Lecture 2: Machine Learning

Dr.-Ing. Sudchai Boonto, Assistant Professor

January 25, 2018

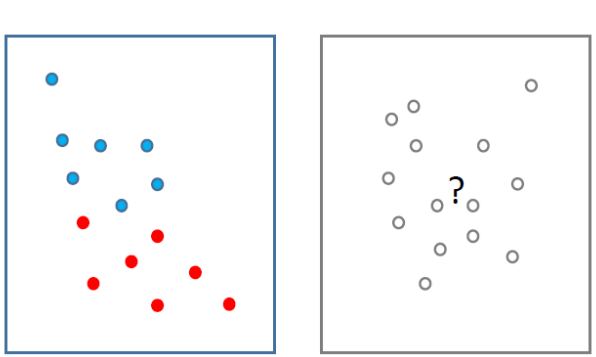
Department of Control System and Instrument Engineering, KMUTT

Types of Machine Learning

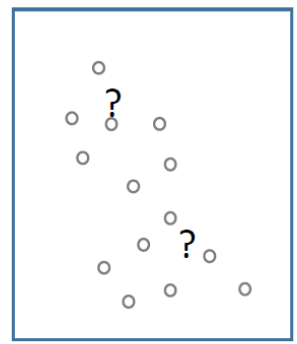
Machine Learning

Machine Learning (is the science (and art) of programming computers so they can learn from data.)

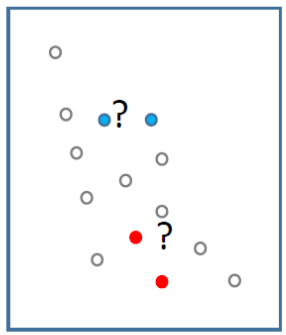
- It is a set of the technique used for the processing of large data by developing algorithms and set of rules to deliver the required results to the user.
- It is the technique used for developing automated machines on the basis of execution of algorithms and set of defined rules.
- You can build models from data. These models are mathematical models and can learn from and make predictions on data.
- There are several subfields of machine learning



