CSCI 651

Algorithm Concepts

Selected Topic: Machine Learning Introduction

Zhida Li, Ph.D.
College of Engineering and Computing Sciences

NEW YORK INSTITUTE OF TECHNOLOGY

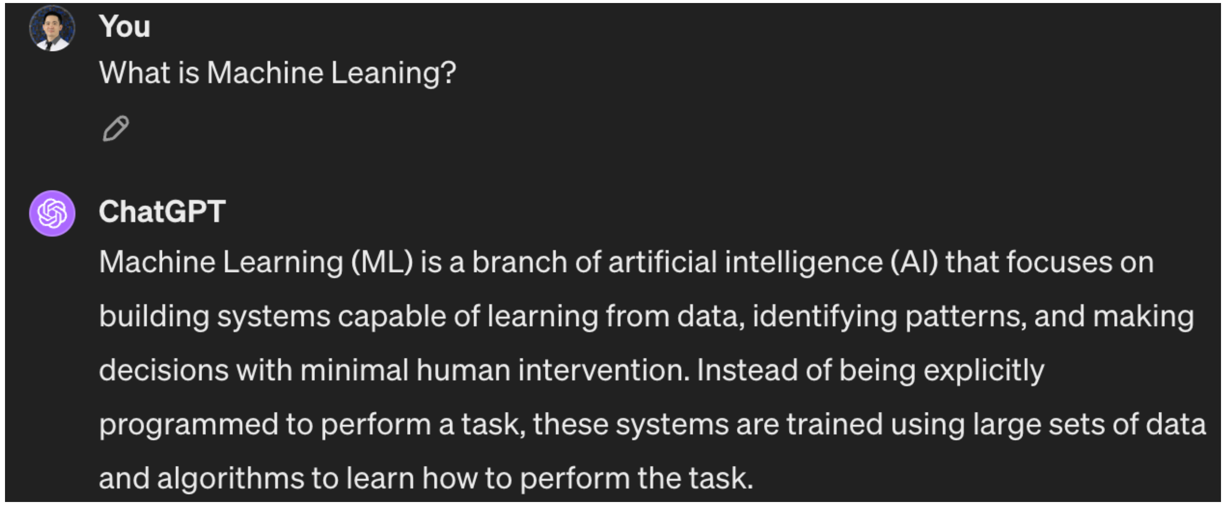
What is Machine Leaning (ML)?

- Automatic discovery of regularities in data
- Algorithms and techniques that allow computers to “learn”.The major focus is to extract information from data automatically, by computational and statistical methods

- Applications: natural language processing, computer vision, recommender systems, bioinformatics, stock market analysis, game playing, and intrusion detection systems

What is Machine Leaning?

- ChatGPT:

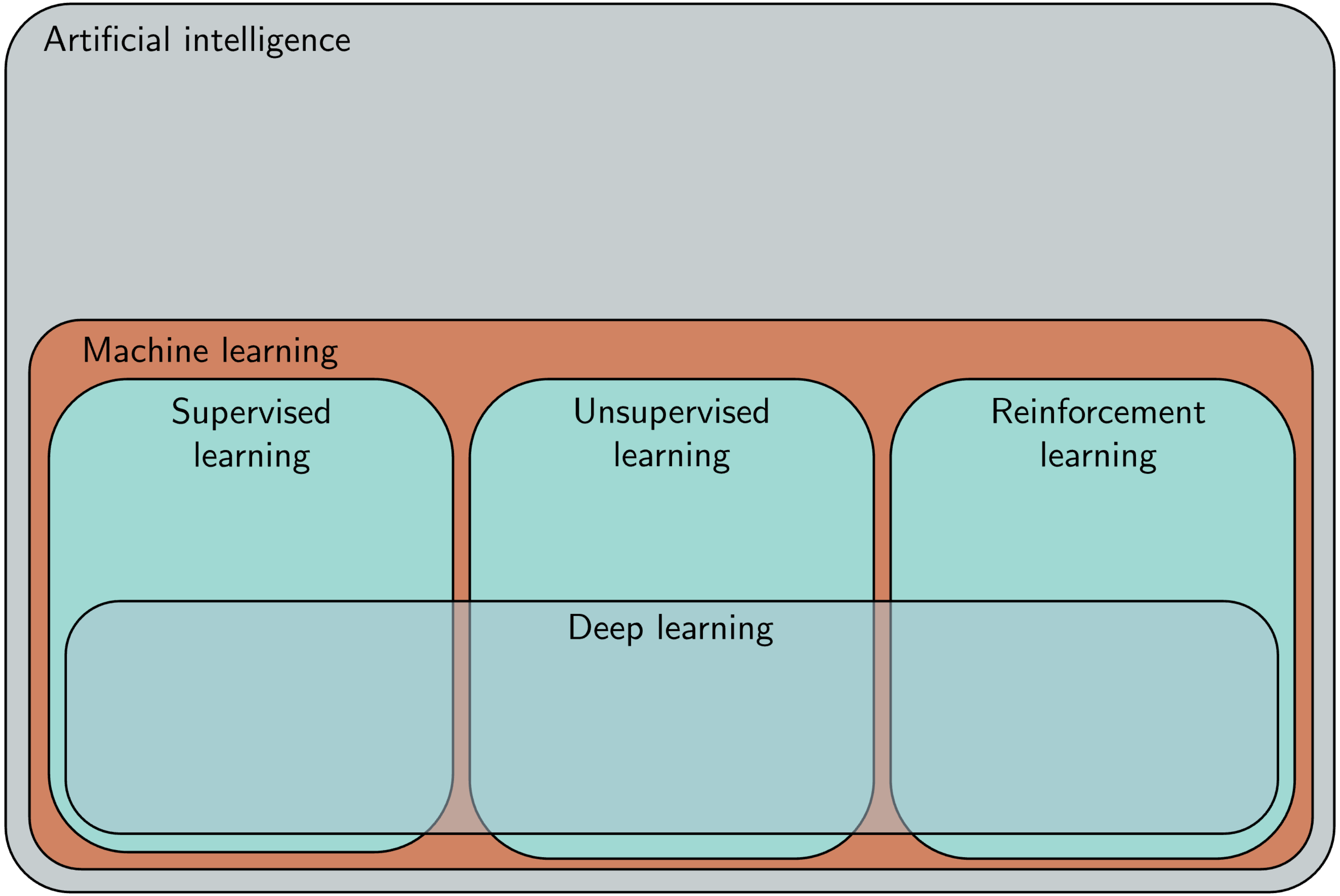

You
What is Machine Leaning?

ChatGPT
Machine Learning (ML) is a branch of artificial intelligence (AI) that focuses on building systems capable of learning from data, identifying patterns, and making decisions with minimal human intervention. Instead of being explicitly programmed to perform a task, these systems are trained using large sets of data and algorithms to learn how to perform the task.

Microsoft:

- ML is the process of using mathematical models of data to help a computer learn without direct instruction. It’s considered a subset of artificial intelligence (AI).
- ML uses algorithms to identify patterns within data, and those patterns are then used to create a data model that can make predictions.
- With increased data and experience, the results of machine learning are more accurate—much like how humans improve with more practice.

https://azure.microsoft.com/en-ca/resources/cloud-computing-dictionary/what-is-machine-learning-platform


Artificial intelligence
Machine learning
Supervised learning
Unsupervised learning
Reinforcement learning
Deep learning

Supervised learning

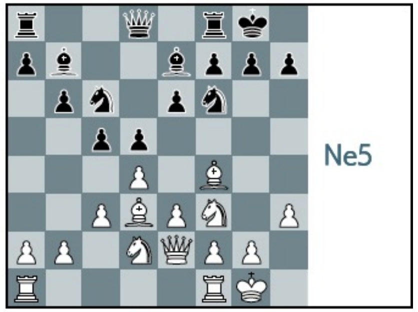
Bicycle

Apple

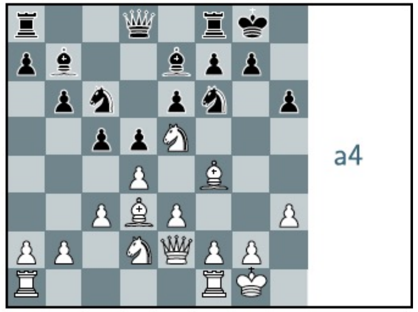
Aardvark

Unsupervised learning

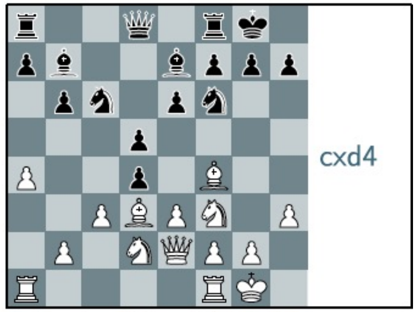
Reinforcement learning



Reward = 0



Reward = -1



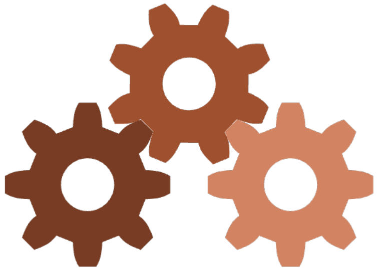
Reward = +1

Regression

Real world input → Model input → Model → Model output → Real world output

6000 square feet,
4 bedrooms,
previously sold for
$235K in 2005,
1 parking spot.

$$\begin{bmatrix} 6000 \\ 4 \\ 235 \\ 2005 \\ 1 \end{bmatrix}$$



Supervised learning model

$$[340]$$

Predicted price is $340k

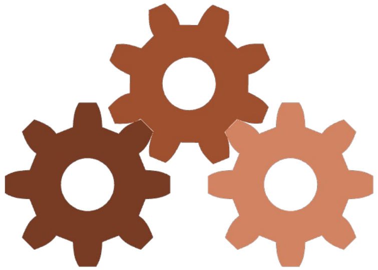
Image classification

Real world input

Model input

$$\begin{bmatrix} 124 \\ 140 \\ 156 \\ 128 \\ 142 \\ 157 \\ \vdots \end{bmatrix}$$

Model



Supervised learning model

Model output

$$\begin{bmatrix} 0.00 \\ 0.00 \\ 0.01 \\ 0.89 \\ 0.05 \\ 0.00 \\ \vdots \\ 0.01 \end{bmatrix}$$

Real world output

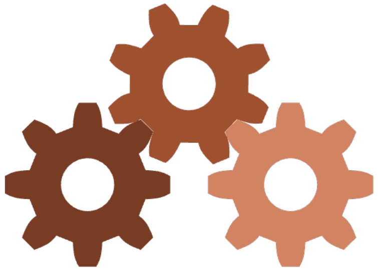
Aardvark
Apple
Bee
Bicycle
Bridge
Clown
⋮

