PnS 2018

Deep Learing with Raspberry Pi

Session 2

PnS 2018 Team

Institute of Neuroinformatics
University of Zürich and ETH Zürich

Outline

Definition of a Learning Algorithm

"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" (Mitchell, 1997)

The task T

Classification specify which k categories some input belongs to. $(f : \mathbb{R}^n \to \{1, \ldots, k\})$

Regression predict a numerical value given some input. $(f : \mathbb{R}^n \to \mathbb{R})$

Transcription output a sequence of symbols, rather than a category code. (similar to classification, *e.g.* speech recognition, machine translation, image captioning)

Denoising predict *clean* samples $\boldsymbol{x}$ from *corrupted* samples $\tilde{\mathbf{x}}$ (estimate $P(\mathbf{x}|\tilde{\mathbf{x}})$).

Many more types are not listed here.

The performance measure P

- Measure P is usually specific to the task T. (*e.g.* accuracy to classification)
- Batches of unseen *test* data is introduced to measure performance.
- Design measure P can be very subtle. It should be effective.

The experience E

Experience is what learning algorithms are allowed to have during learning process.

- Experience is usually an *dataset*, a collection of *examples*.
- *Unsupervised learning algorithms* experience a dataset containing many features, learning useful structure of the dataset (estimate $p(\mathbf{x})$).
- *Supervised learning algorithms* experience a dataset containing features, but each example is also associated with a *label* or *target* (estimate $p(\mathbf{y}|\mathbf{x})$).

Problem Description

The *task* is to build a system that can take a vector $\mathbf{x} \in \mathbb{R}^n$ as input and predict the value of a scalar $y \in \mathbb{R}$ as its output. Let $\hat{y}$ be the value that out model predicts y should take on. We define the output to be

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b$$

where $\mathbf{w} \in \mathbb{R}^n$ is a vector of parameters and b is a bias term.

Performance Measure and Experience

The *experience* contains a set of training examples where each sample is a pair of input and output $(\mathbf{x}, y)$.
One *performance measure* here can apply is *mean squared error* of the model on test set. Let test example as $\mathbf{x}^{(\text{test})}$ and regression targets as $\mathbf{y}^{(\text{test})}$, $\hat{\mathbf{y}}^{(\text{test})}$ is the predictions of the model on the test set, the mean square error is given by:

$$\text{MSE}_{\text{test}} = \frac{1}{N} \sum_{i=1}^{N} (\hat{\mathbf{y}}^{(\text{test})} - \mathbf{y}^{(\text{test})})_i^2.$$

Logistic Function

$$\sigma(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{W}^\top \mathbf{x} + b)}$$