Supervised Learning



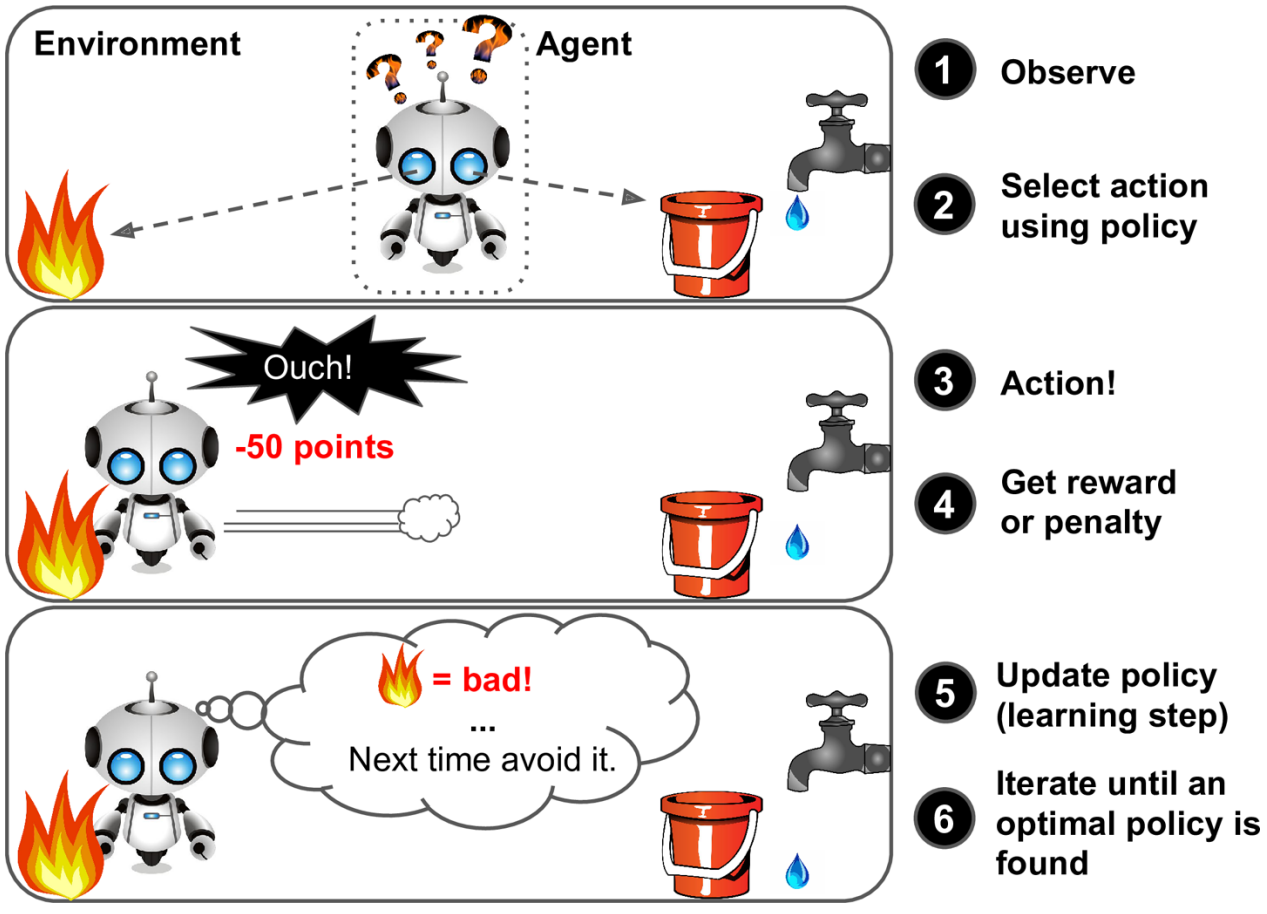
Unsupervised Learning



Semi - supervised Learning

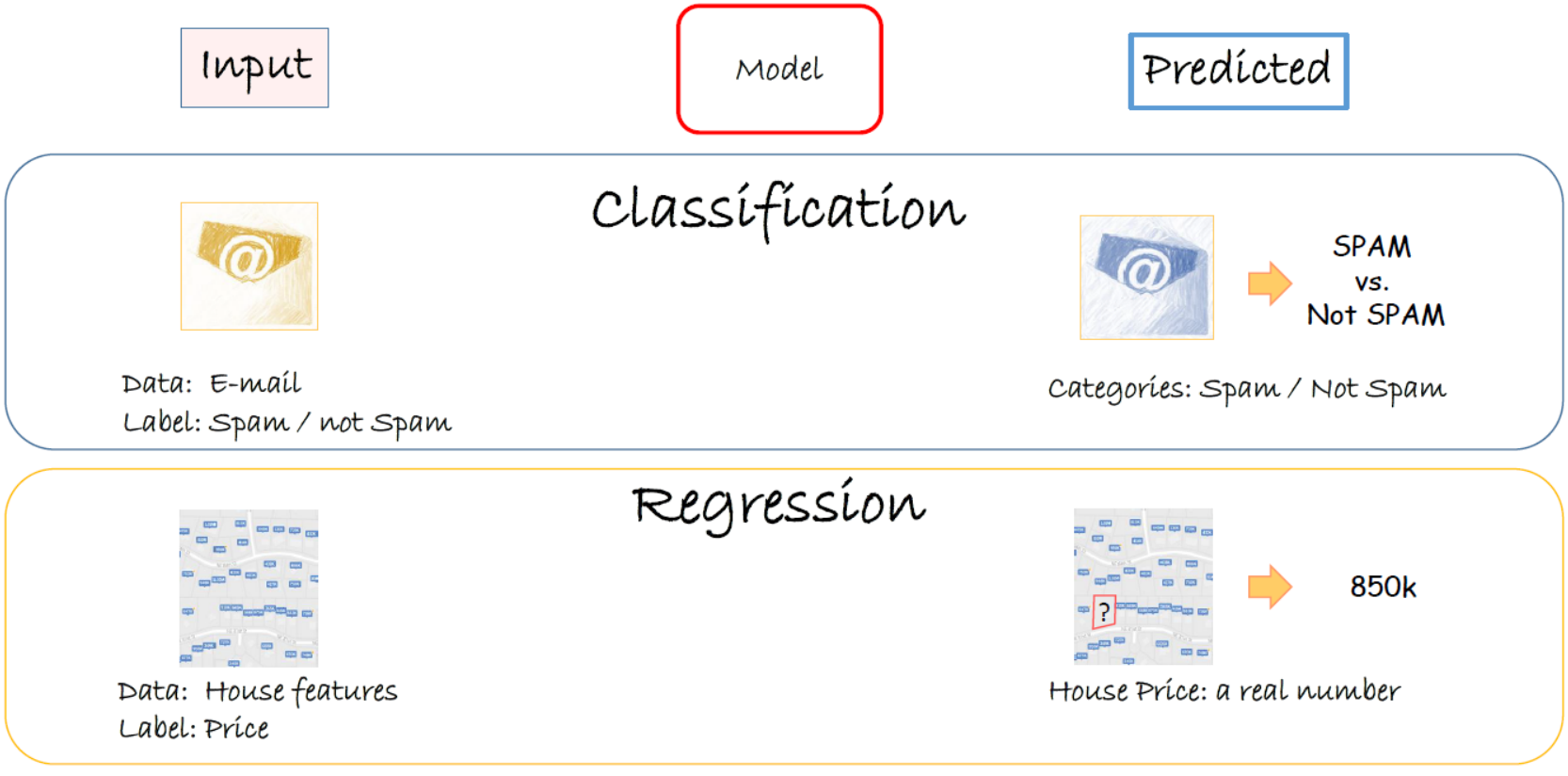
example from Deep Learning EdX

Reinforcement Learning



from "Hands-On Machine Learning with Scikit-Learn & TensorFlow", Aurélien Géron

Machine Learning: Supervised Learning



example from Deep Learning EdX

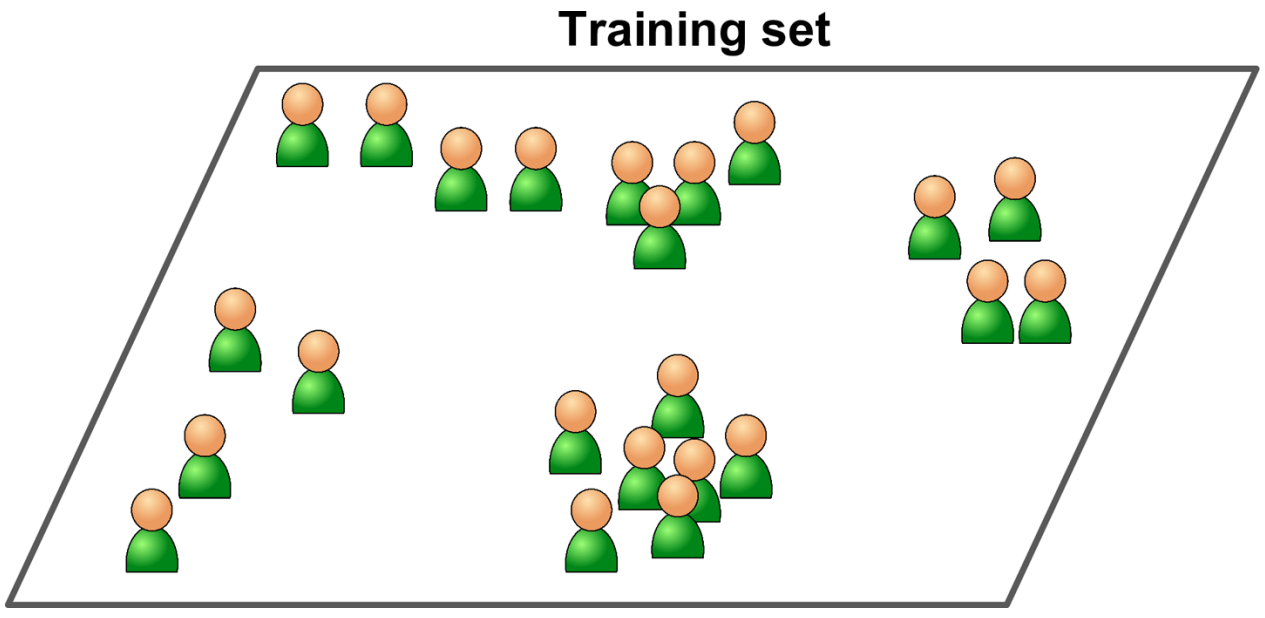
Machine Learning: Supervised Learning

The most important supervised learning algorithms:

- k-Nearest Neighbors
- Linear Regression
- Logistic Regression
- Support Vector Machines (SVMs)
- Decision Trees and Random Forests
- Neural networks

Machine Learning: Unsupervised Learning

Unsupervised learning: the training data is unlabeled. The system tries to learn without a teacher.



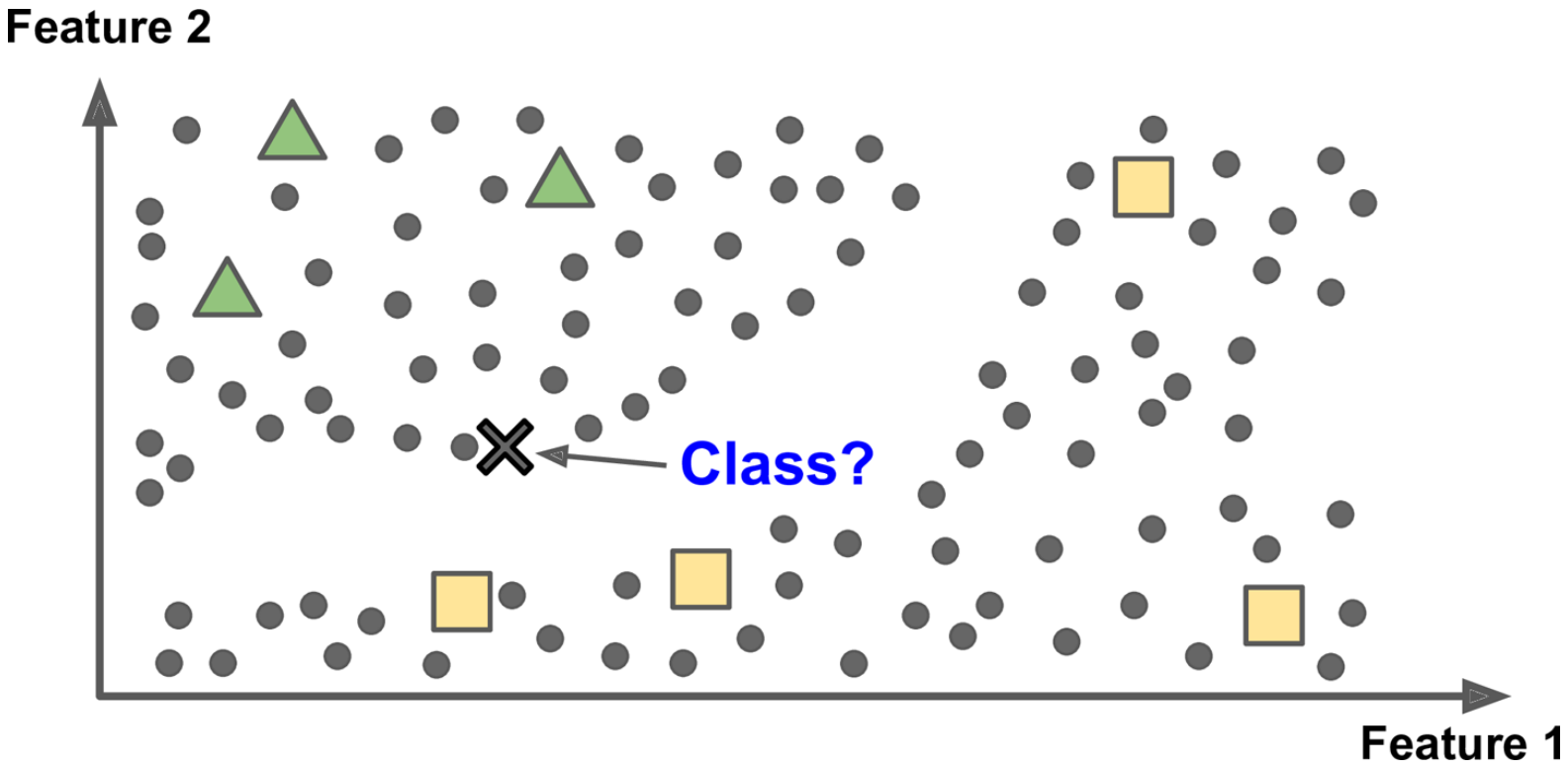
from ”Hands-On Machine Learning with Scikit-Learn & TensorFlow”, Aurélien Géron