Text classification

Real world input | Model input | Model | Model output | Real world output

"The steak was terrible, the salad was rotten, and the soup tasted like socks"

$$\begin{bmatrix} 8672 \\ 8194 \\ 9804 \\ 8634 \\ 8672 \\ \vdots \end{bmatrix}$$

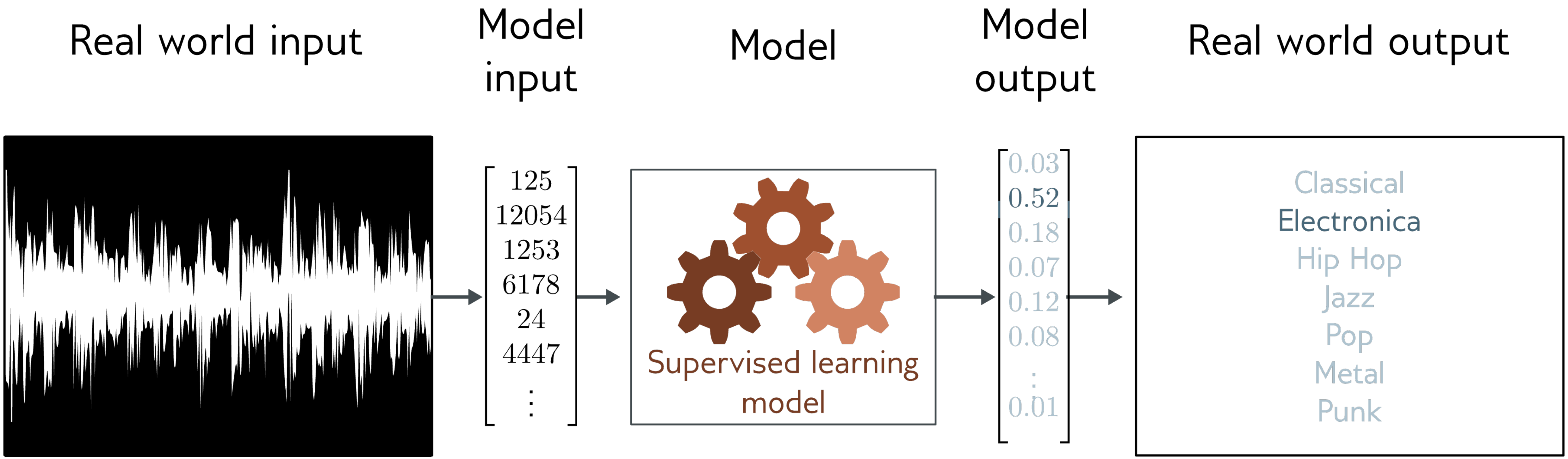


Supervised learning model

$$\begin{bmatrix} 0.02 \\ 0.98 \end{bmatrix}$$

Positive
Negative

Music genre classification

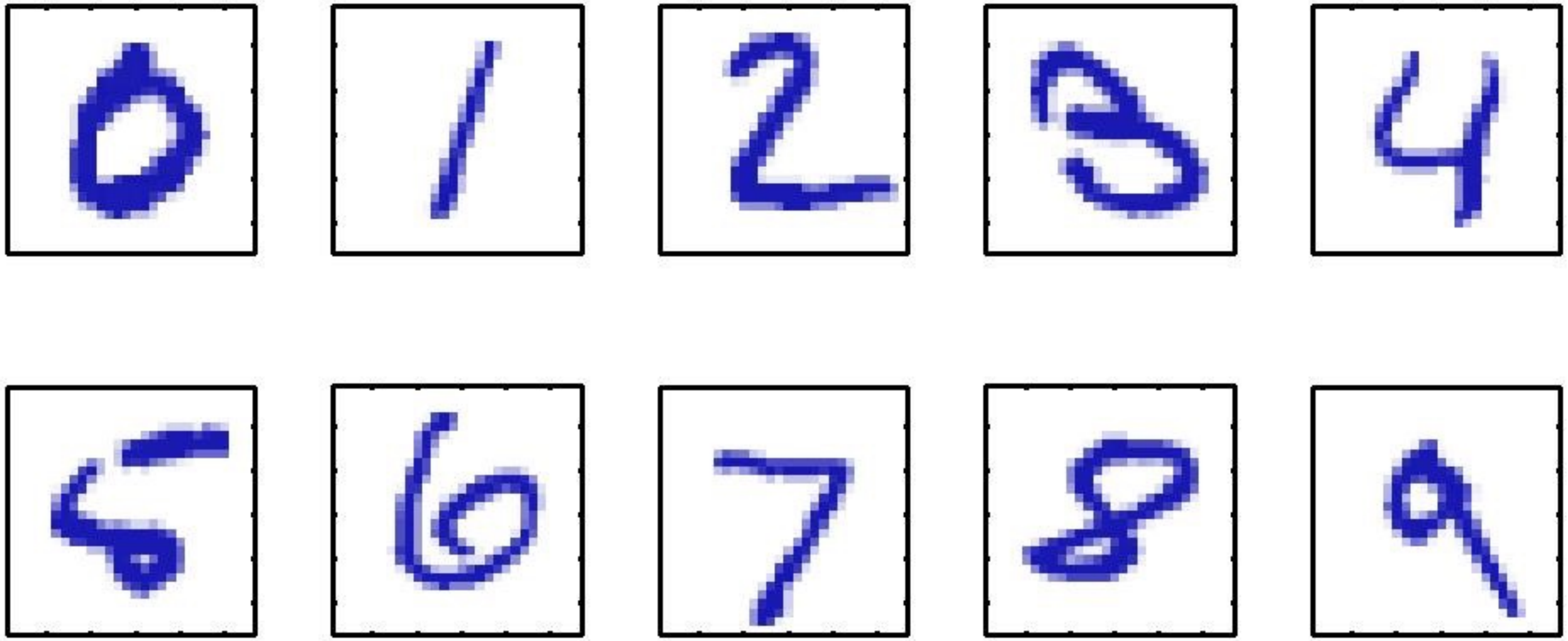


Supervised, Unsupervised, Reinforcement Learnings

- Supervised learning:
  - Define a mapping from input to output
  - Learn this mapping from paired input/output data examples
- Unsupervised learning:
  - Learning about a dataset without labels: e.g., clustering
  - Generative models can create examples: e.g., generative adversarial networks
  - Probabilistic generative models learn distribution over data: e.g., variational autoencoders
- Reinforcement learning:
  - A set of states; A set of actions; A set of rewards
  - Goal: take actions to change the state so that you receive rewards
  - You don’t receive any data – you have to explore the environment yourself to gather data as you go

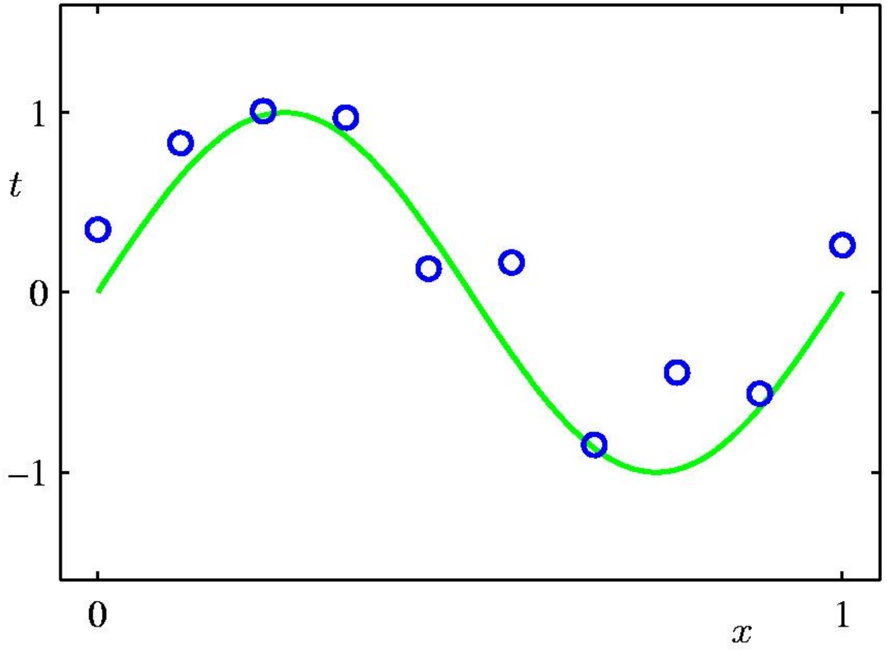
Example:
Handwritten Digit Recognition

- Example: 28 x 28 pixel image as vector ***x***
- Build ML model to identify digit {0,1,...,9} as output



Polynomial Curve Fitting

- Training set: N observations of x; $X = (x_1, ..., x_N)^T$
- Targets: N observations of t; $T = (t_1, ...,t_N)^T$
- Goal is generalization: predict x for some unseen t



Plot of training data (10 points). Green curve shows function sin(2 π x) used to generate data.

