Machine Learning Basics

“Scientific study of algorithms and statistical models that computer systems use to progressively improve their performance on a specific task”

Wikipedia

"A computer program is said to learn from **experience** *E* with respect to some class of **tasks** *T* and **performance measure** *P*, if its performance at tasks in *T* as measured by *P*, improves with **experience** *E*."

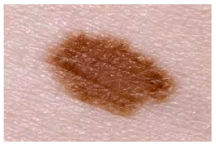
Machine Learning (T. Mitchell, 1997)

Example: melanoma diagnosis

Melanoma or healthy mole?

- **Task:** determine if image is melanoma or not (binary classification)
- **Performance measure:** misclassification probability (on test set)
- **Experience:** labelled images (training set)

melanoma



healthy

...

melanoma

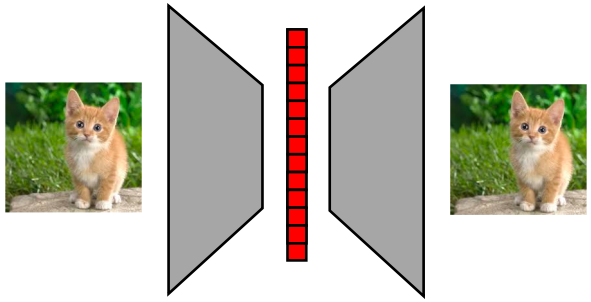
Different settings of learning

Supervised

Unsupervised
Clustering

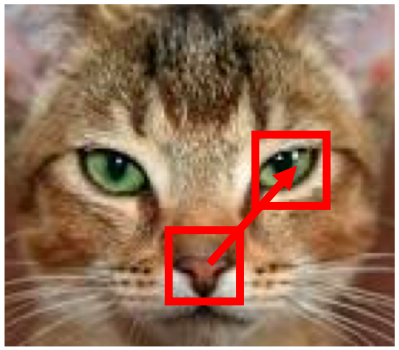
Different settings of learning

Supervised



Unsupervised
Clustering, Autoencoders

Semi-supervised
Partially labelled



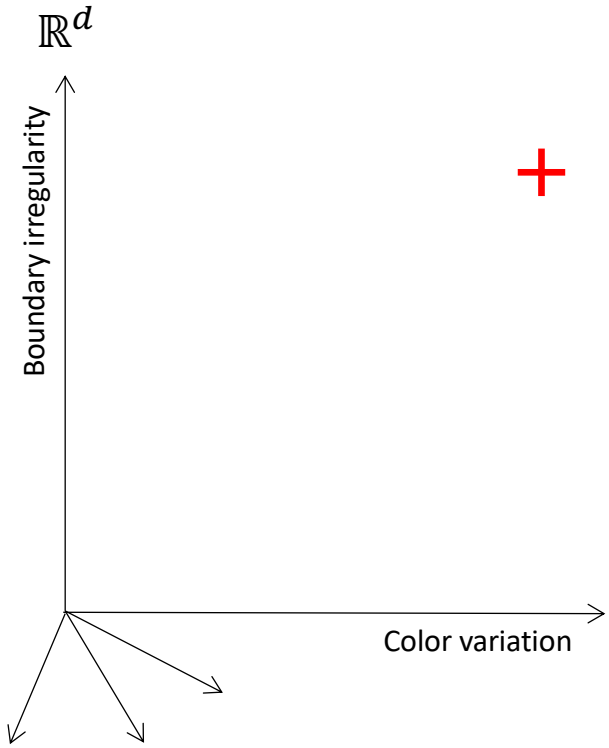
Self-supervised
Proxy task

(Reinforcement)

Example: image classification

Feature extractor

Input

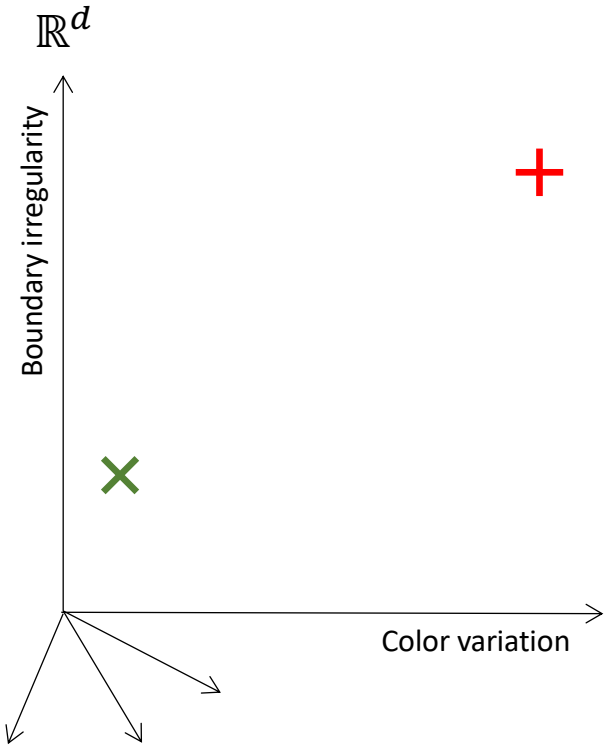


Representation (features)