Machine Learning: Unsupervised Learning

The most important unsupervised learning algorithms:

- Clustering
  - k-Means
  - Hierarchical Cluster Analysis (HCA)
  - Expectation Maximization
- Visualization and dimensionality reduction
  - Principal Component Analysis (PCA)
  - Kernel PCA
  - Locally-Linear Embedding (LLE)
  - t-distributed Stochastic Neighbor Embedding (t-SNE)
- Association rule learning
  - Apriori
  - Eclat

Machine Learning: Semisupervised Learning

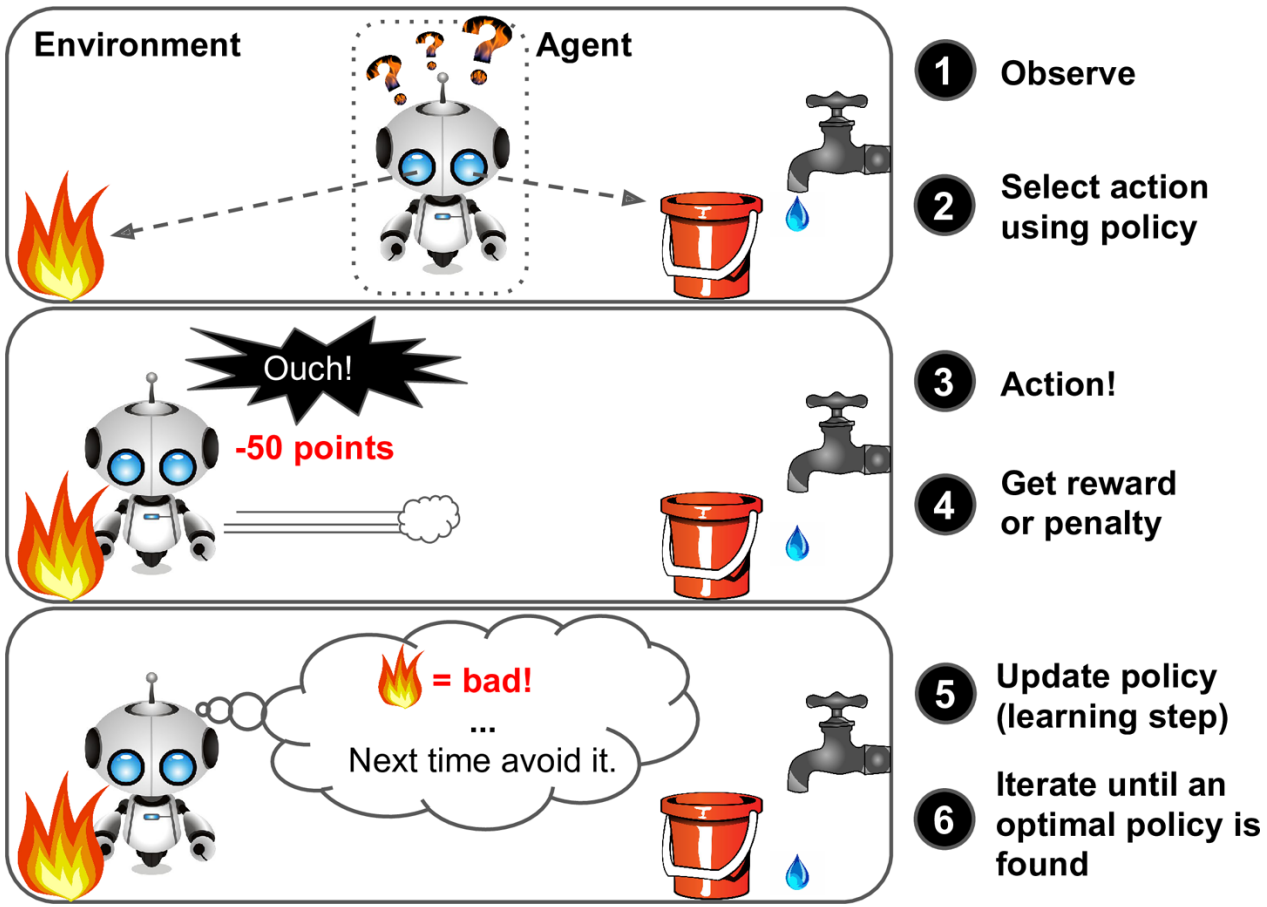


- Some algorithms can deal with partially labeled training data, usually a lot of unlabeled data and a little bit of labeled data. This is called *semisupervised learning*.
- For example the machine learning using in Google Photos.

from ”Hands-On Machine Learning with Scikit-Learn & TensorFlow”, Aurélien Géron

Machine Learning: Reinforcement Learning

The learning system, called an *agent*, can observe the environment, select and perform actions, and get *rewards* in return (or *penalties* in the form of negative reward). It must learn by itself what is the best strategy, called a *policy*.