Polynomial Curve Fitting

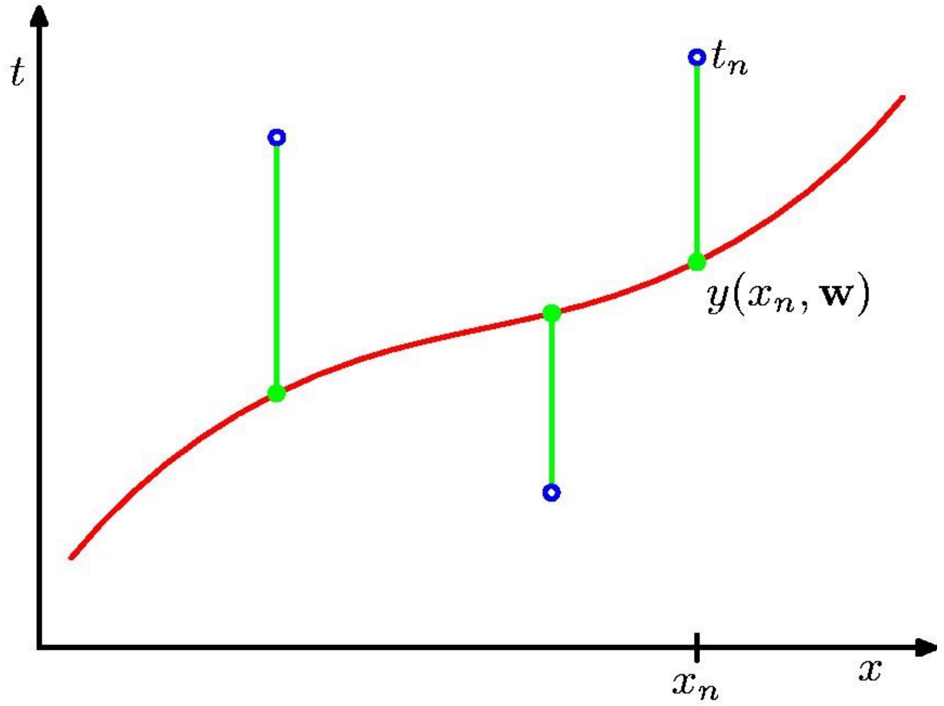
- Polynomial functional:

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

- Error Function: minimize (one half of) sum of the squares of the errors between predictions $y(x_n,w)$ for each point x_n and corresponding target values t_n

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

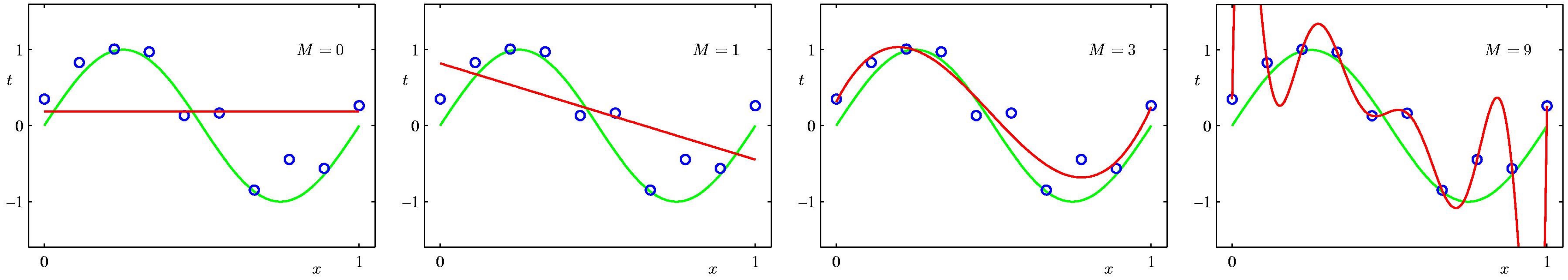
Sum-of-Squares Error Function



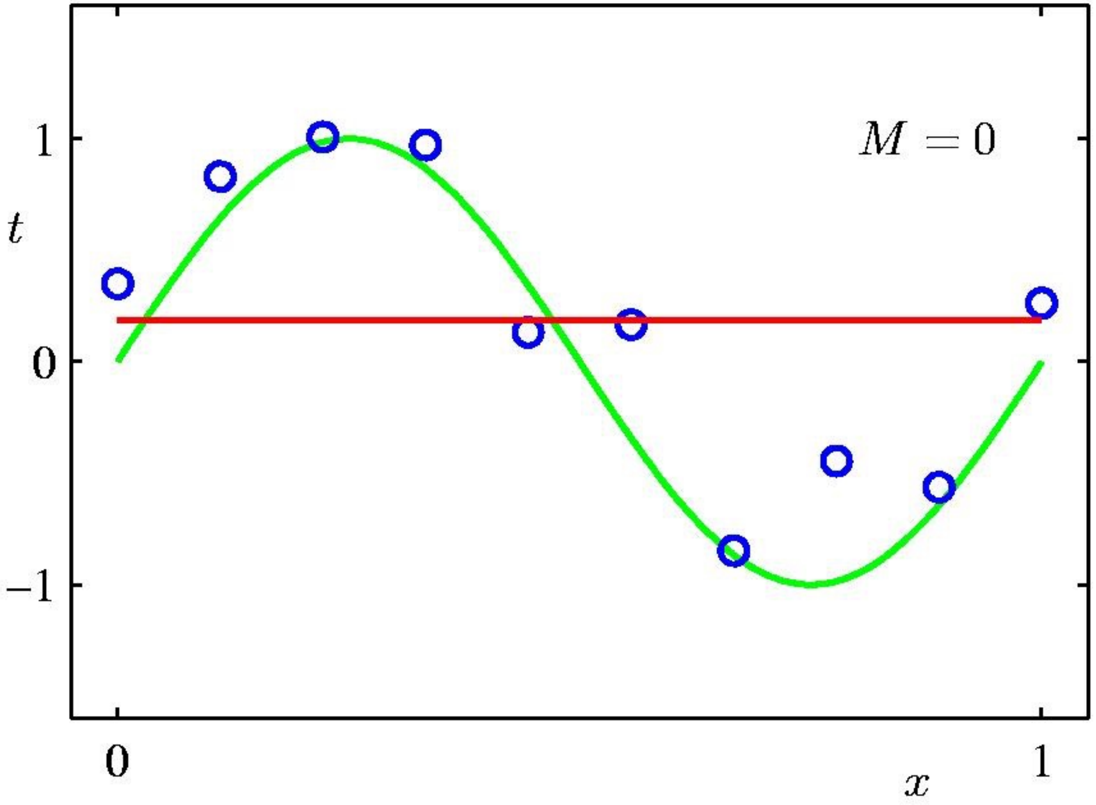
$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

Model Selection

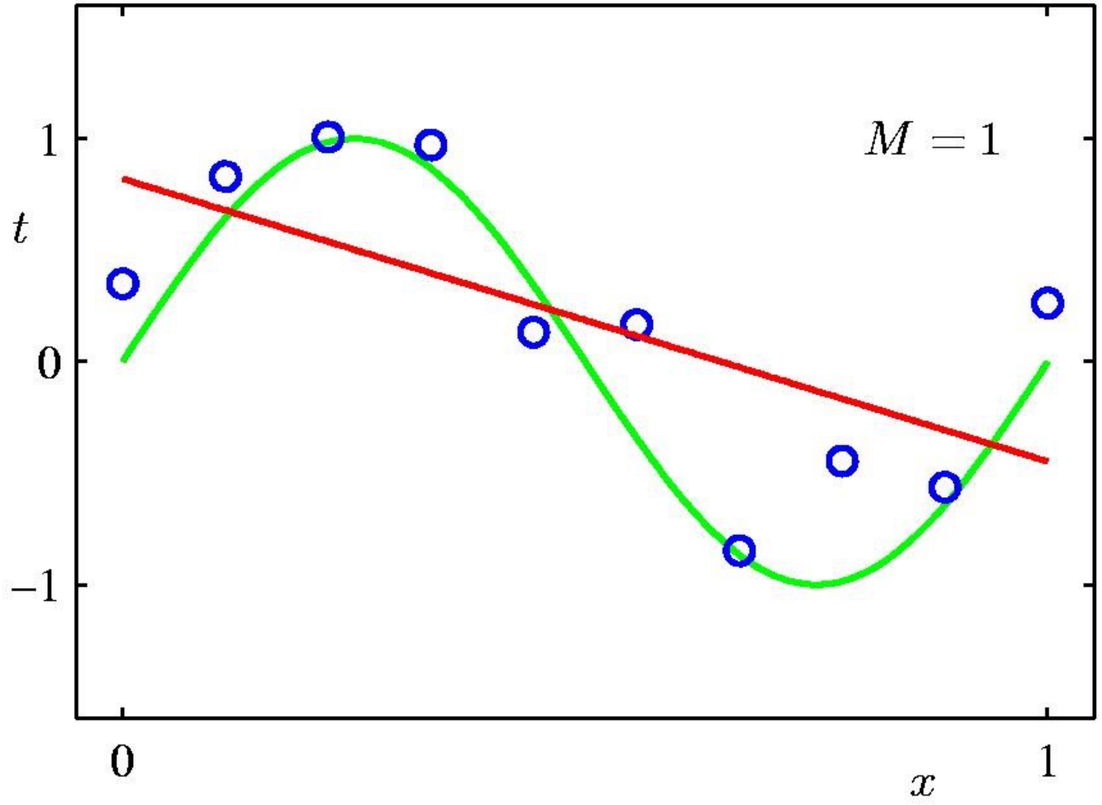
- What is the appropriate polynomial order?