Example: image classification

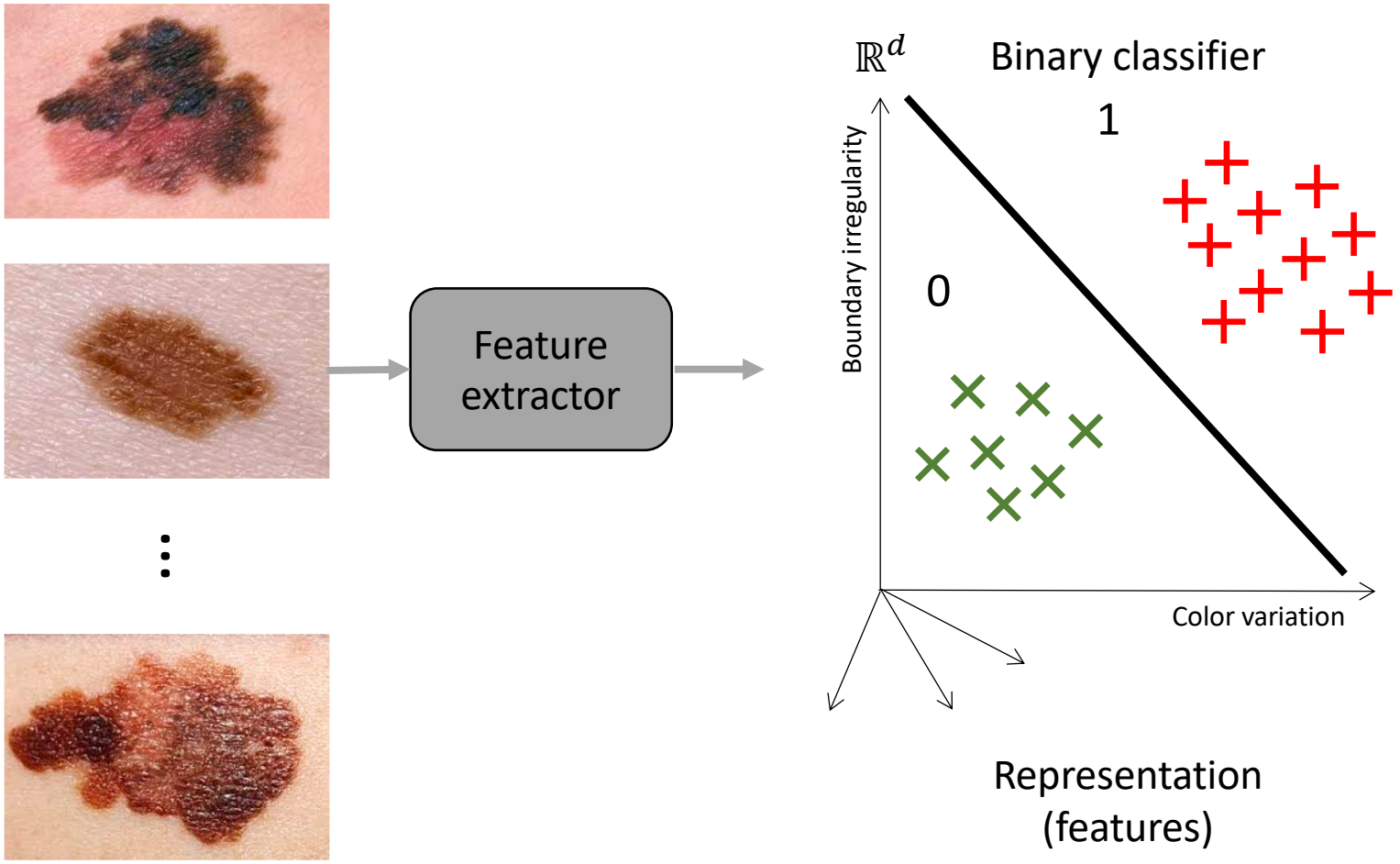
Input

Feature extractor

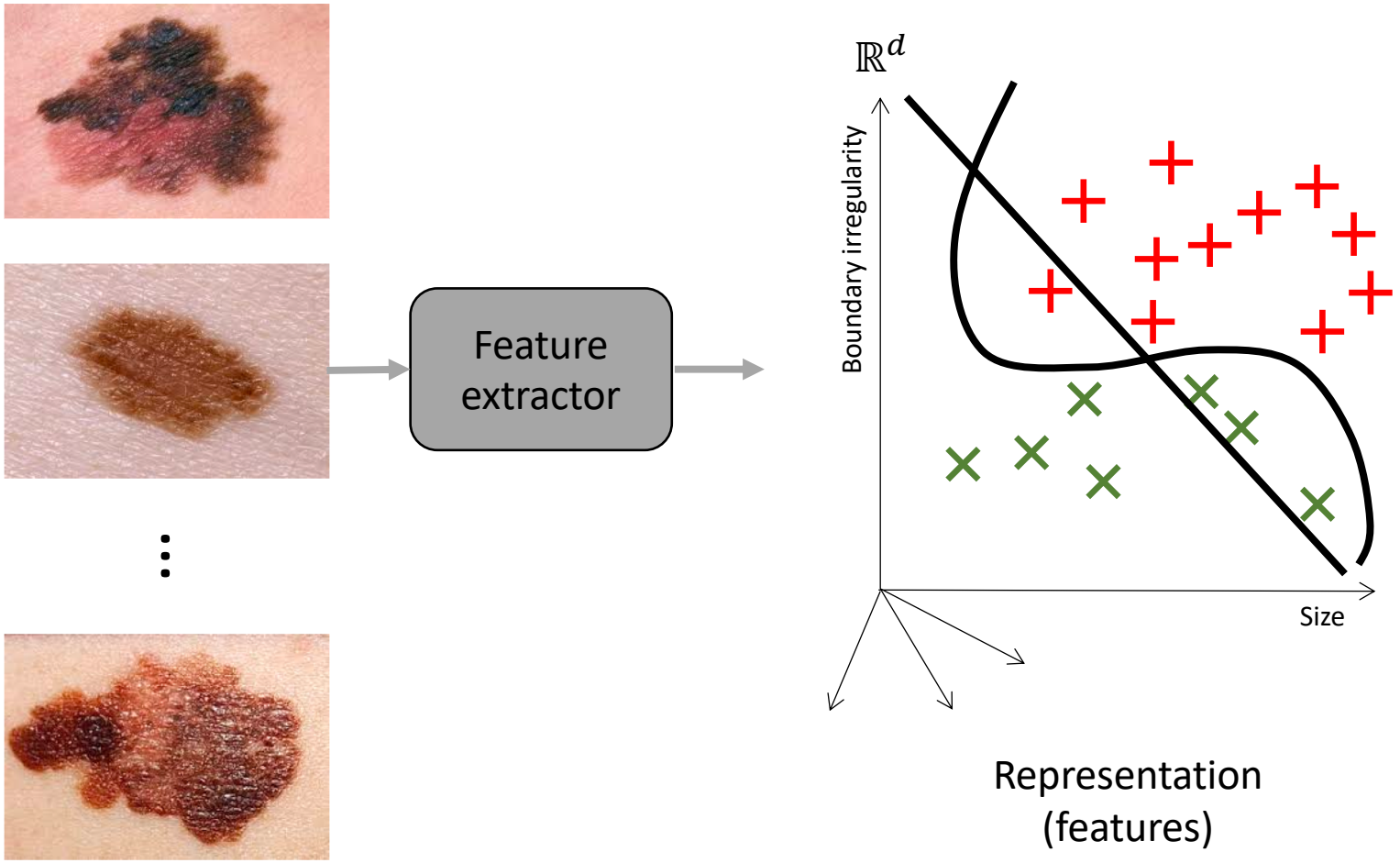


Representation (features)