from "Hands-On Machine Learning with Scikit-Learn & TensorFlow", Aurélien Géron

Batch and Online Learning

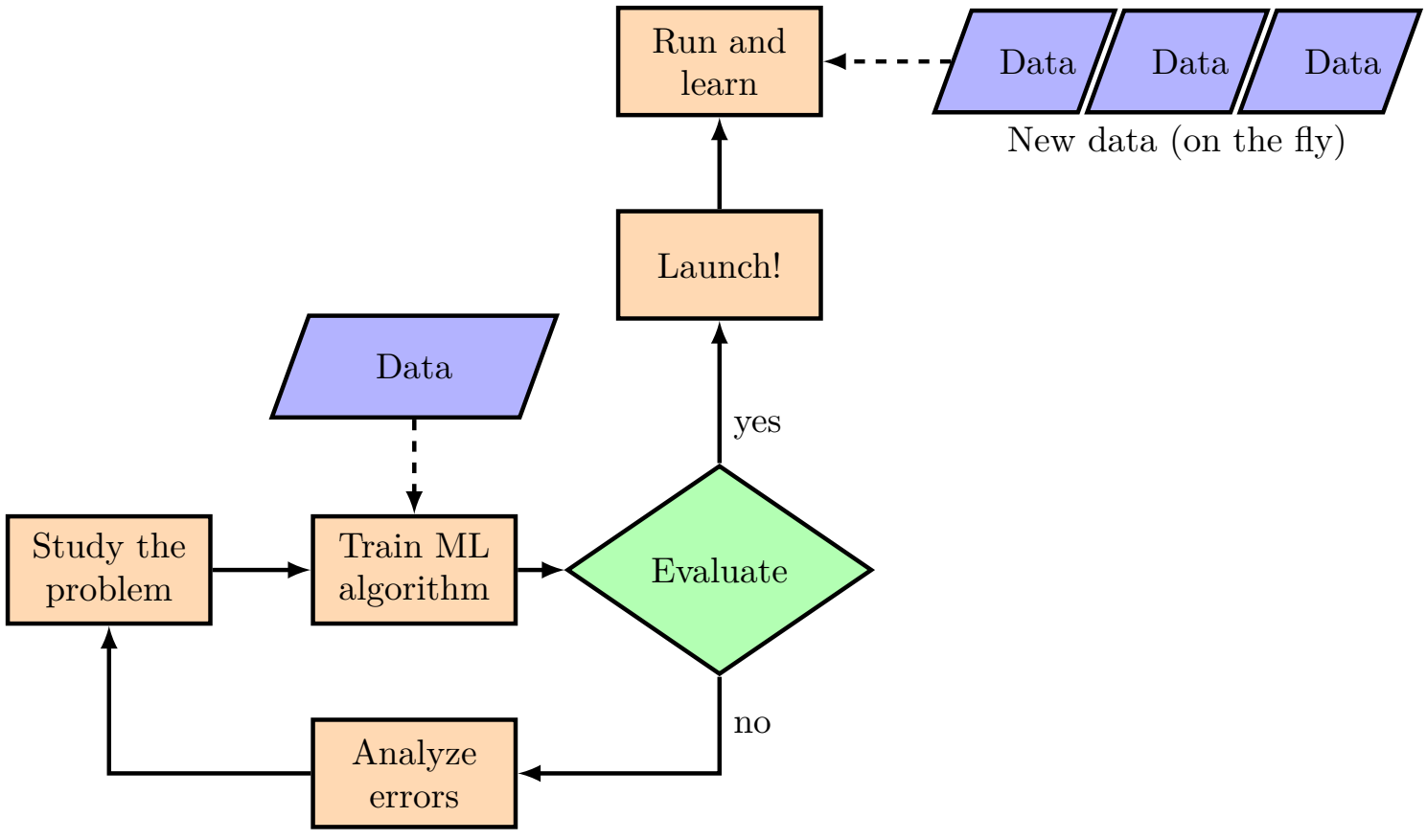
Batch Learning

In *batch learning*, the system is incapable of learning incrementally: it must be trained using all the available data.

- This will general take a lot of time and computing resource, so it typically done offline.
- First the system is trained, and then it is launched into production and runs without learning anymore.
- This is called *offline learning*
- If you wan a batch learning system to know about new data, you need to train a new version of the system from scratch on the full dataset.
- You need to stop the old system and replace it with the new machine.
- Training on the full set of data requires a lot of computing resources (CPU, memory space, disk space, disk I/O, network I/O, etc).
- If you have a lot of data and you automate your system to train from scratch every day, it will end up costing you a lot of money.

Online Learning

In *online learning*, you train the system incrementally by feeding it data instances sequentially, either individually or by small groups called *mini-batches*. Each learning step is fast and cheap, so the system can learn about new data on the fly, as it arrives.



Online Learning

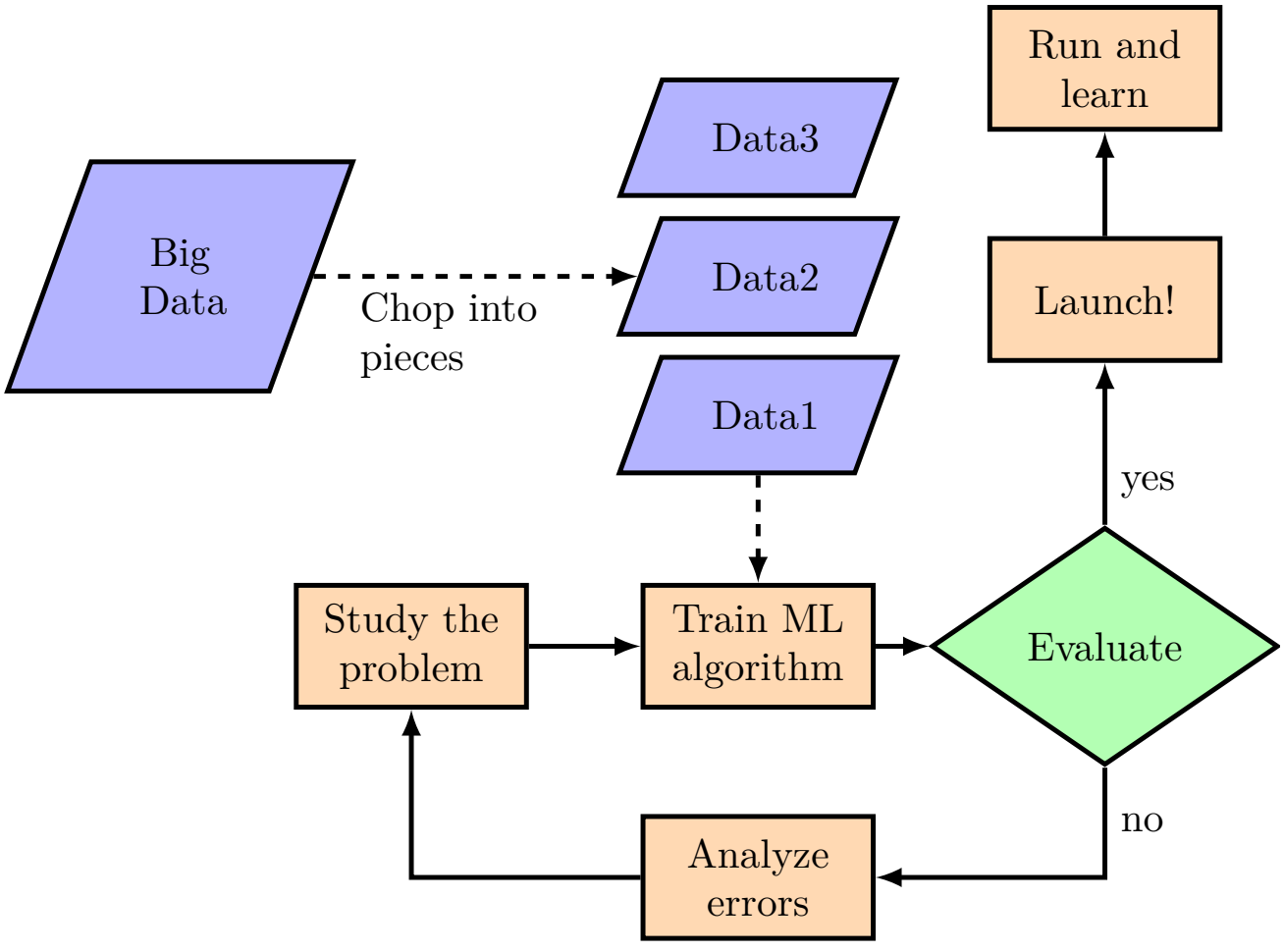
The online learning is good for:

- It is good for systems that receive data as a continuous flow (e.g., stock prices) and need to adapt to change rapidly or autonomously.
- a good option for limited computing resources: once an online learning system has learned about new data instances, it does not need them anymore, so you can discard them. This can save a huge amount of space.