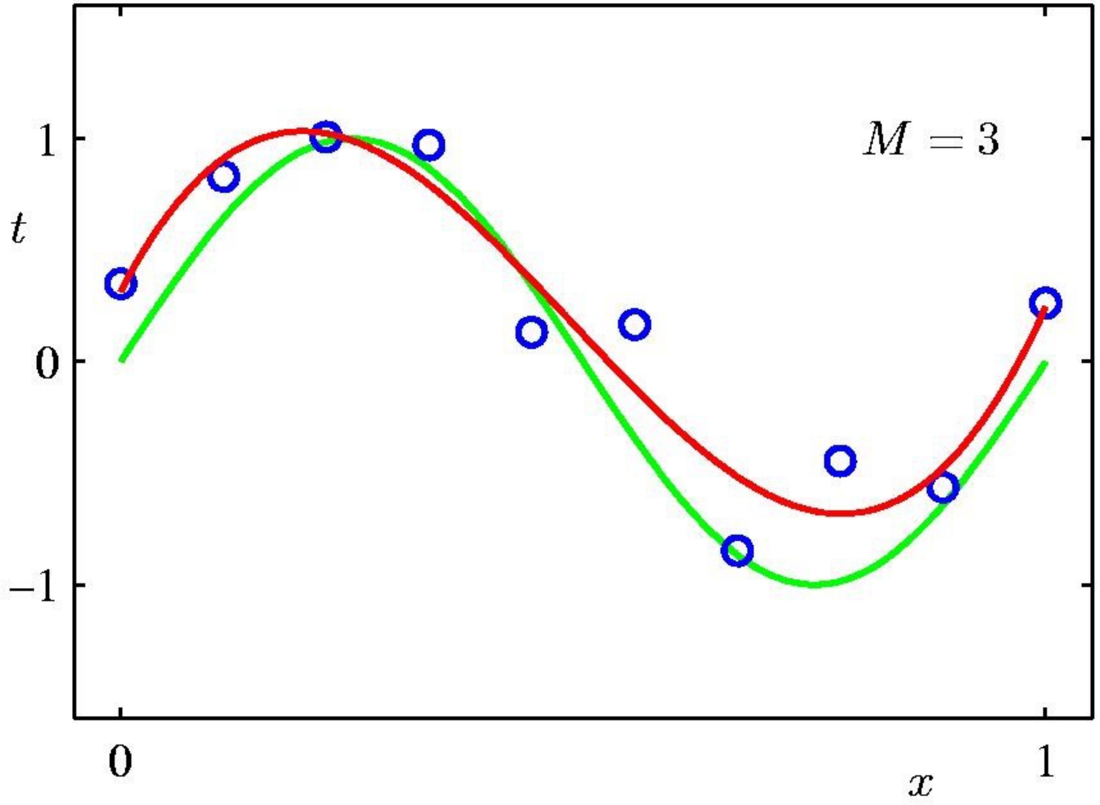
Model Selection: 0th Order Polynomial



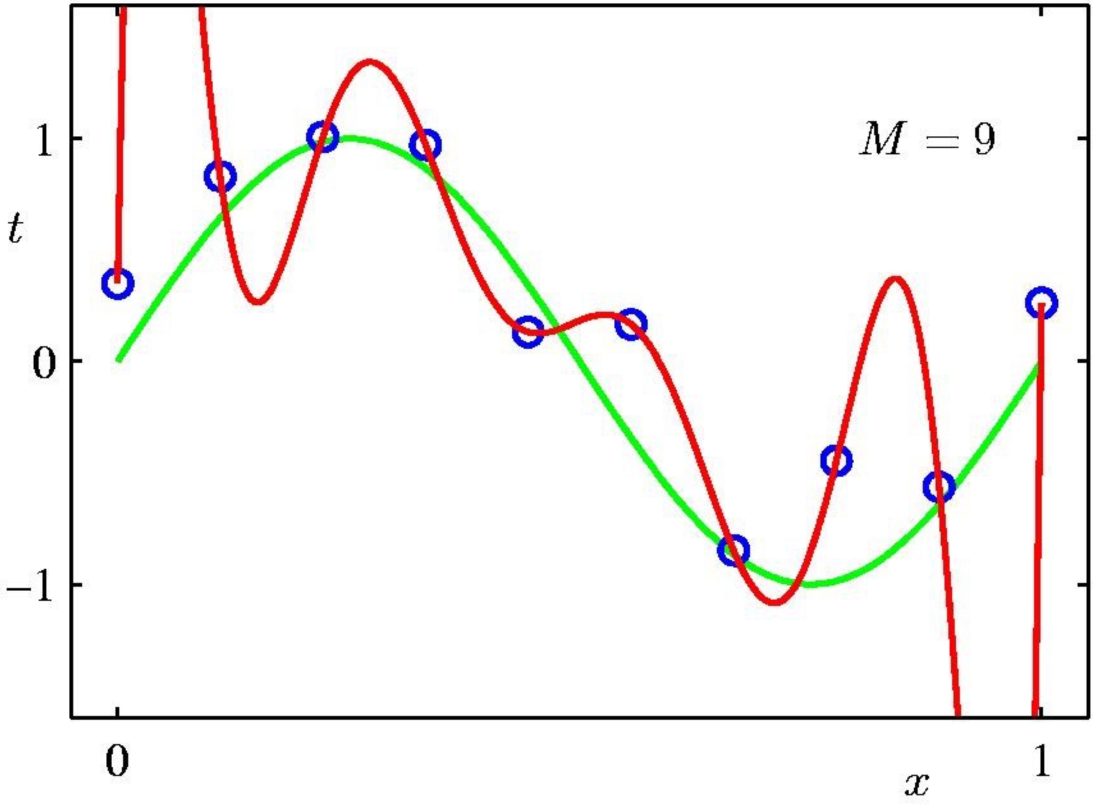
Model Selection: 1st Order Polynomial



Model Selection: 3rd Order Polynomial

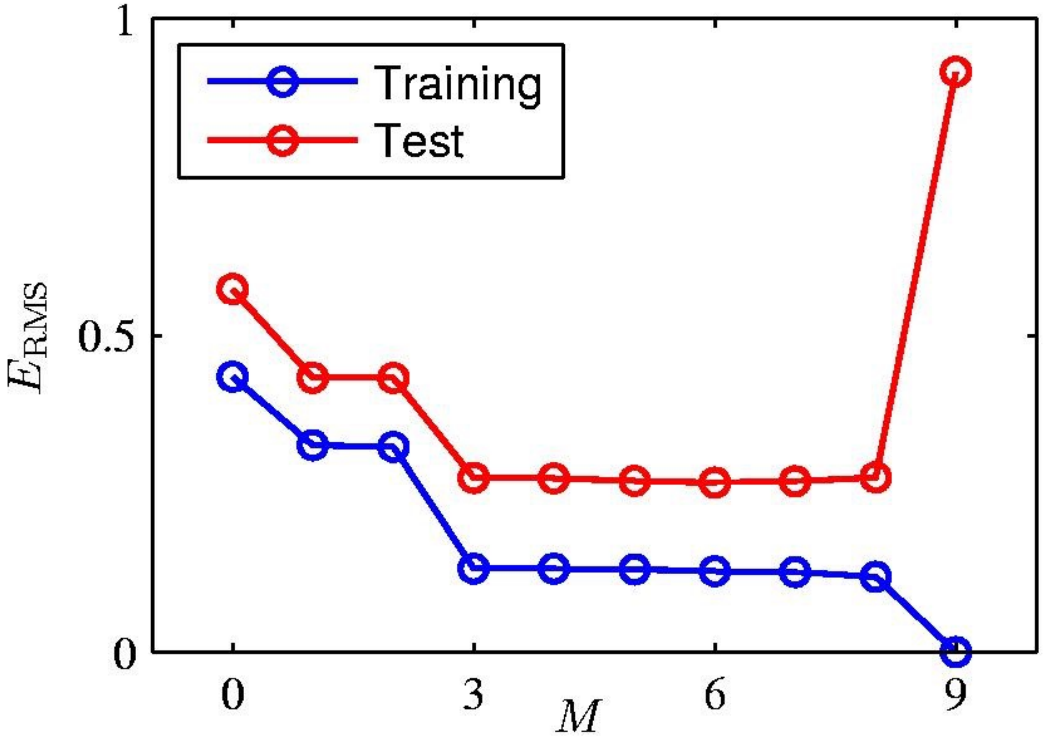


Model Selection: 9th Order Polynomial



Training vs. Test

- What is the best model given these results?
- Why does M=3 perform better than M=9?

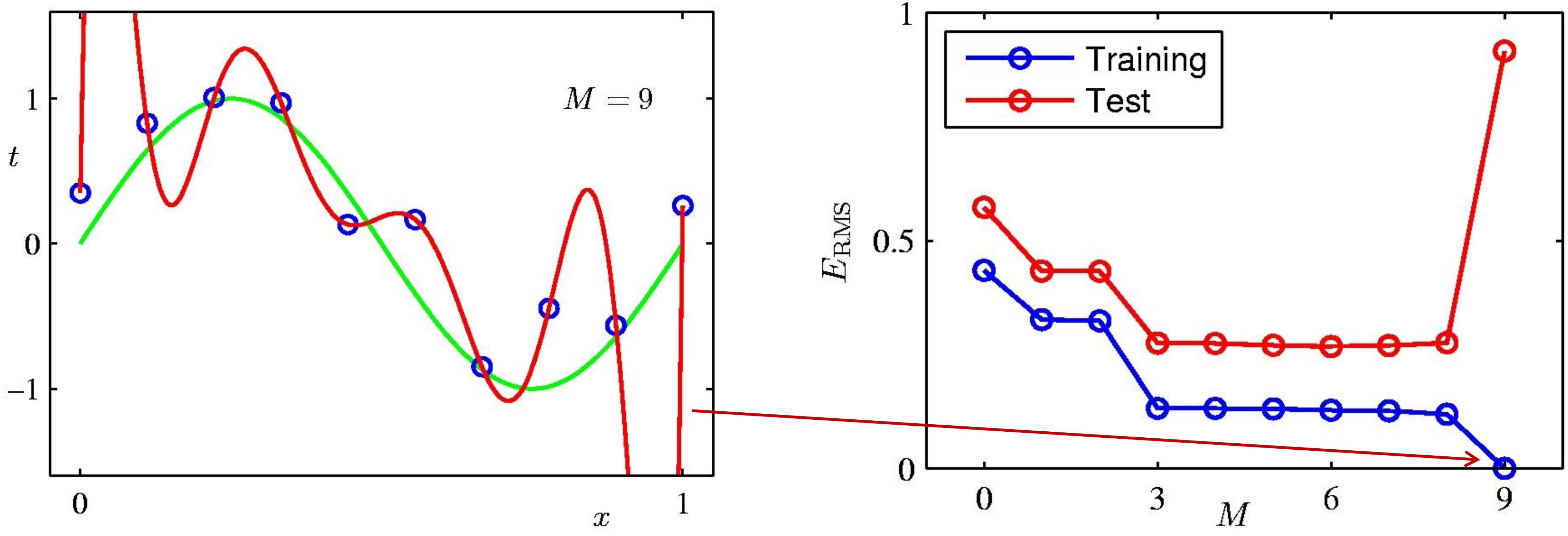


Graph of the RMS error evaluated on training set and on an independent test set.

