuunnos: useampi mahdollinen lähestymistapa:**
  - lue tehtävän kuvauksesta mahdollisista menetelmistä

# Kiitoksia osallistumisesta!

- **Muistakaa ilmoittautua tenttiin, jos aiotte siihen osallistua!**
  - DL 14.5.2026
  - Lisätietoja MyCo-sivulla
- **Olkaa hyvät, ja täyttäkää kurssipalautelomake!**
  - Se aukeaa 23.5.2026