Online learning algorithms can also be used to train systems on huge datasets that cannot fit in one machines's main memory (*out-of-core* learning)

- The algorithm loads part of the data, runs a training step on the data, and repeats the process until it has run on all of the data.

Online Learning



Online Learning

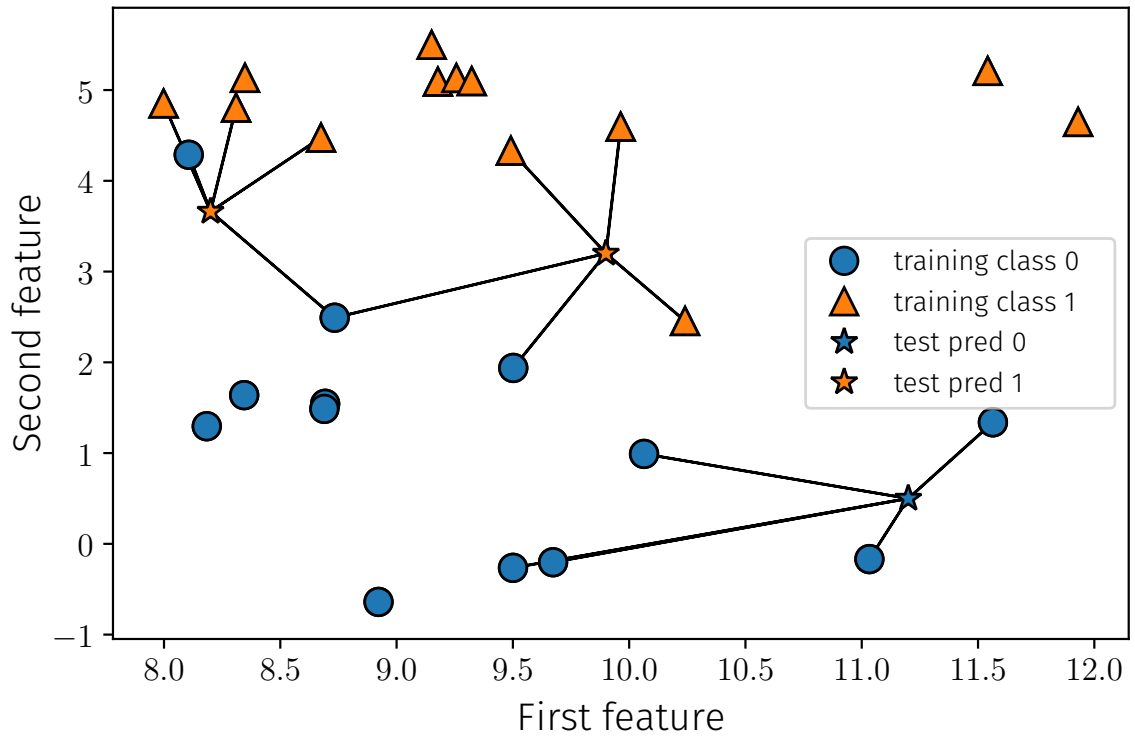
- One importance parameter of online learning systems is how fast they should adapt to changing data this is call *learning rate*.
- If you set a high learning rate, then your system will rapidly adapt to new data, but it will also ten to quickly forget the old data.
- If you set a low learning rate, the system will have more inertia; that is, it will learn more slowly, but it will also be less sensitive to noise in the new data or to sequences of nonrepresentative data points.

A big challenge with online learning is that if bad data is fed to the system, the system's performance will gradually decline.

Instance-Based Versus Model-Based Learning

Online Learning

Instance-based learning (sometimes called **memory-based learning**) is a family of learning algorithms that, instead of performing explicit generalization, compares new problem instances with instances seen in training, which have been stared in memory.



Example of instance-based learning algorithm are the k-**nearest neighbor, kernel machines**, and **RBF networks**.

Model-Based Learning

We have data of the form

$$\mathbb{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

where $x \in \mathbb{R}^n$ and $y \in \mathbb{R}$
Our task is to generate a computational procedure that implements the function $f : x \to y$. We have *real valued prediction*.

- Assuming that we have a data set generated from:

$$y(x) = 2 + x + 2x^2 + \epsilon,$$

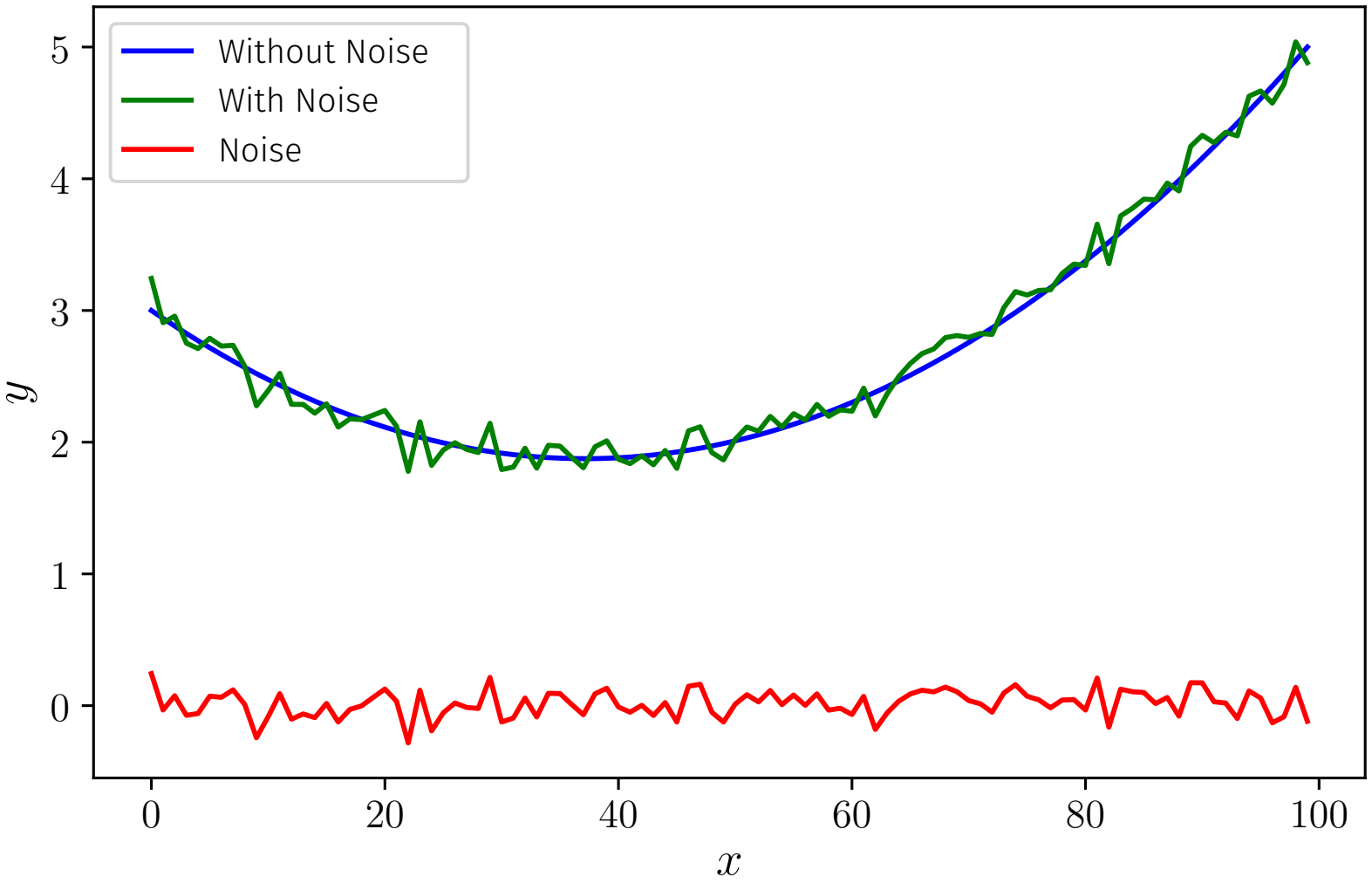
  where $\epsilon \approx \mathcal{N}(0, 0.1)$ is noise (random variation) from an normal distribution with 0 mean and 0.1 being the standard deviation.