Root-Mean-Square (RMS) Error:

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^{\star})/N}$$

Over-fitting

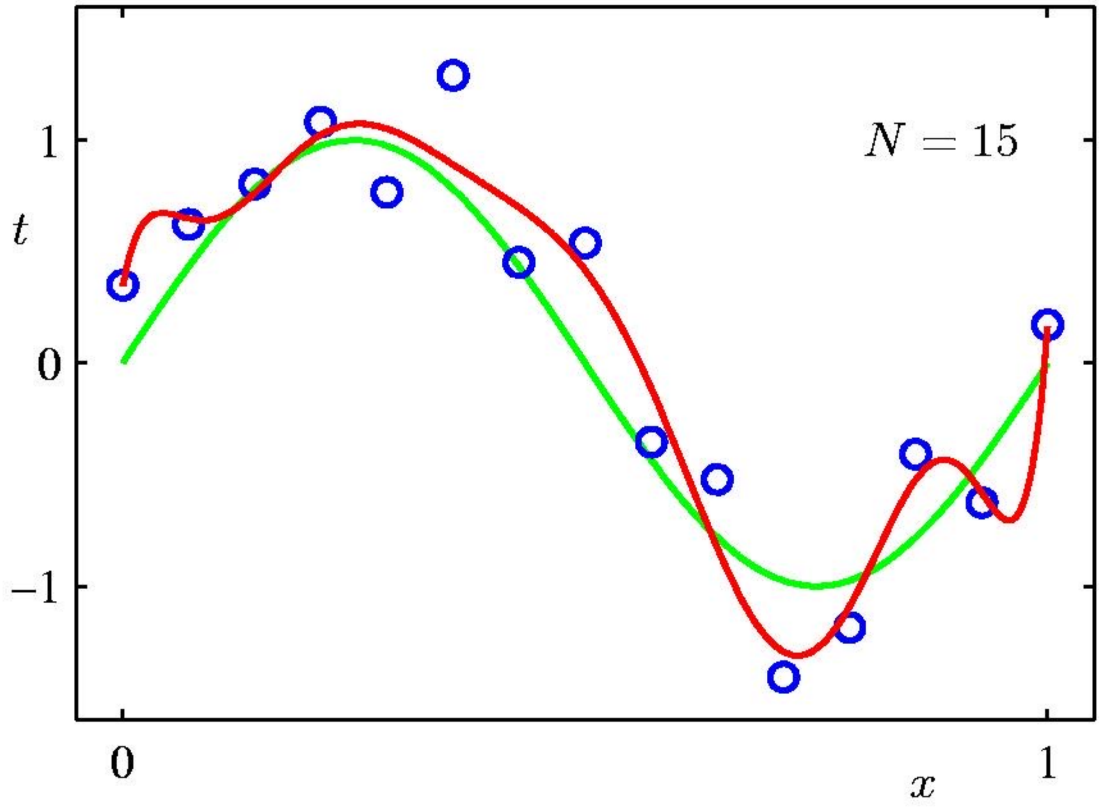


Polynomial Coefficients

	$M = 0$	$M = 1$	$M = 3$	$M = 9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

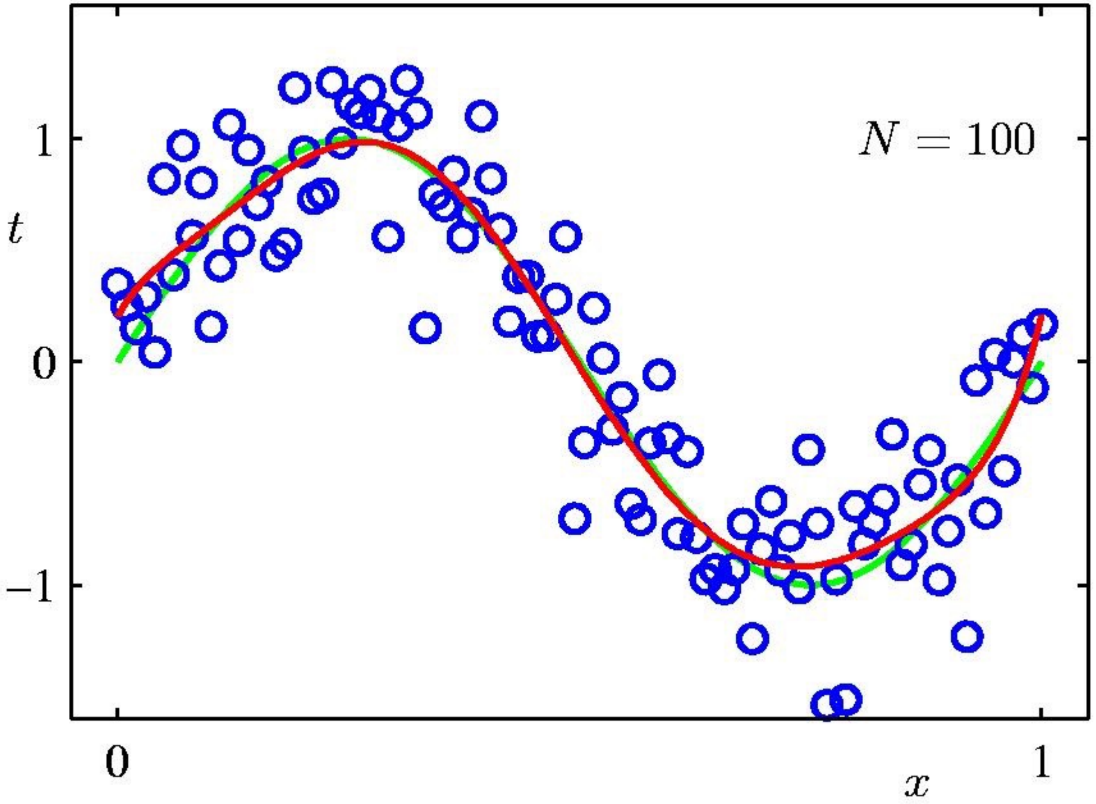
Data Set Size: $N = 15$

9^{th} Order Polynomial



Data Set Size: $N = 100$

9th Order Polynomial



Regularization

- Add penalty term to error function:

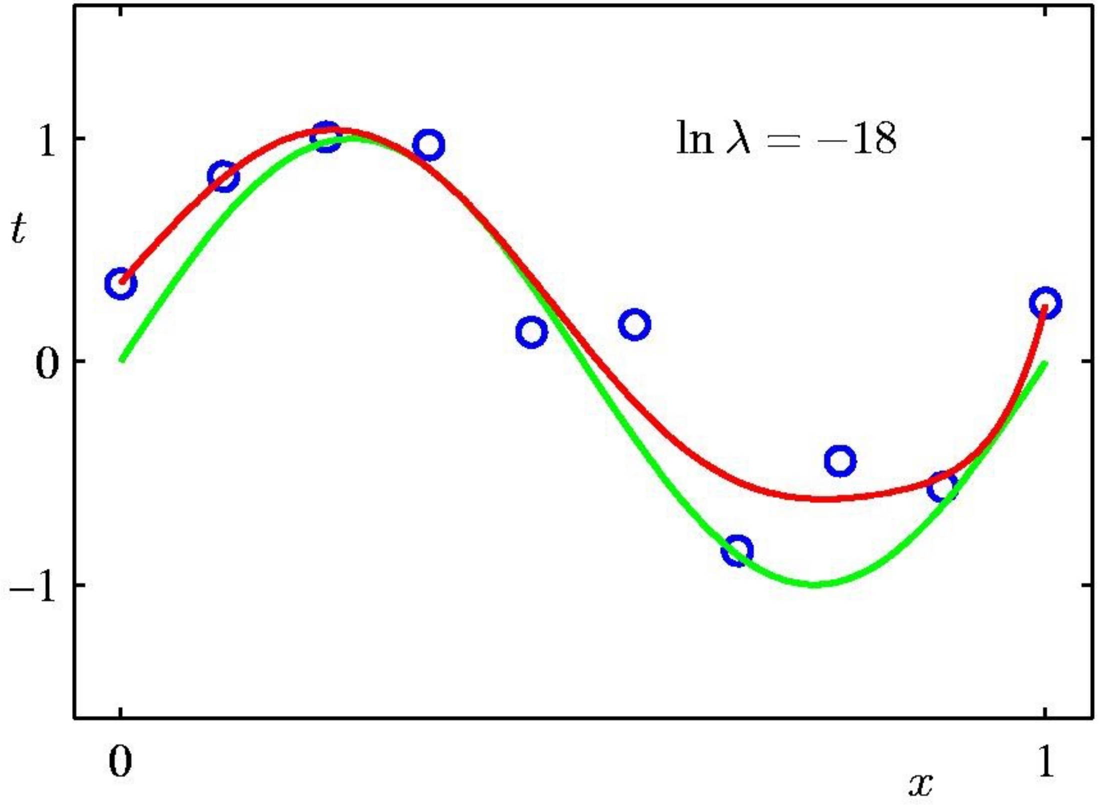
$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

where

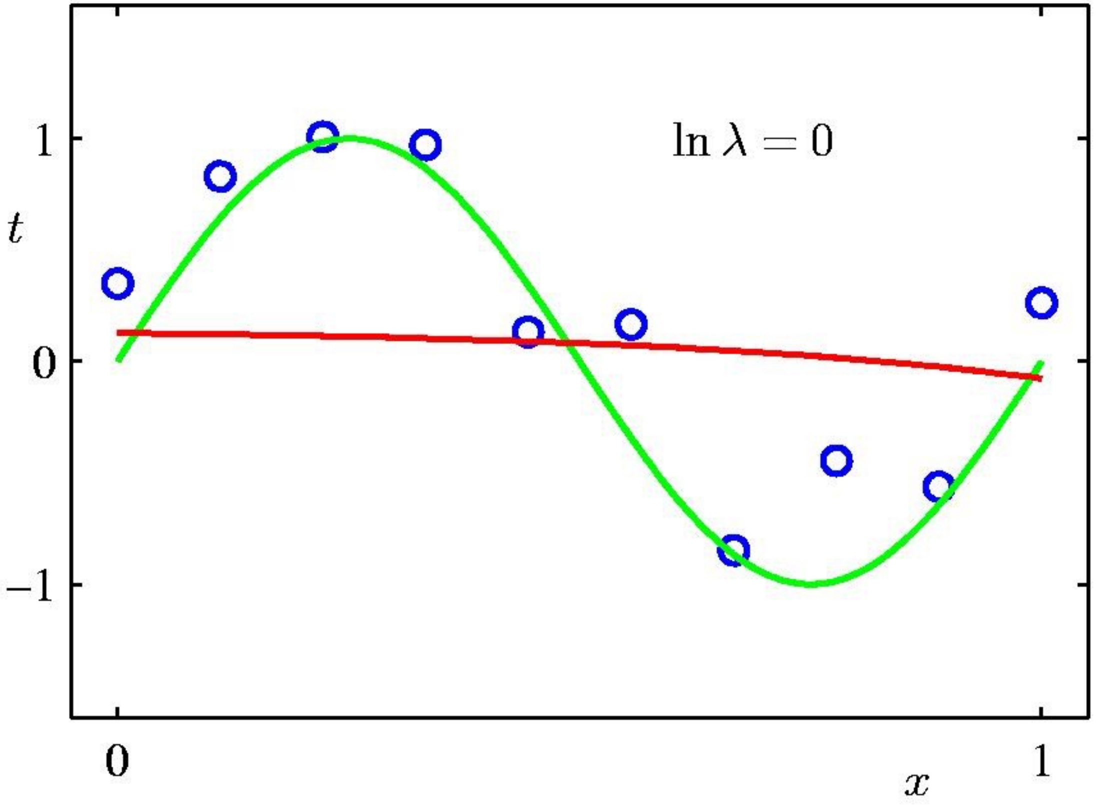
$$||\mathbf{w}||^2 = \mathbf{w}^T\mathbf{w} = w_0^2 + w_1^2 + \ldots + w_M^2$$