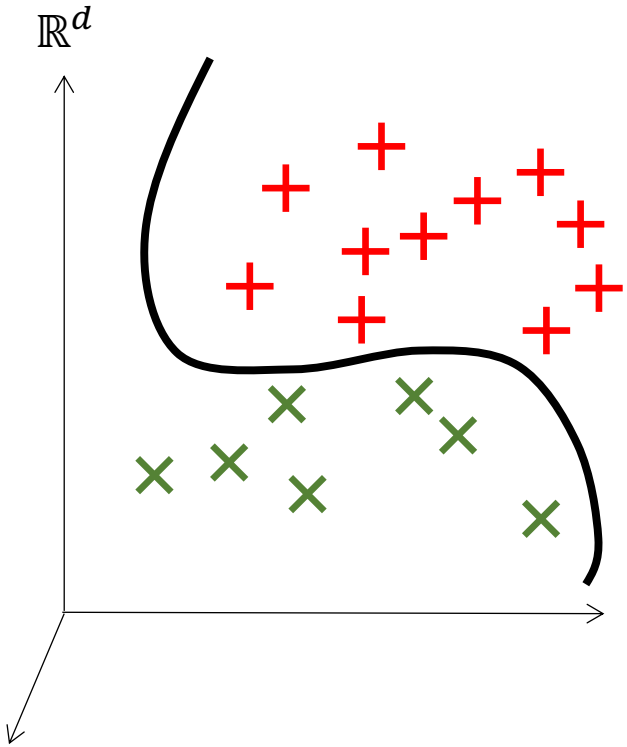
Example: image classification



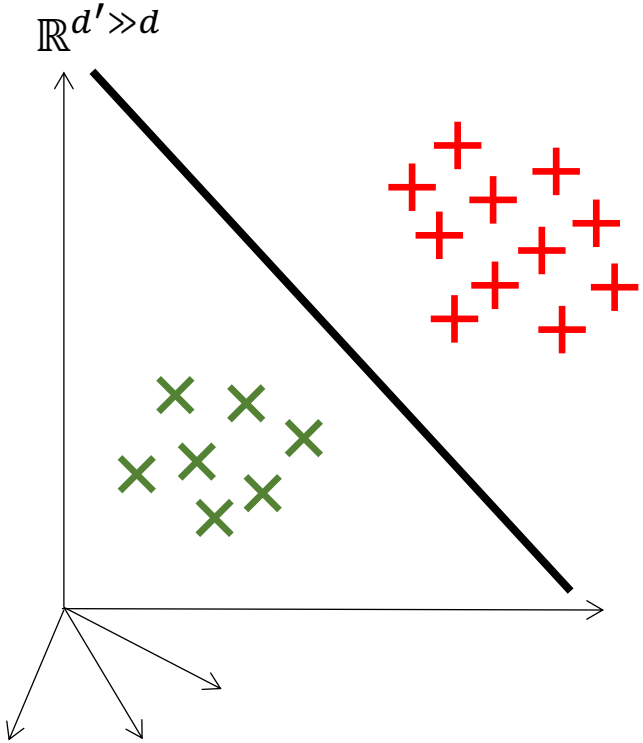
Example: image classification



Representation

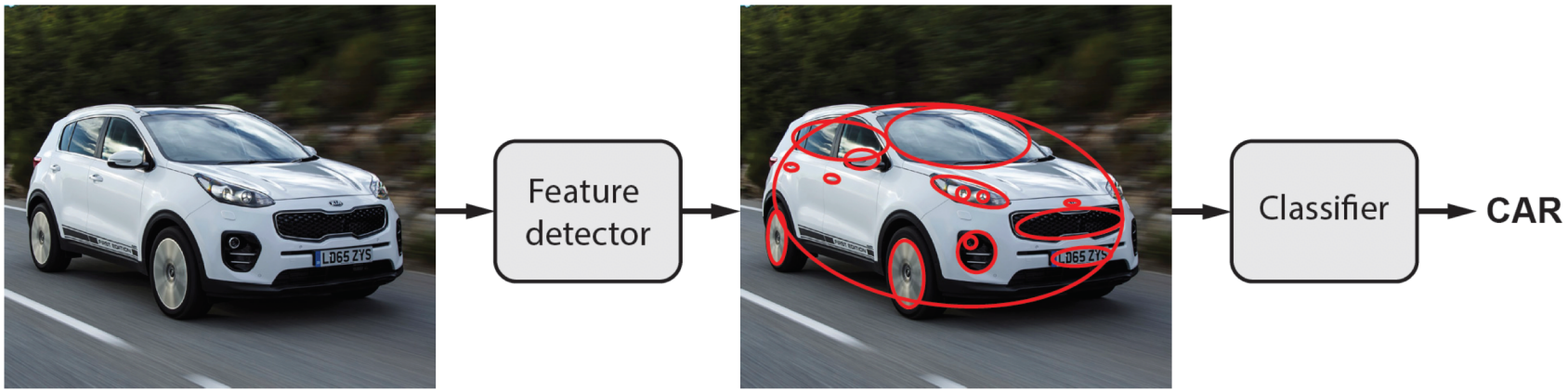


Original data $\mathbf{x}$
Non-linear classifier

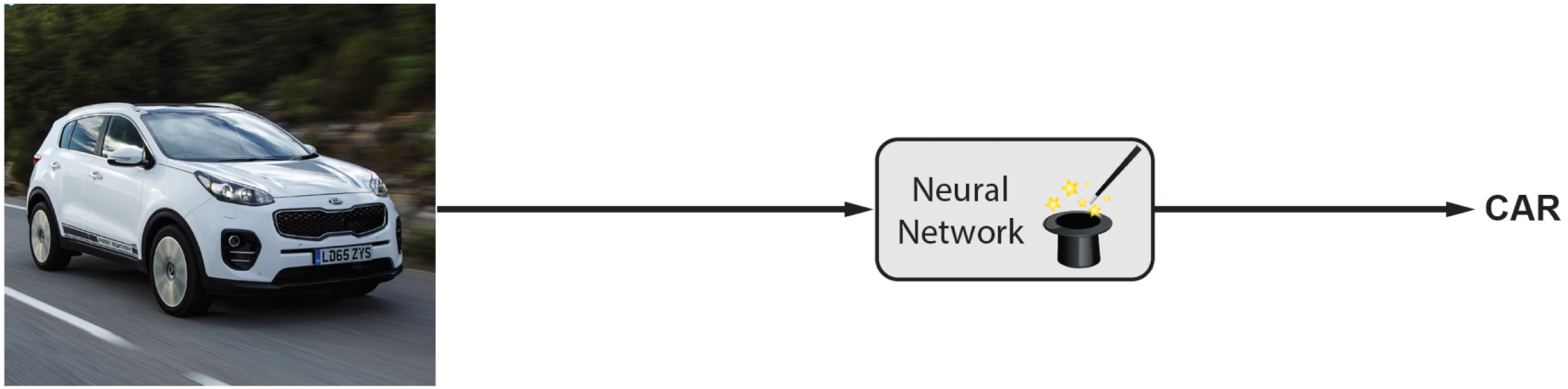


Non-linear features $\phi(\mathbf{x})$
Linear classifier

Representation learning

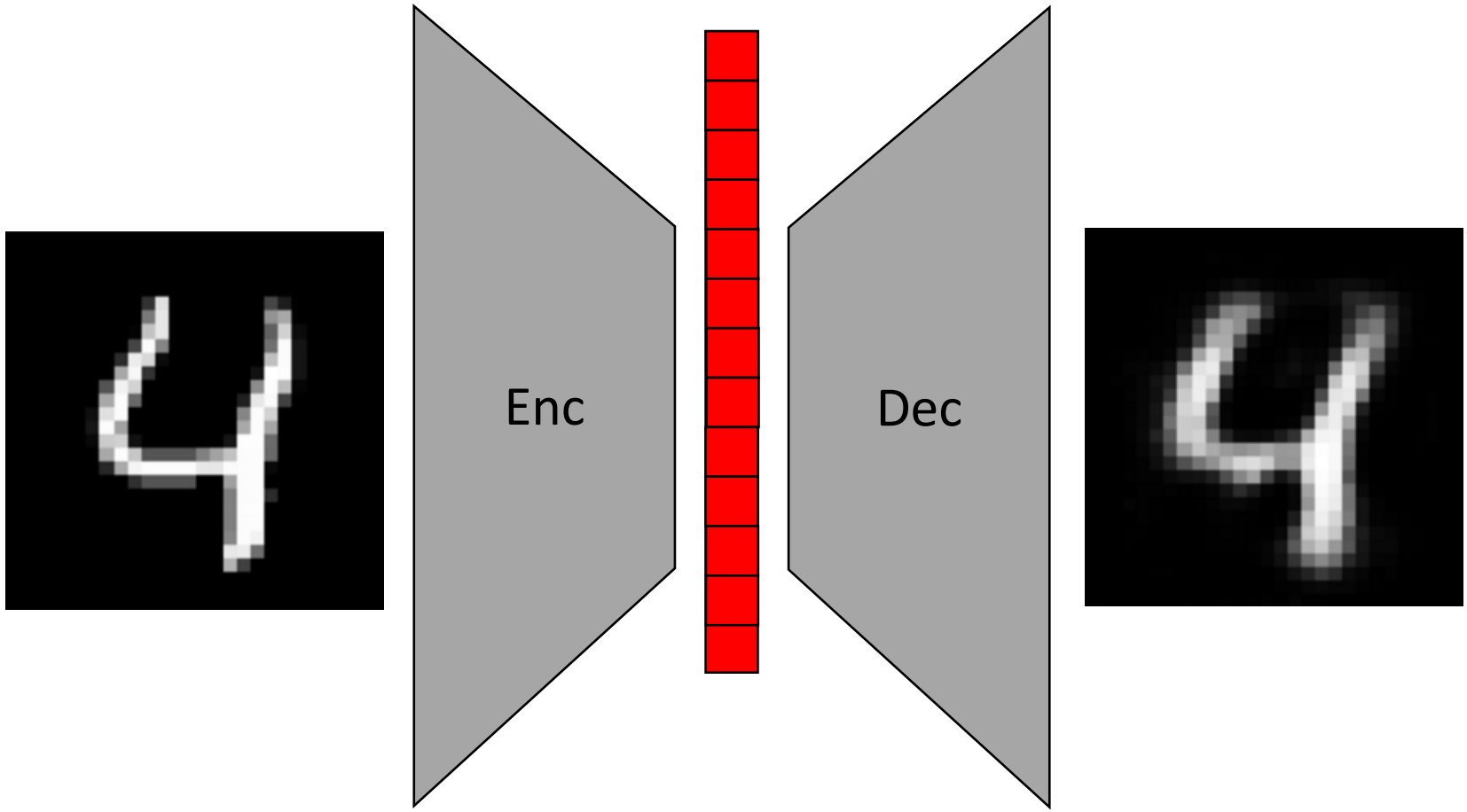


Classical way: extract ”handcrafted” features, then apply a simple (e.g. linear) classifier



Modern way: learn features directly from image

Representation learning



Autoencoder

ICLR | 2019

Seventh International Conference on Learning Representations

Ernest N. Morial Convention Center, New Orleans

Mon May 6th through Thu the 9th

Tourism in New Orleans »

Schedule

Workshops »

General Chair

- Tara Sainath, Google

Senior Program Chair

- Alexander Rush, Harvard University

Program Chairs

- Sergey Levine, UC Berkeley
- Karen Livescu, TTI-Chicago
- Shakir Mohamed, Google DeepMind

Workshop Chairs

- Been Kim, Google Brain
- Graham Taylor, University of Guelph / Vector Institute

Diversity+Inclusion Chairs

- Alice Oh, KAIST
- Richard Zemel, University of Toronto / Vector Institute

Contact

The organizers can be contacted here.

Announcements

- Registration will open on January 29, 2019
- Student travel award and Volunteer applications will open with registration
- Do not book travel until you have registered for the conference
- Apply for your visa (if necessary) by March 7, 2019. See our Registration Cancellation Policy

Important Dates:

- **Sep 27,** Paper Submission Deadline
- **Late Jan 2019,** Registration Opens
- **Mar 19,** Early Reg Pricing Deadline
- **Apr 16**, Reg Cancellation Deadline

View All Dates »

Sponsors

View ICLR 2019 sponsors »

Become a 2019 Sponsor »