- We want to find a model

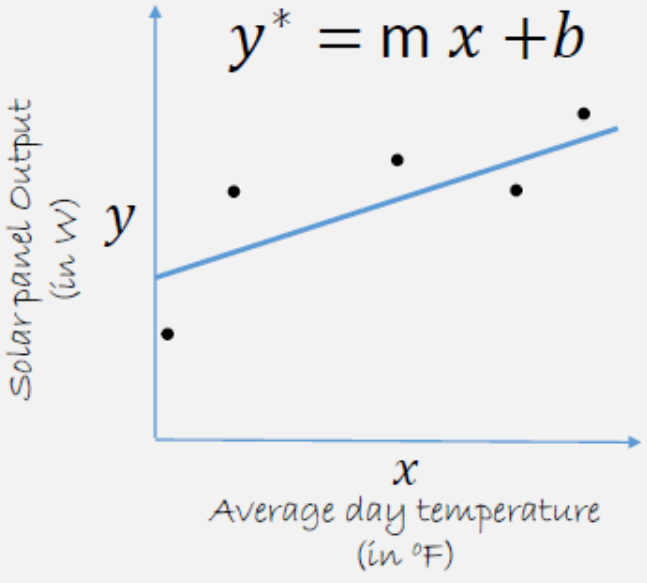
$$\hat{y}(x) = \theta_0 + \theta_1 x + \theta_2 x^2,$$

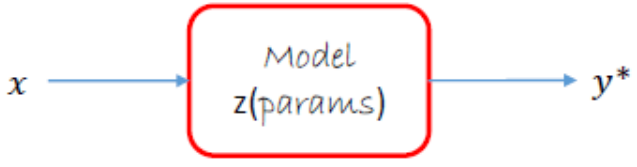
  where θ_i, $i = 0, 1, 2$ are parameters that we estimate.

Model-Based Learning



Model-Based Learning Recap





Input

x = Feature

Output

y = observed output (labels)

y^* = predicted output

Model function (z)