- Quadratic regularization yields ridge regression

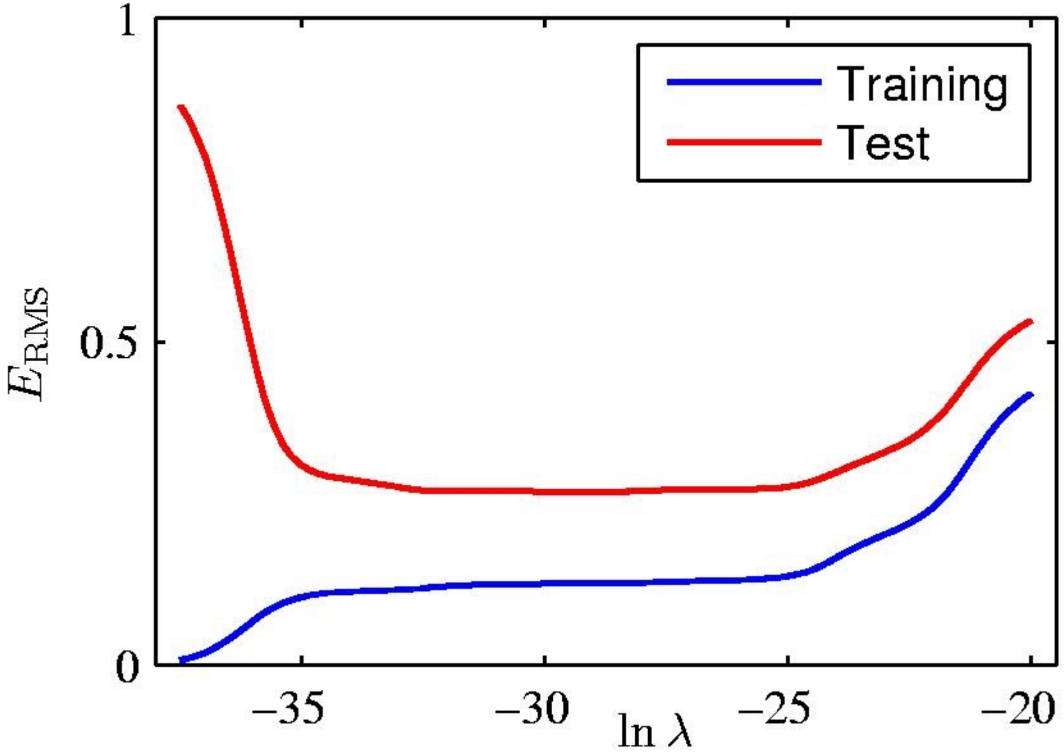
Regularization: $\ln \lambda = -18$



Regularization: $\ln \lambda = 0$



Regularization: E_{RMS} vs. $\ln \lambda$

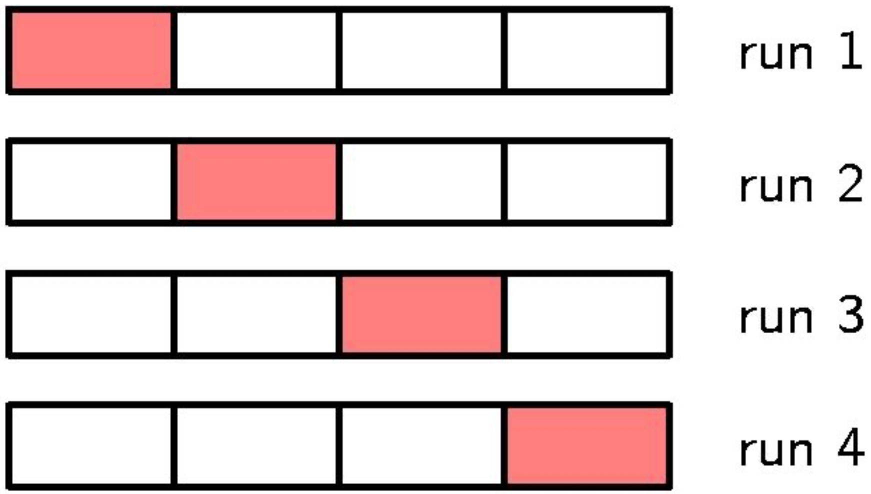


Polynomial Coefficients

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

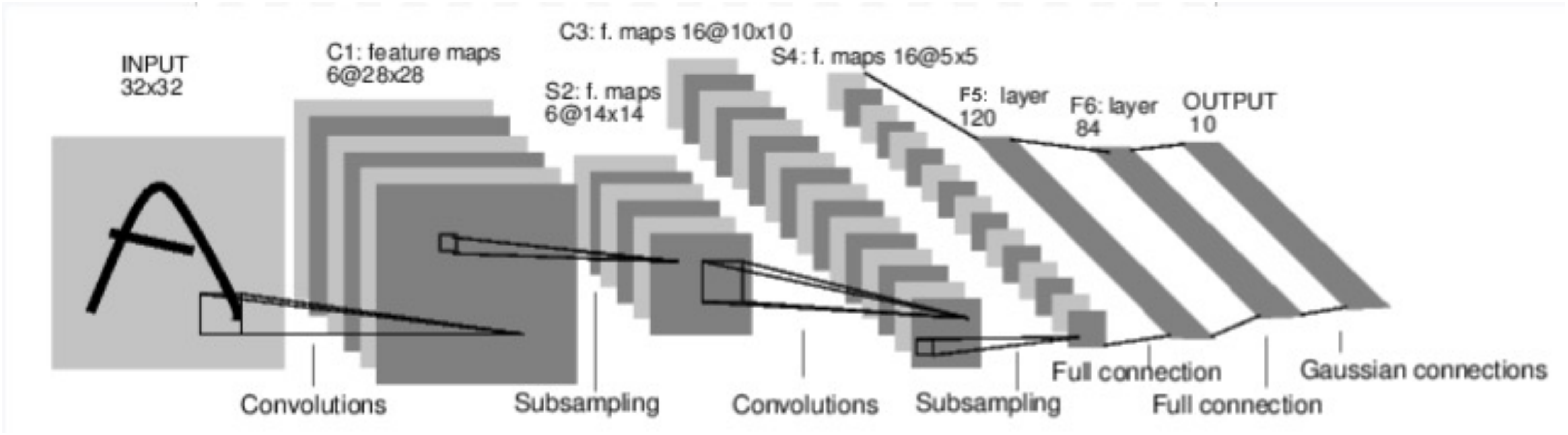
Model Selection

- Performance on training set not a good measure
- Can divide original (training) data into the training set and validation set for comparing models and test set for final test performance
- Cross-validation:



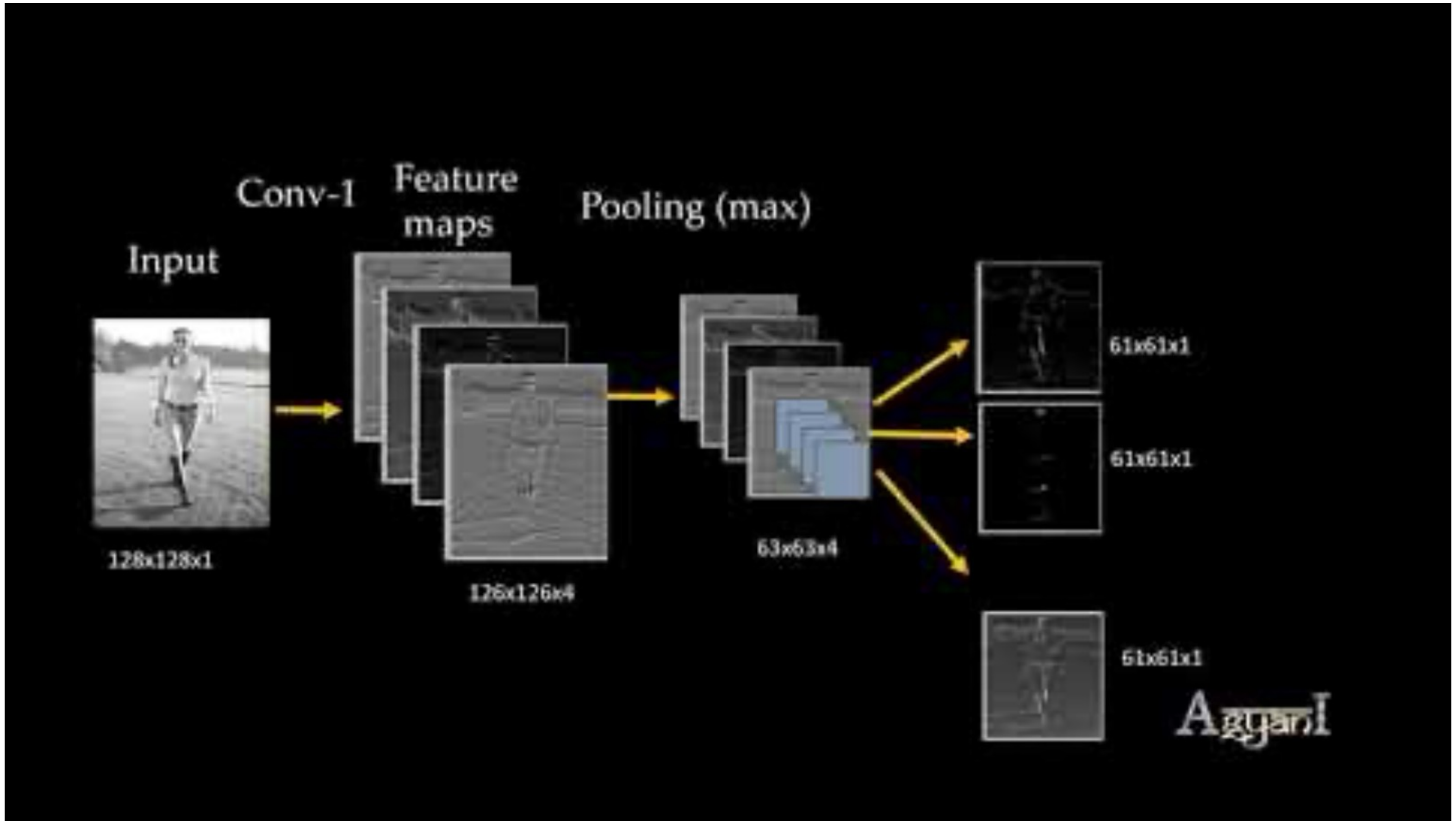
Neural Networks: CNN

- It is a simple feed-forward network. It takes the input, feeds it through several layers one after the other, and then finally gives the output.