$$p(y = 1|\mathbf{x}) = \sigma(\mathbf{x})$$

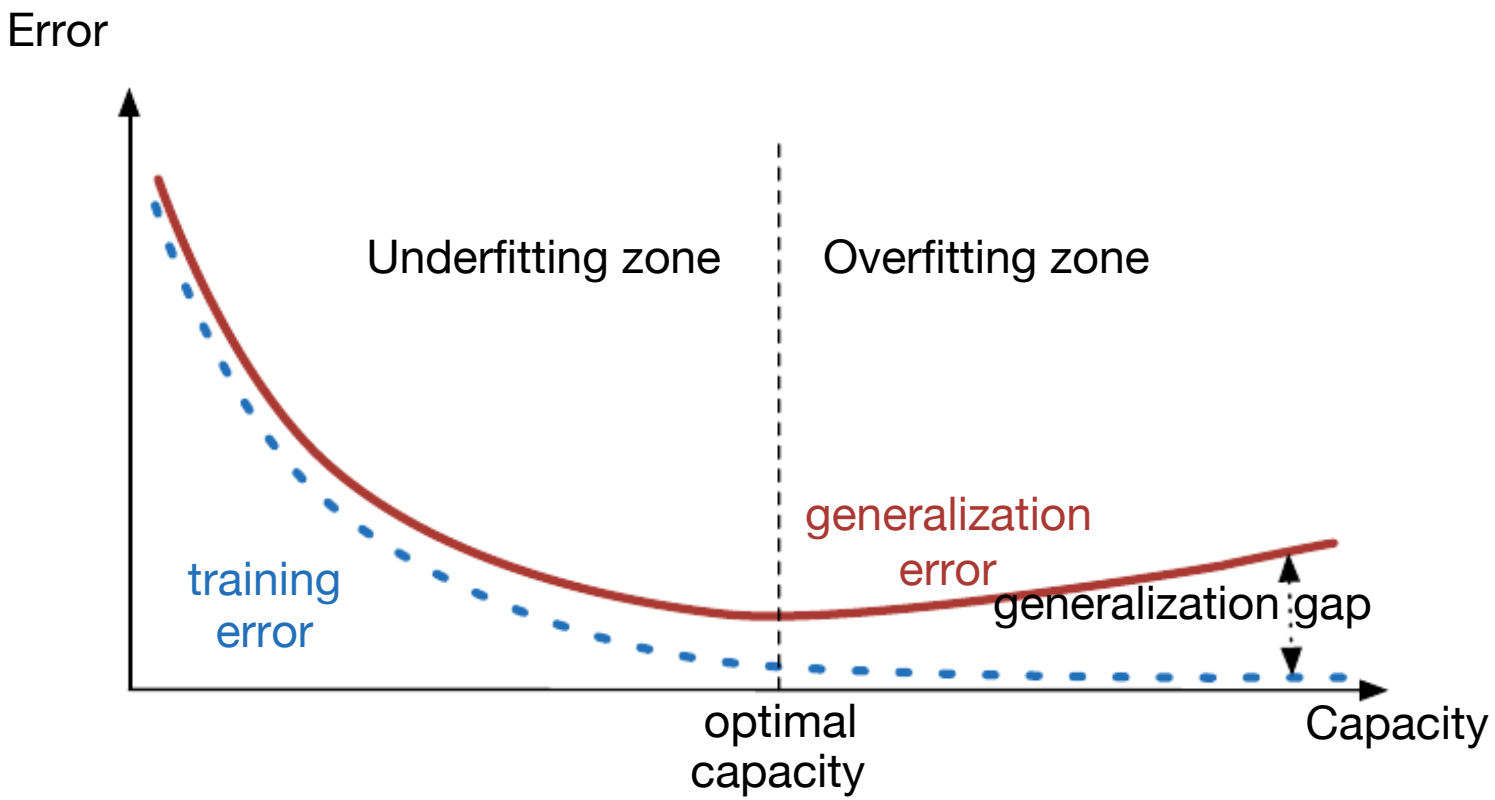
$$p(y = 0|\mathbf{x}) = 1 - \sigma(\mathbf{x})$$

Logistic Regression — Binary Classifier

$$J(\theta) = -\frac{1}{N}\sum_{i}\left(y^i \log(P(y=1|\mathbf{x}^i)) + (1-y^i)\log(p(y=0|\mathbf{x}^i))\right)$$