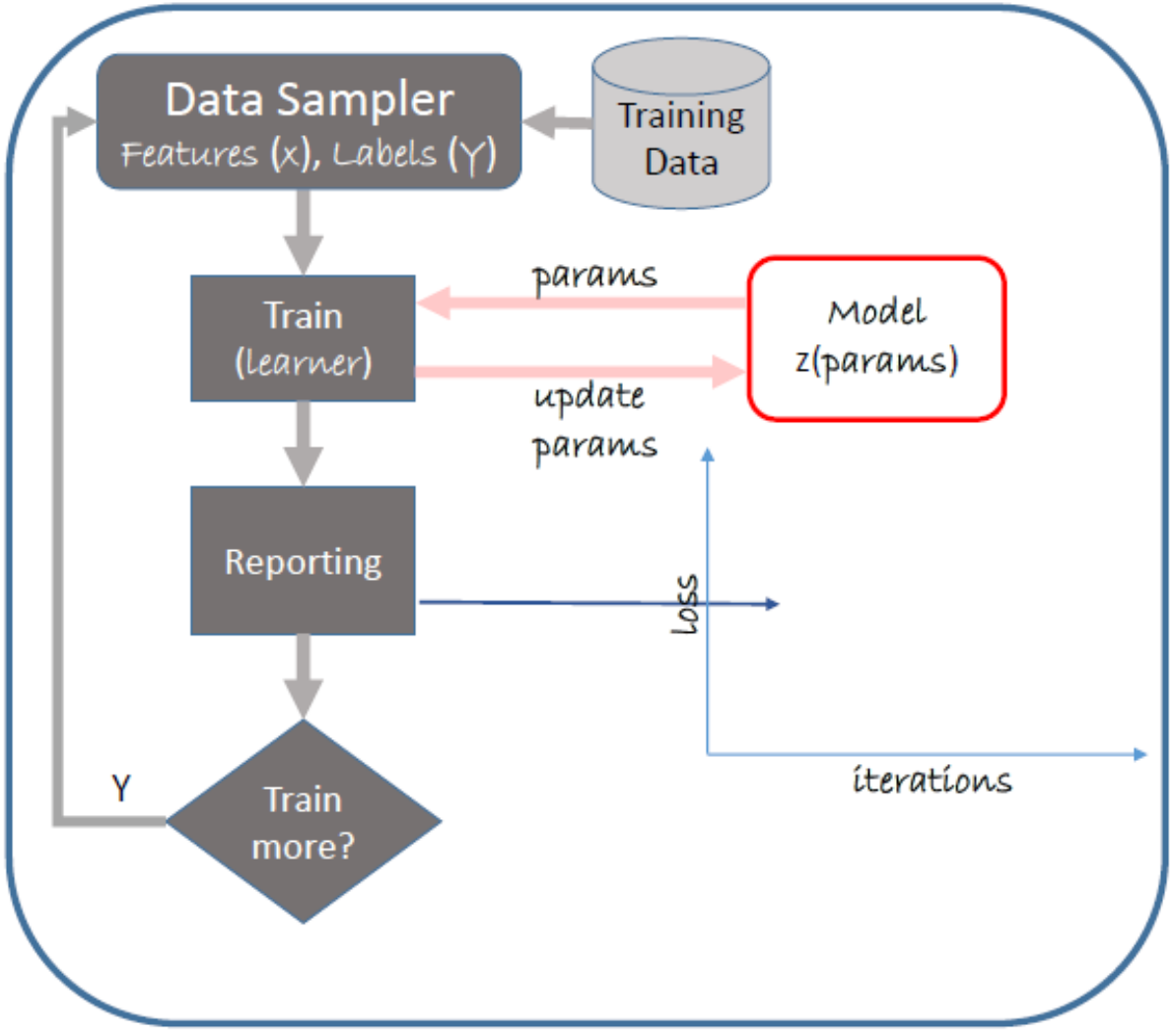
m: Slope

b: Intercept

are model parameters

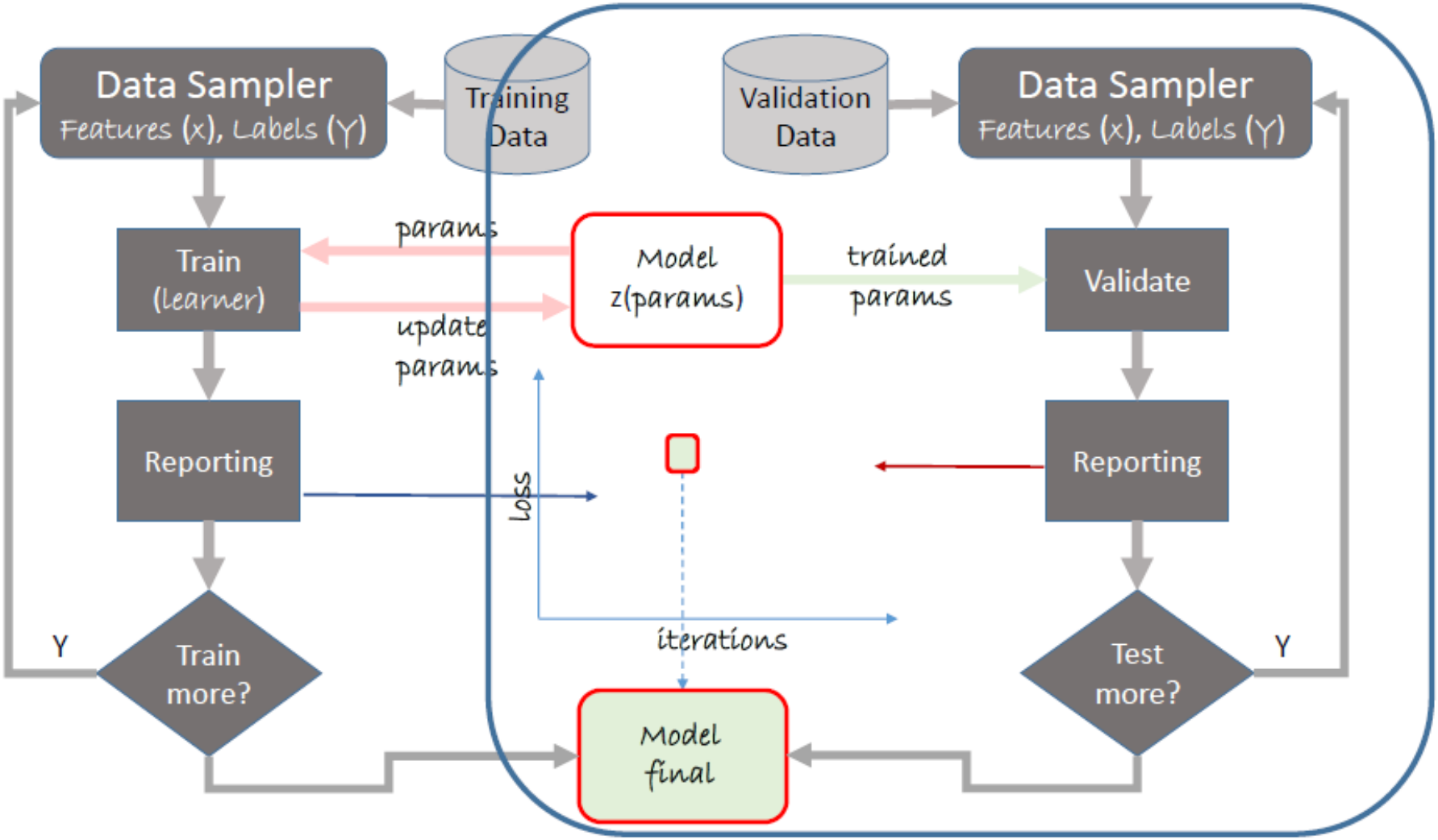
from Deep Learning EdX

Model-Based Learning Recap



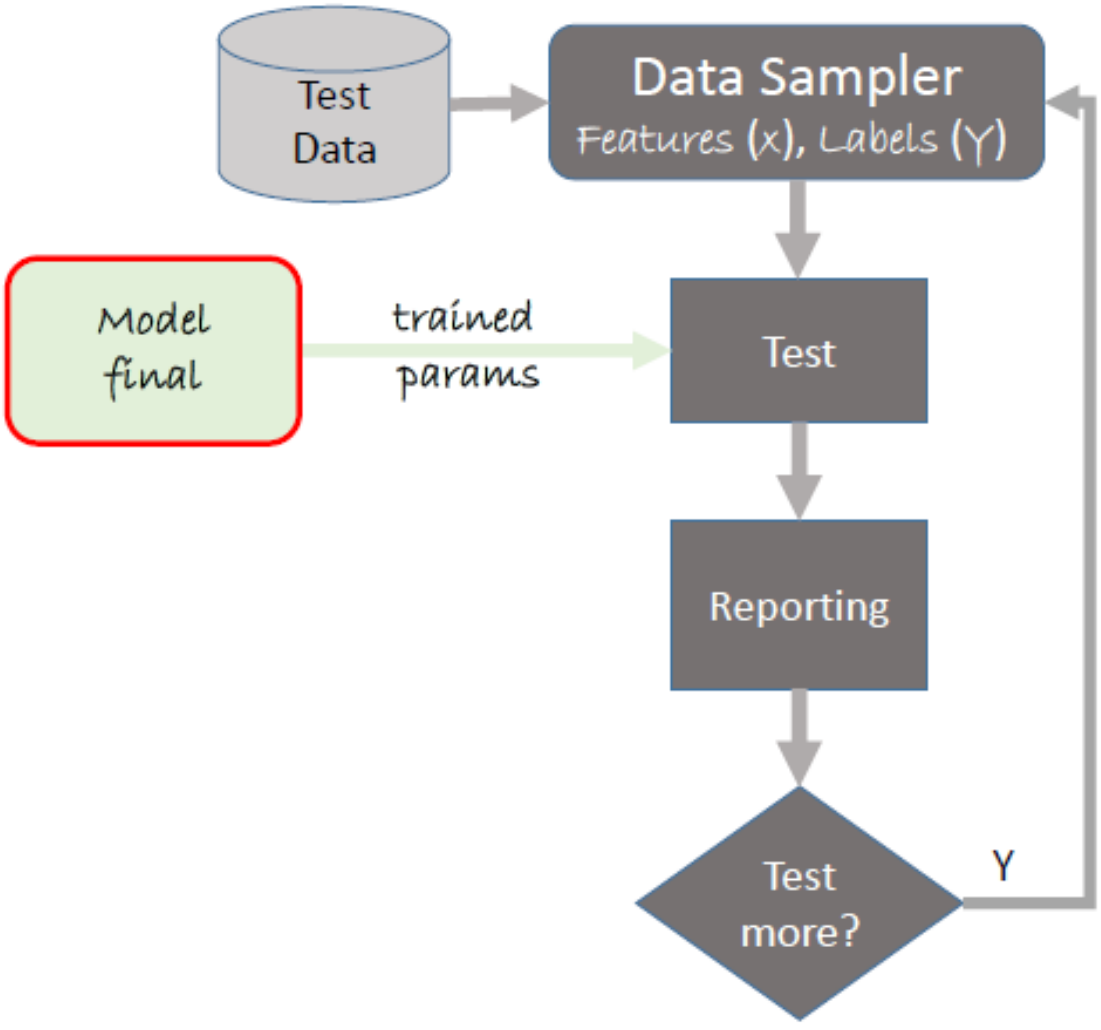
from Deep Learning EdX

Model-Based Learning Recap



from Deep Learning EdX

Model-Based Learning Recap

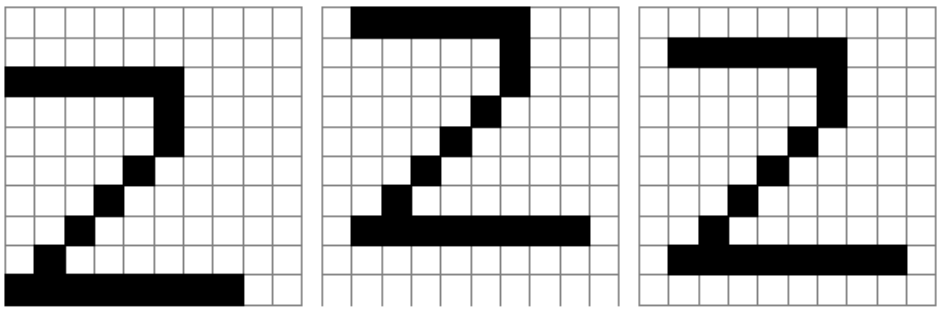


from Deep Learning EdX

Key Issue in Machine Learning

Training data is limited

- For a toddler to learn what an apple is, all it takes is for you to point to an apple and say “apple”. Now the child is able to recognize apples in all sorts of colors and shapes.
- Machine Learning is not quite there yet; it takes a lot of data for most Machine learning algorithms to work properly.
- If the classifier just memorizes the training data, it may perform poorly on new data.
- **Generalization** is ability to extend accurate predictions to new data.
- If the training dataset is like this, can the classifier generalize?



Model Expressiveness and Overfitting

- A model with more weight parameters may fit training data better
- But since training data is limited, expressive model stand the risk of overfitting to peculiarities of the data

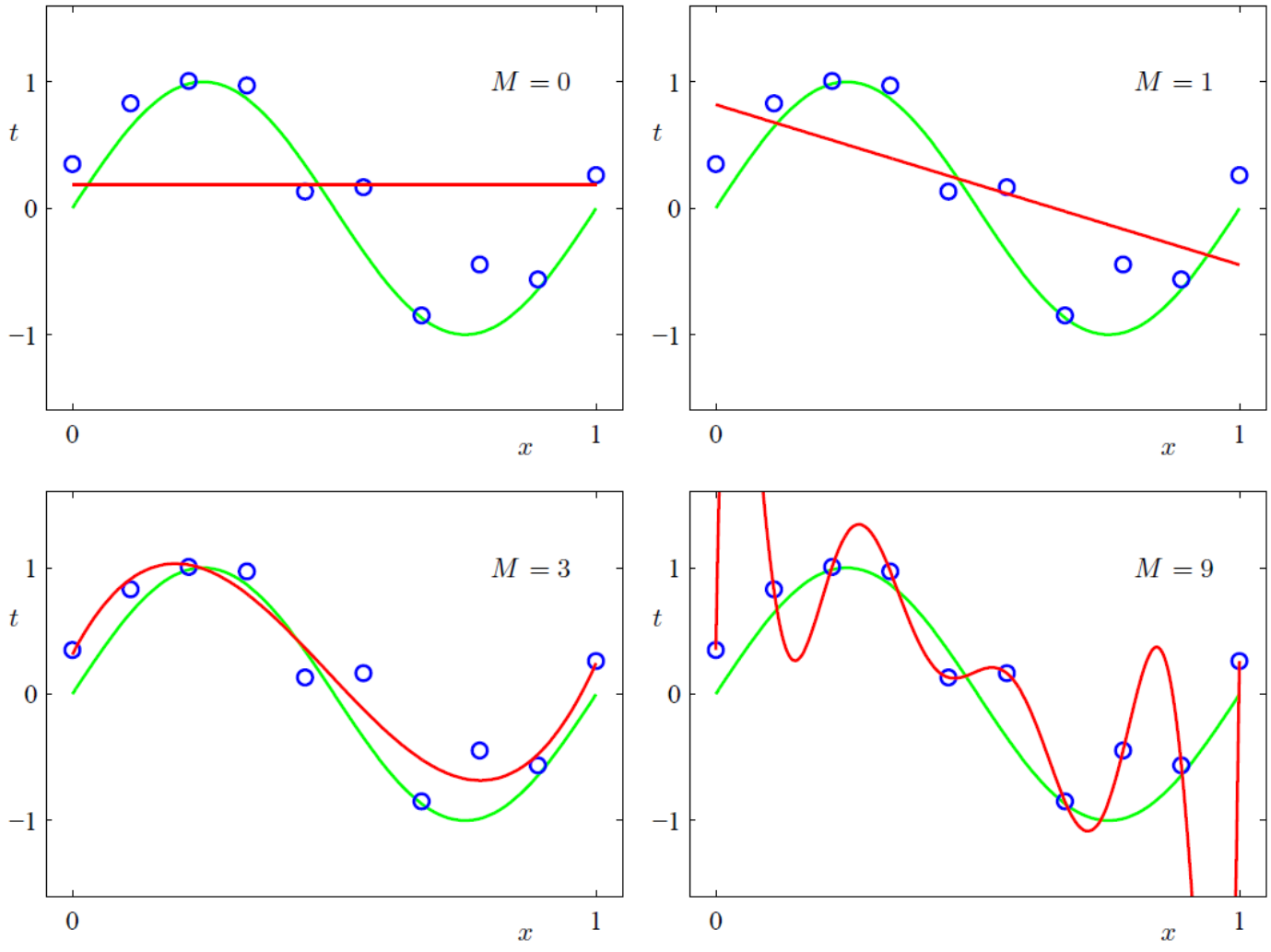
$$\text{Less Expressive Model} \iff \text{More Expressive Model}$$
$$\text{(fewer weights)} \qquad\qquad \text{(more weights)}$$
$$\text{Underfit training data} \iff \text{Overfit training data}$$