Neural Networks: CNN

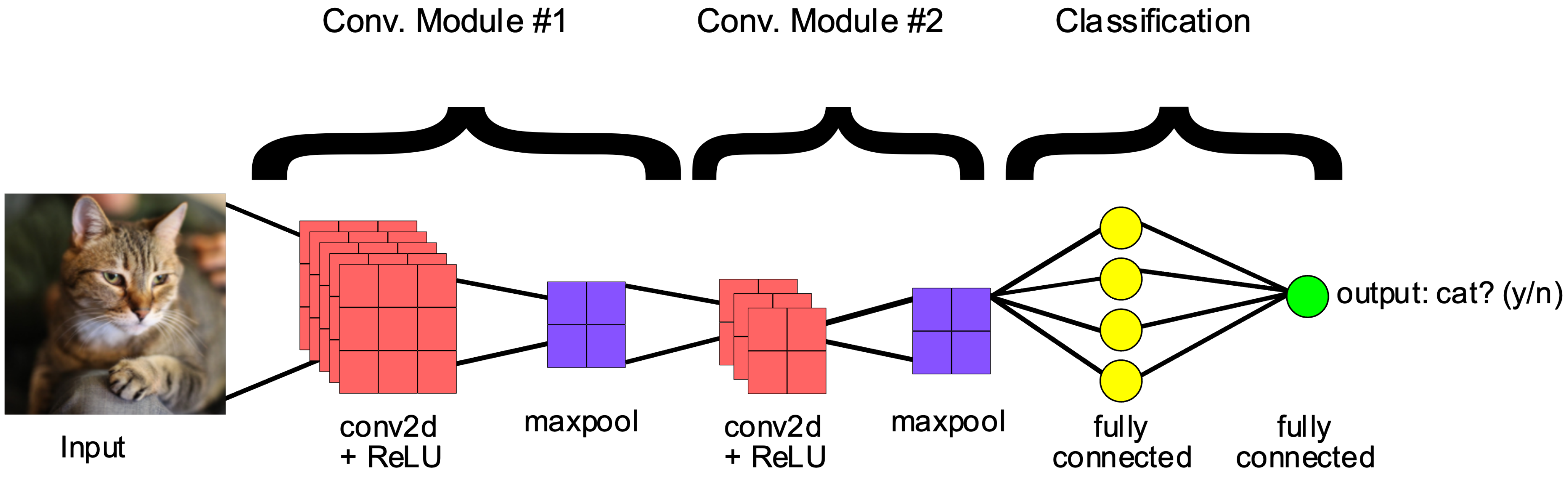
- 2D Convolution Neural Network Animation



https://www.youtube.com/watch?v=CXOGvCMLrkA

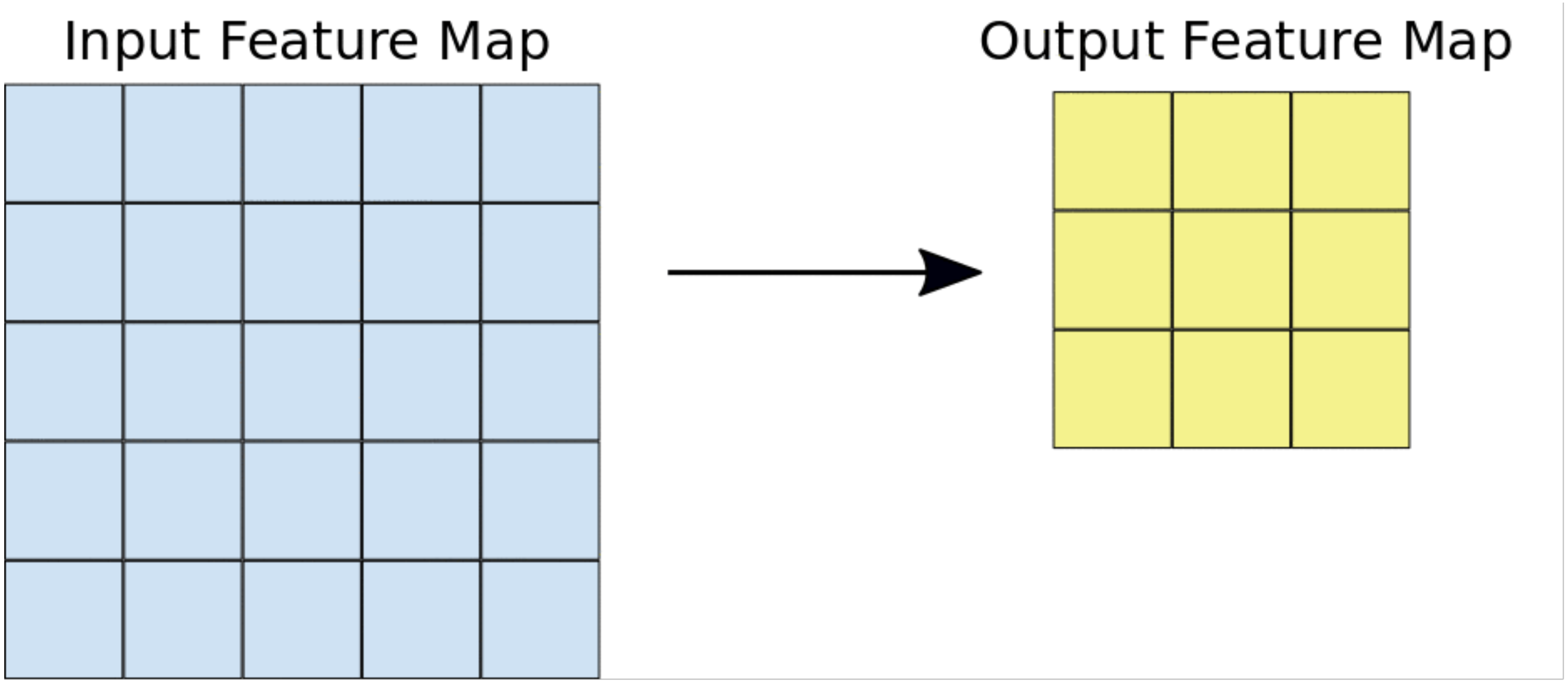
Neural Networks: CNN

- End-to-end structure of a convolutional neural network.



Neural Networks: CNN

- Convolution



Neural Networks: CNN

- Convolution

Input Feature Map

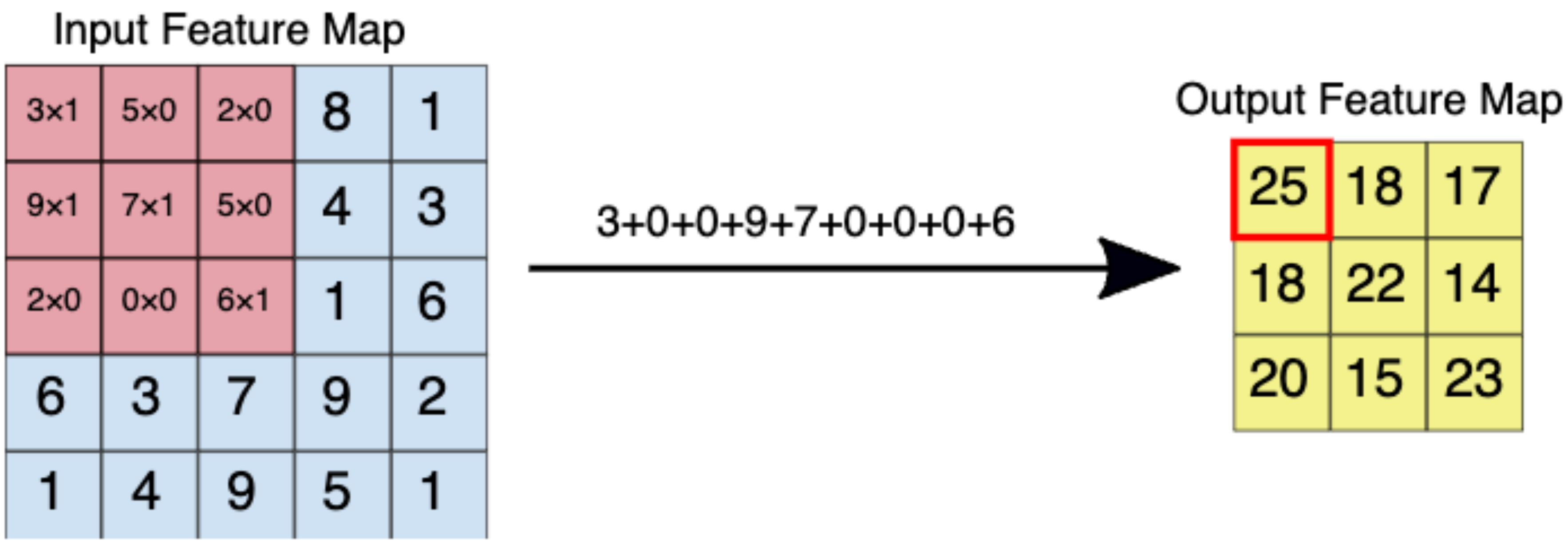
3	5	2	8	1
9	7	5	4	3
2	0	6	1	6
6	3	7	9	2
1	4	9	5	1