ICLR 2019

New Orleans, Louisiana

May 6 – 9, 2019

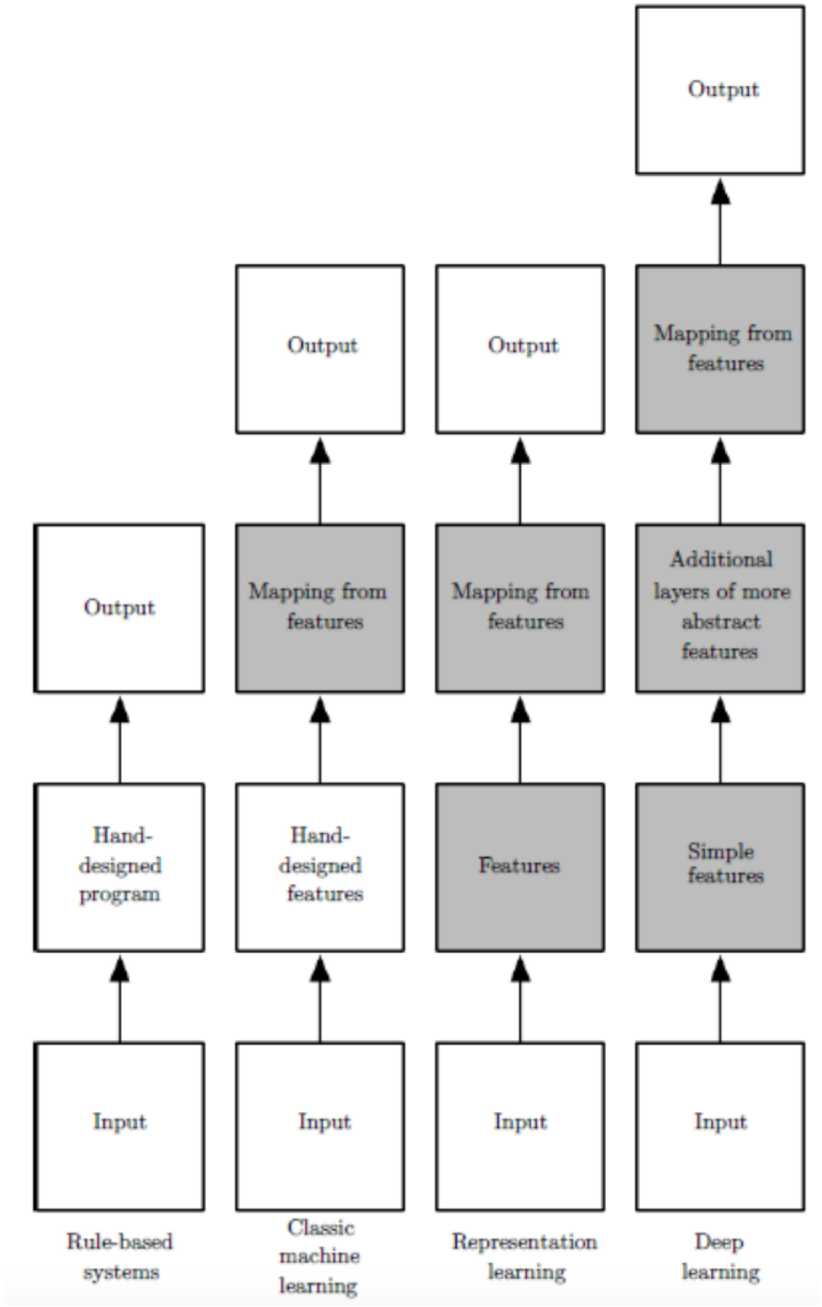
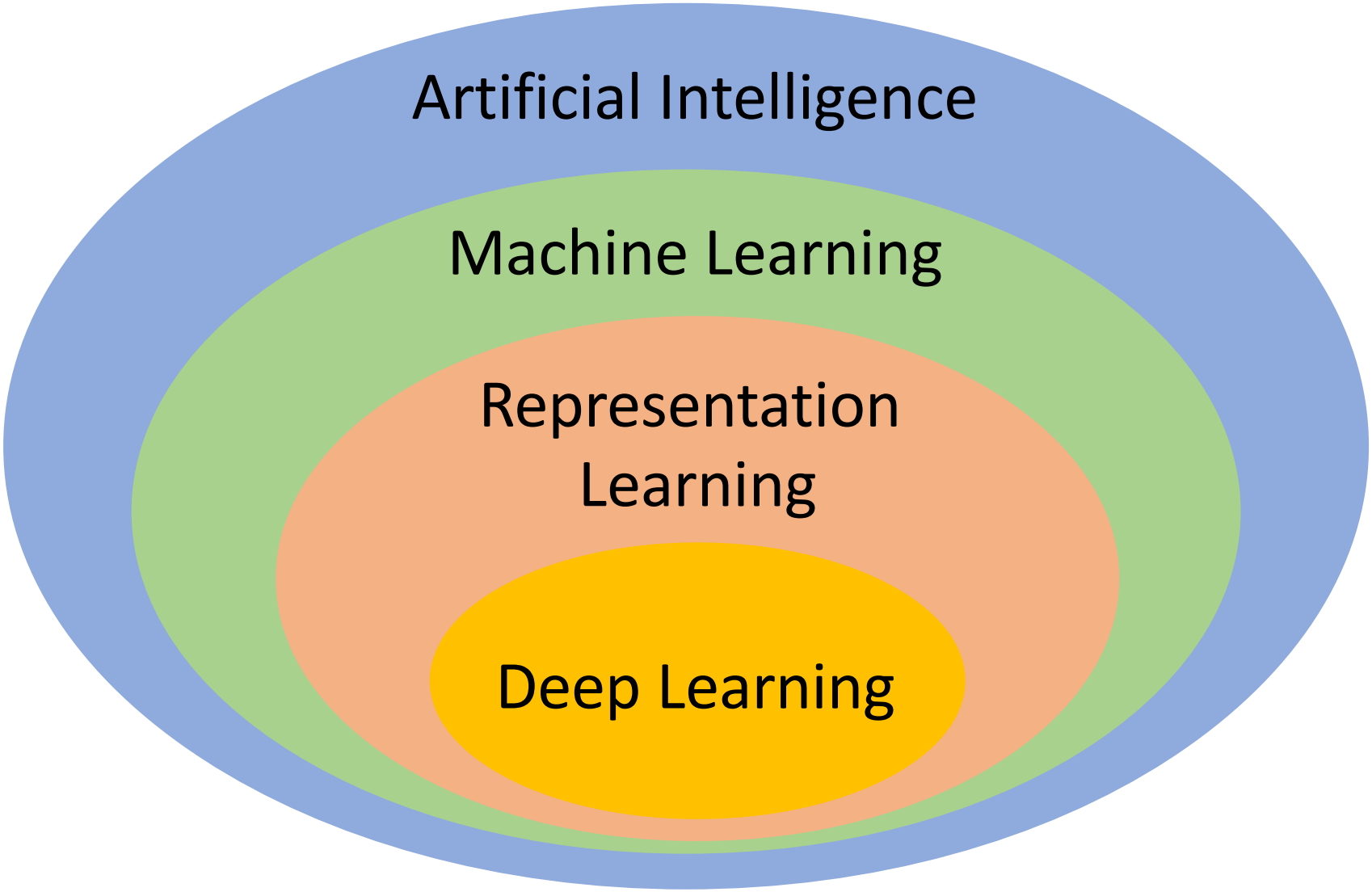
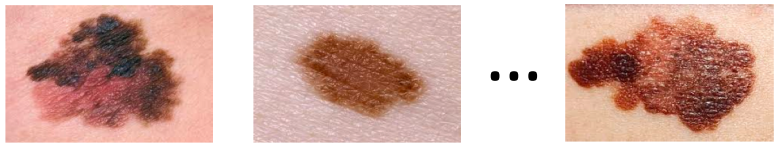


Figure: Goodfellow et al. 2016


Artificial Intelligence
Machine Learning
Representation Learning
Deep Learning

Slightly more formally…

- Given training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$



- Find a model $\hat{y} = f(\mathbf{x})$ using training data
- such that $f(\mathbf{x}) \approx y$ on test data

Machine learning = function estimation

Slightly more formally…

- Given training data $\{(\mathbf{x}_i, y_i) \sim p \text{ i.i.d.}\}_{i=1}^{n}$
- Find optimal parameters $\boldsymbol{\theta}$ of a parametric model $\hat{y} = f_{\boldsymbol{\theta}}(\mathbf{x})$ using training data
- such that $f(\mathbf{x}) \approx y$ on test data $\sim p$ i.i.d.