$$\theta^\star = \arg\min_{\theta} J(\theta)$$

Generalization, Capacity, Overfitting, Underfitting

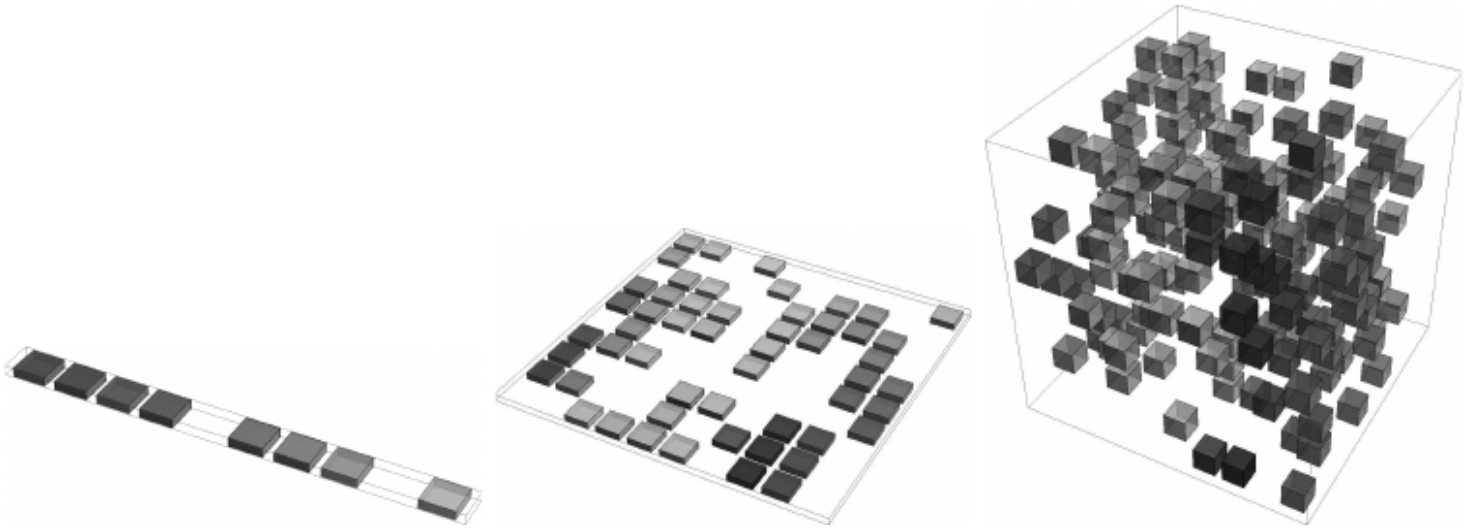


No Free Lunch Theorem

The no free lunch theorem for machine learning (Wolpert, 1996) states that, averaged over all possible data generating distributions, every classification algorithm has the same error rate when classifying previous unobserved points. In some sense, no ML algorithm is universally any better than any other.

Seek solution for some relevant distributions, NOT universal distribution.

Curse of Dimensionality

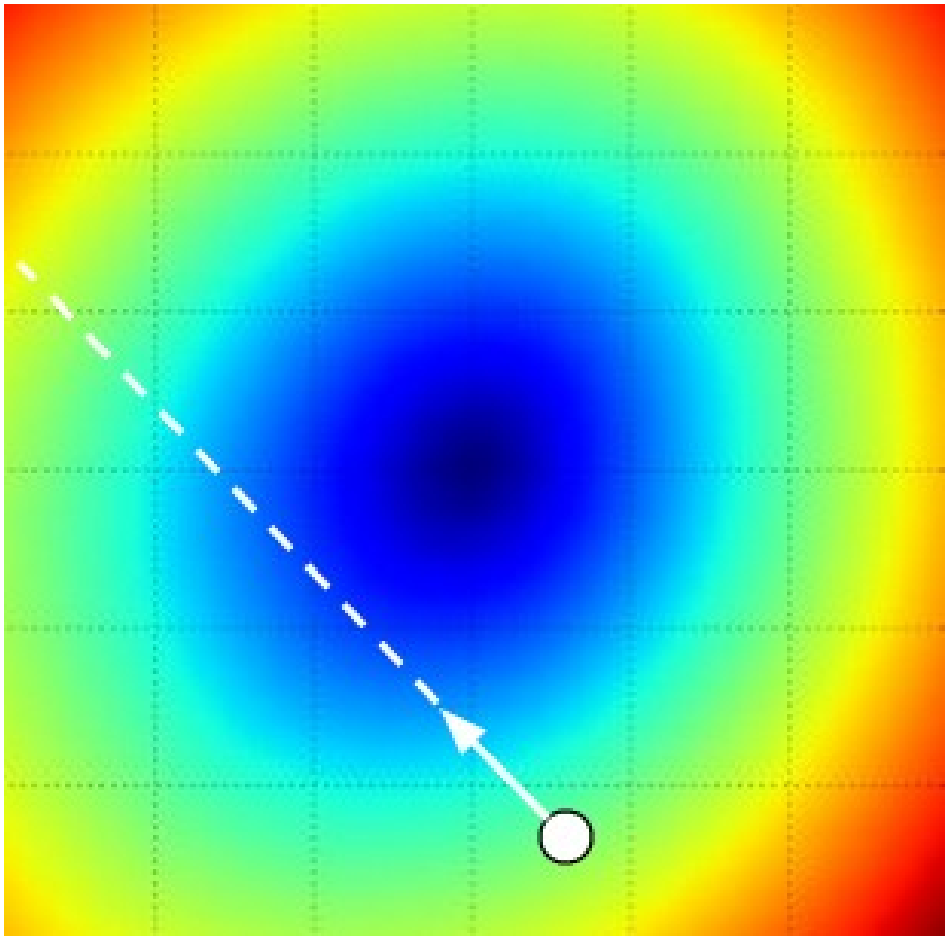


Q& A

Cost

- Target: to learn some optimal parameters θ
- Strategy: to minimize some cost L respect to θ
- Cost choice: cross-entropy cost, mean-squared error cost
- Solution: Gradient Descent!

Gradient optimization



Demo 1; Demo 2

Stochastic Gradient Descent (SGD)

$$\theta^* = \theta - \alpha \frac{\partial}{\partial \theta} J(\theta)$$

Variants: momentum SGD

$$V^* = \mu V + \alpha \nabla L(\theta)$$
$$\theta^* = \theta - V^*$$

Q&A

References I

Mitchell, T. M. (1997). *Machien learning*. New York: McGraw-Hill.

Wolpert, D. H. (1996, October). The lack of a priori distinctions between learning algorithms. *Neural Comput.*, *8*(7), 1341–1390.