Convolutional Filter

1	0	0
1	1	0
0	0	1

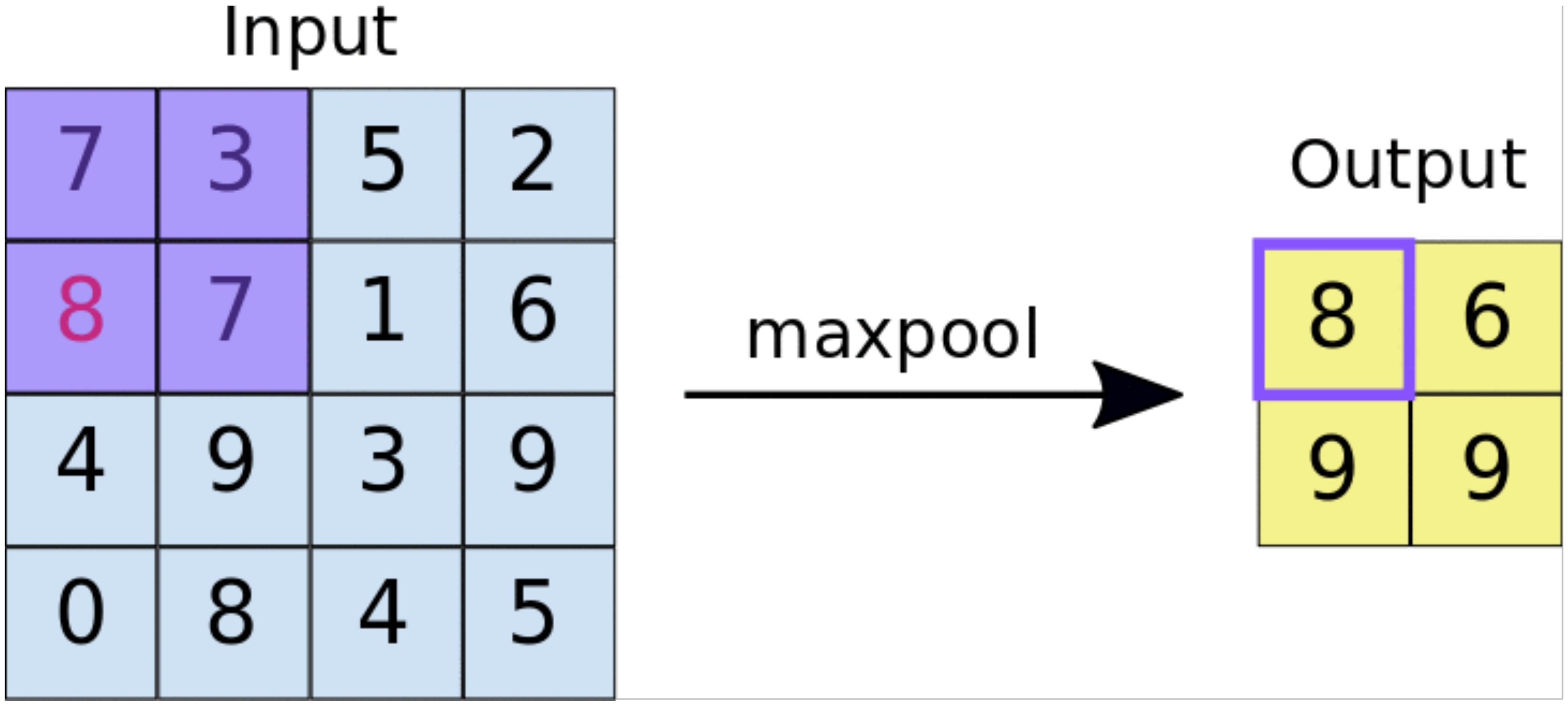
Neural Networks: CNN

- Convolution



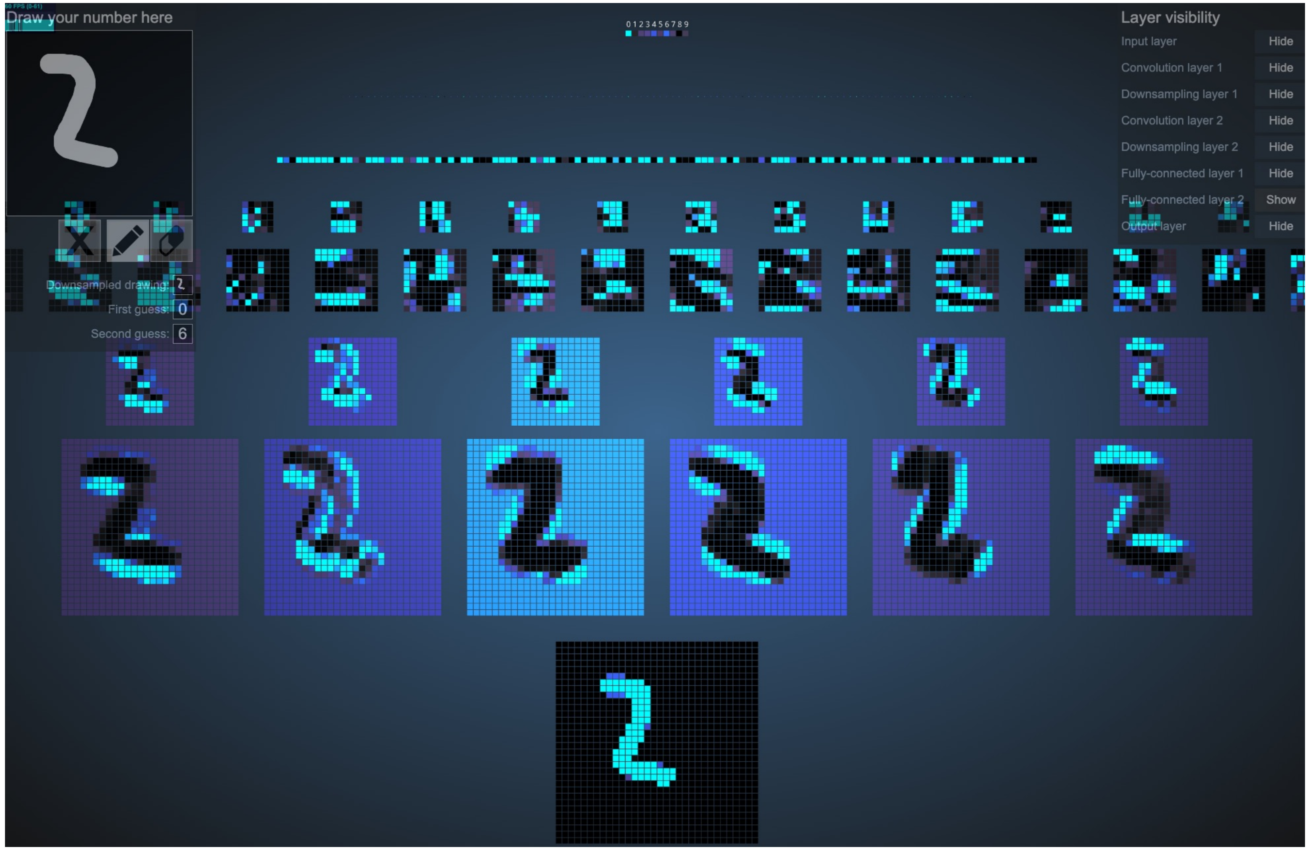
Neural Networks: CNN

- Convolution



Neural Networks: CNN

- [2D Convolutional neural network visualization]()



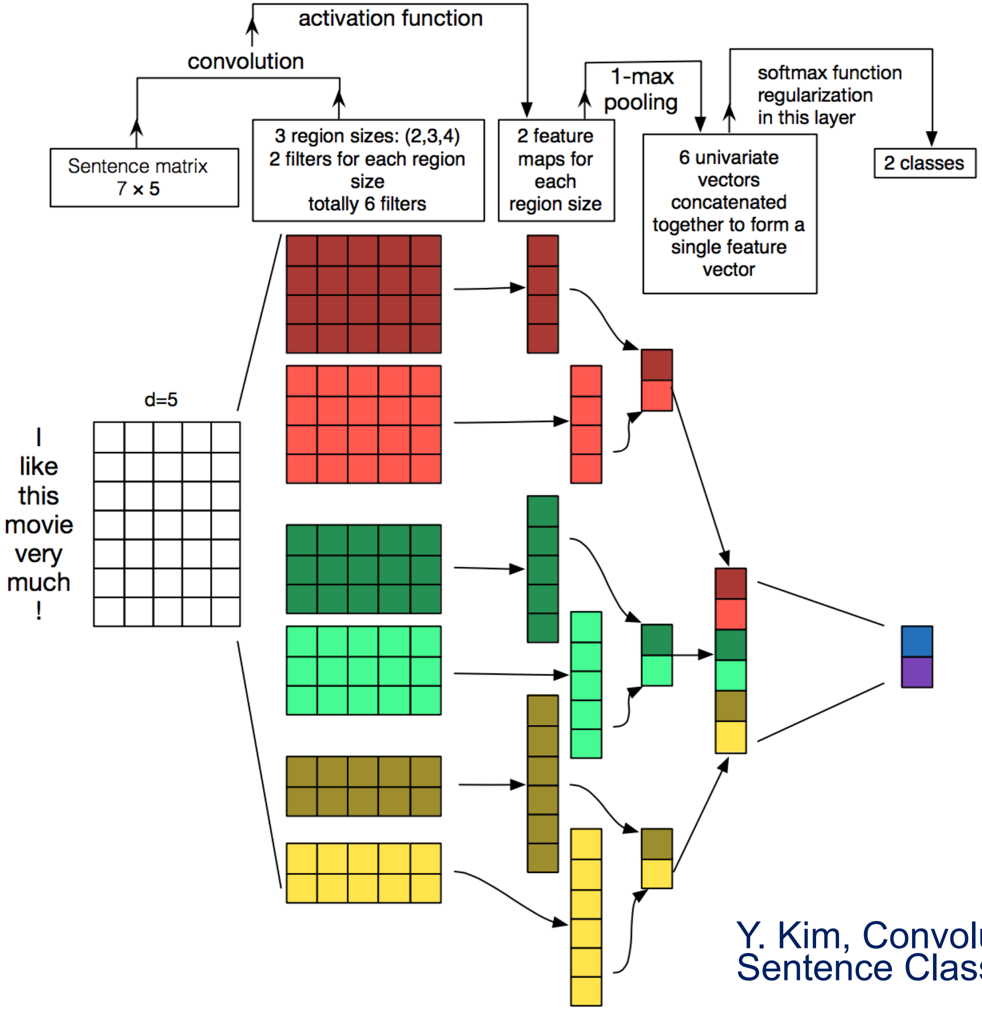
Neural Networks: CNN

A typical training procedure for a neural network is as follows:

- Define the neural network that has some learnable parameters (or weights)
- Iterate over a dataset of inputs
- Process input through the network
- Compute the loss (how far is the output from being correct)
- Propagate gradients back into the network's parameters
- Update the weights of the network, typically using a simple update rule: weight = weight - learning_rate * gradient

Neural Networks: CNN

- Sentiment Analysis
  Method: CNN model



Y. Kim, Convolutional Neural Networks for Sentence Classification, EMNLP 2014

Reference

- C. M. Bishop, Pattern Recognition and Machine Learning. Secaucus, NJ, USA: Springer-Verlag, 2006
- Simon J.D. Prince, Understanding Deep Learning, MIT Press, 2023, https://udlbook.github.io/udlbook/
- J. Liang, Getting Started with Machine Learning, Sept. 2019, Ver. 0.96.
- Machine Learning and Music (IFT6080 Winter 08) Prof. Douglas Eck, Université de Montréal, https://www.iro.umontreal.ca/~pift6080/H09/documents/intro_ml.pdf
- ML Practicum: Image Classification: https://developers.google.com/machine-learning/practica/image-classification/convolutional-neural-networks
- An Interactive Node-Link Visualization of Convolutional Neural Networks: https://adamharley.com/nn_vis/