Slightly more formally…

- Given training data $\{(\mathbf{x}_i, y_i) \sim p \text{ i.i.d.}\}_{i=1}^n$
- Find optimal parameters $\boldsymbol{\theta}$ of a parametric model $\hat{y} = f_{\boldsymbol{\theta}}(\mathbf{x})$ using training data
- such that expected loss $L(\boldsymbol{\theta}) = \mathbb{E}_{(\mathbf{x},y)\sim p}\, \ell(f_{\boldsymbol{\theta}}(\mathbf{x}), y)$ is small
  - L_2 loss: $\ell(\hat{y}, y) = |\hat{y} - y|^2$
  - L_1 loss: $\ell(\hat{y}, y) = |\hat{y} - y|$
  - 0-1 loss: $\ell(\hat{y}, y) = \begin{cases} 1 & \hat{y} = y \\ 0 & \hat{y} \neq y \end{cases}$
  - many other…

Slightly more formally…

- Given training data $\{(\mathbf{x}_i, y_i) \sim p \ \text{ i.i.d.}\}_{i=1}^{n}$
- Find optimal parameters $\boldsymbol{\theta}$ of a parametric model $\hat{y} = f_{\boldsymbol{\theta}}(\mathbf{x})$ by minimizing empirical loss

$$\hat{L}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$$

- such that expected loss $L(\boldsymbol{\theta}) = \mathbb{E}_{(\mathbf{x},y)\sim p}\, \ell(f_{\boldsymbol{\theta}}(\mathbf{x}), y)$ is small

Main ingredients of an ML problem

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \; \frac{1}{n} \sum_{i=1}^{n} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$$

- **Data** (training/test set, features)
- **Model** $f_{\boldsymbol{\theta}}$ and **loss function** ℓ
- **Optimisation** (how to find best model parameters $\boldsymbol{\theta}^*$)

Function approximation

Statistical estimation

Optimisation theory

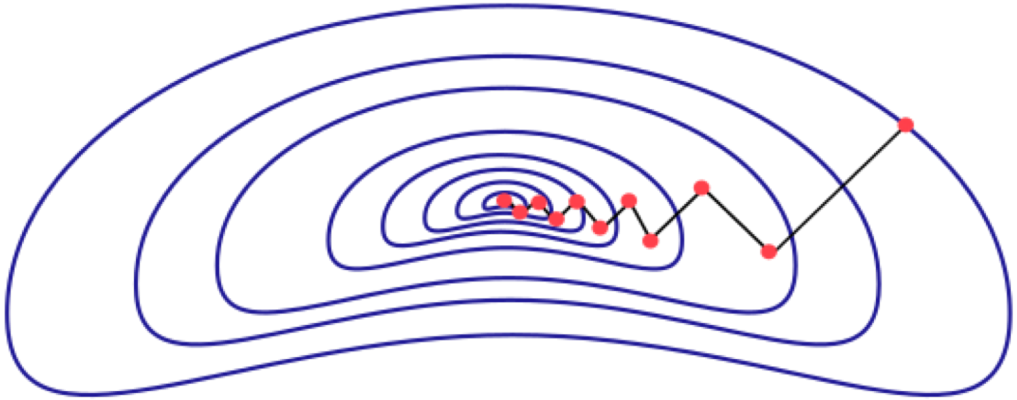
Learning as optimization problem

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$$

- **Gradient descent** to produce a sequence of iterative solutions
  - Initialize $\boldsymbol{\theta}^{(0)}$
  - Update $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}} \hat{L}\left(\boldsymbol{\theta}^{(t)}\right)$
  - Repeat until convergence

step size

gradient



Stochastic gradient descent

- Observe that

$$\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$$

- **Stochastic gradient descent (SGD):** approximate the gradient using current sample $\nabla_{\boldsymbol{\theta}} \hat{L}(\boldsymbol{\theta}) \approx \nabla_{\boldsymbol{\theta}} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_t), y_t)$ and do update as

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_t), y_t)$$

- **Mini-batch:** use a small batch of b samples $\{(\mathbf{x}_{tb+i}, y_{tb+i})\}_{i=1}^{b}$

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha^{(t)} \frac{1}{b} \sum_{i=1}^{b} \nabla_{\boldsymbol{\theta}} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_{tb+i}), y_{tb+i})$$

Stochastic gradient descent

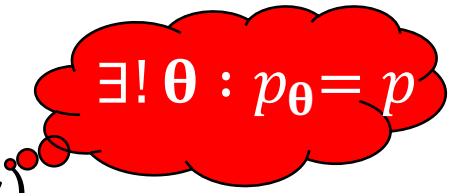
- Step size $\alpha^{(t)}$ selection
  - Fixed (simplest)
  - Line search (expensive)
  - Adaptive (AdaGrad, Adam,…)

Maximum likelihood estimation

Goodfellow et al. 2016 (sec. 5.5)

Maximum likelihood

- Given training data $\{(\mathbf{x}_i, y_i) \sim p \ \text{i.i.d.}\}_{i=1}^{n}$ drawn from unknown **data-generating distribution** $p(\mathbf{x}, y)$
- Parametric family of **model distributions** $p_{\boldsymbol{\theta}}(\mathbf{x}, y)$
- Find parameters $\boldsymbol{\theta}$ such that $p_{\boldsymbol{\theta}}(\mathbf{x}, y) \approx p(\mathbf{x}, y)$