- Fitting the training data (blue points: x_n) with a polynomial model: $f(x) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M$ under squared error objective

$$V(x) = \frac{1}{2} \sum_{i=0}^{n} \left(f(x_i) - t_i \right)^2 ,$$

where t_i are target values.

Model Expressiveness and Overfitting



from Bishop, C “Pattern Recognition and Machine Learning”, Springer, 2006

Basic Problem Setup in Machine Learning

Model Expressiveness and Overfitting

- **Training Data:** a set of $\left(x^{(m)}, y^{(m)}\right)$, $m = 1, 2, \ldots, M$ pairs, where input $x^{(m)} \in \mathbb{R}^d$ and output $y^{(m)} = \{0, 1\}$
  - e.g. $x =$ vectorized image pixels, $y = 2$ or non-2
- **Goal:** Learn function $f : x \to y$ to predicts correctly on new inputs x.
  - **Step 1:** Choose a function model family:
    - e.g. logistic regression, support vector machines,neural networks
  - **Step 2:** Optimize parameters w on the Training Data
    - e.g. minimize loss function

$$\min_{w} \sum_{m=1}^{M} \left(f_w\left(x^{(m)}\right) - y^{(m)}\right)^2$$

Reference

1. Aurélien, Géron, ”*Hands-On Machine Learning with Scikit-Learn & TensorFlow*”, O’reilly, 2017
2. S.P.K. Spielberg, R. B. Gopaluni, P. D. Loewen, ”Deep Reinforcement Learning Approaches for Process Control”, 2017 6th International Symposium on Advanced control of Industrial Processes (AdCONIP), May 28-31, 2017, Taipei, Taiwan