Maximum likelihood

- Given training data $\{(\mathbf{x}_i, y_i) \sim p \ \text{ i. i. d. }\}_{i=1}^{n}$ drawn from unknown **data-generating distribution** $p(\mathbf{x}, y)$
- Parametric family of **model distributions** $p_{\boldsymbol{\theta}}(\mathbf{x}, y)$
- Find parameters $\boldsymbol{\theta}$ such that **Kullback-Leibler divergence** is small

$$DL(p, p_{\boldsymbol{\theta}}) = \int_{\mathcal{X}\times\mathcal{Y}} p(\mathbf{x}, y) \log\left(\frac{p(\mathbf{x}, y)}{p_{\boldsymbol{\theta}}(\mathbf{x}, y)}\right) d\mathbf{x}dy$$

$$= \int_{\mathcal{X}\times\mathcal{Y}} p(\mathbf{x}, y)[\log p(\mathbf{x}, y) - \log p_{\boldsymbol{\theta}}(\mathbf{x}, y)]\, d\mathbf{x}dy$$

$$= \mathbb{E}_{(\mathbf{x},y)\sim p}\,[\log p(\mathbf{x}, y) - \log p_{\boldsymbol{\theta}}(\mathbf{x}, y)]$$

Maximum likelihood

- Given training data $\{(\mathbf{x}_i, y_i) \sim p \text{ i.i.d.}\}_{i=1}^{n}$ drawn from unknown **data-generating distribution** $p(\mathbf{x}, y)$
- Parametric family of **model distributions** $p_{\boldsymbol{\theta}}(\mathbf{x}, y)$
- Minimize **Kullback-Leibler divergence** w.r.t. parameters $\boldsymbol{\theta}$

$$\widehat{\boldsymbol{\theta}} = \operatorname*{argmin}_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x},y)\sim p} \left[\log p(\mathbf{x}, y) - \log p_{\boldsymbol{\theta}}(\mathbf{x}, y)\right]$$

$$= \operatorname*{argmin}_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x},y)\sim p} - \log p_{\boldsymbol{\theta}}(\mathbf{x}, y)$$

$$\approx \operatorname*{argmin}_{\boldsymbol{\theta}} -\log \prod_{i=1}^{n} p_{\boldsymbol{\theta}}(\mathbf{x}_i, y_i) = \operatorname*{argmin}_{\boldsymbol{\theta}} - \sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(\mathbf{x}_i, y_i)$$

i.i.d.

Some equivalent terminology

- **Cross entropy**

$$H(p, p_{\boldsymbol{\theta}}) = H(p) + DL(p, p_{\boldsymbol{\theta}}) = -\mathbb{E}_{(\mathbf{x},y)\sim p} \log p_{\boldsymbol{\theta}}(\mathbf{x}, y)$$

- **Log-likelihood** $\mathbb{E}_{(\mathbf{x},y)\sim p} \log p_{\boldsymbol{\theta}}(\mathbf{x}, y)$

Conditional likelihood

- Given training data $\{(\mathbf{x}_i, y_i) \sim p \ \text{ i.i.d.}\}_{i=1}^{n}$ drawn from unknown data-generating distribution $p(\mathbf{x}, y)$
- We are only interested in predicting y from $\mathbf{x}$: use parametric family of **conditional** model distributions $p_{\boldsymbol{\theta}}(y|\mathbf{x})$
- Find optimal model parameters

$$\widehat{\boldsymbol{\theta}} = \operatorname*{argmin}_{\boldsymbol{\theta}} -\sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(y_i|\mathbf{x}_i)$$

Example: L_2 loss as MLE

- Given training data $\{(\mathbf{x}_i, y_i) \sim p \ \text{i.i.d.}\}_{i=1}^n$
- Find optimal parameters $\boldsymbol{\theta}$ of a parametric model $\hat{y} = f_{\boldsymbol{\theta}}(\mathbf{x})$

$$\widehat{\boldsymbol{\theta}} = \operatorname*{argmin}_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^{n} (f_{\boldsymbol{\theta}}(\mathbf{x}_i) - y_i)^2$$

- Assume $p_{\boldsymbol{\theta}}(y|\mathbf{x}) = \mathcal{N}(f_{\boldsymbol{\theta}}(\mathbf{x}), \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(f_{\boldsymbol{\theta}}(\mathbf{x}) - y)^2}$
- $\log p_{\boldsymbol{\theta}}(y_i|\mathbf{x}_i) = -\frac{1}{2\sigma^2}(f_{\boldsymbol{\theta}}(\mathbf{x}_i) - y_i)^2 + \text{const}$

$$\widehat{\boldsymbol{\theta}} = \operatorname*{argmin}_{\boldsymbol{\theta}} -\sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(y_i|\mathbf{x}_i) = \operatorname*{argmin}_{\boldsymbol{\theta}} \frac{1}{2\sigma^2} \sum_{i=1}^{n} (f_{\boldsymbol{\theta}}(\mathbf{x}_i) - y_i)^2$$

Bias and variance

Goodfellow et al. 2016 (sec. 5.4)

Properties of estimators

- $\text{Bias}\,\widehat{\boldsymbol{\theta}}_n = \mathbb{E}\widehat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}$

  unbiased: $\text{Bias}\,\widehat{\boldsymbol{\theta}}_n = 0$

  asymptotically unbiased: $\text{Bias}\,\widehat{\boldsymbol{\theta}}_n \xrightarrow{n\to\infty} 0$

- **Mean squared error (MSE):**

$$\mathbb{E}\left(\widehat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}\right)^2 = \left(\text{Bias}\,\widehat{\boldsymbol{\theta}}_n\right)^2 + \text{Var}\,\widehat{\boldsymbol{\theta}}_n$$

- **Consistency:** $\operatorname{plim}_{n\to\infty} \widehat{\boldsymbol{\theta}}_n = \boldsymbol{\theta}$, implying that for any $\epsilon > 0$,
  $P\left(\left|\widehat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}\right| > \epsilon\right) \to 0$

Properties of ML estimator

Under some technical conditions, the ML estimator is

- **Consistent** (i.e., converges in probability to the value being estimated)
- **Efficient** (i.e., asymptotically achieves the Cramér-Rao bound = no other consistent estimator has lower asymptotic MSE)

Bias and variance tradeoff