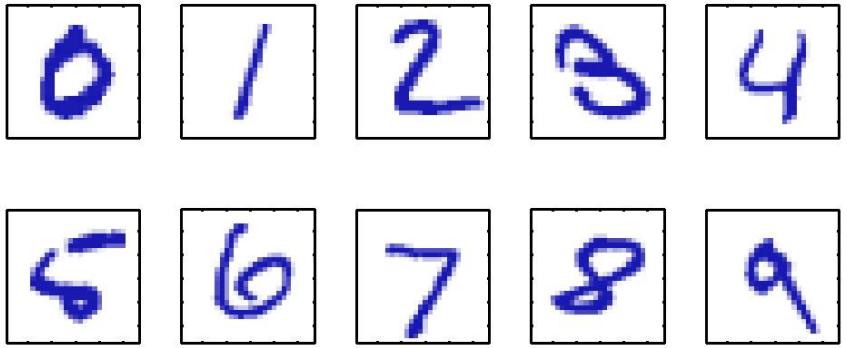
CS559/659 Machine Learning

Xubo Song
songx@ohsu.edu

Machine Learning

- The field of **machine learning** studies the design of computer programs (agents) capable of learning from past experience or adapting to changes in the environment

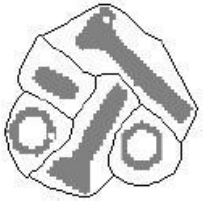
- The need for building agents capable of learning is everywhere
  - predictions in medicine,
  - text and web page classification,
  - speech recognition,
  - image/text retrieval,
  - commercial software

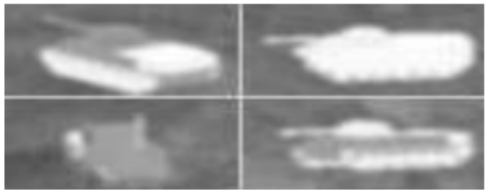


Examples of Machine Learning Problems

Machine vision

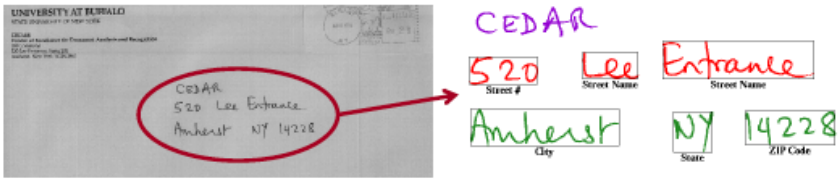
- Visual inspection, ATR
- Imaging device detects ground target
- Classification into "friend" or "foe"





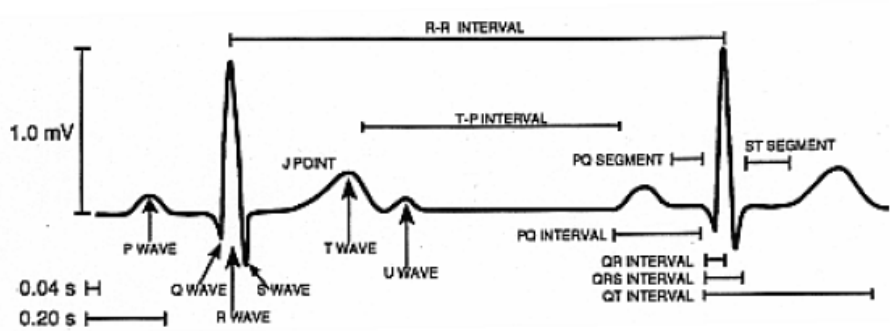
Character recognition

- Automated mail sorting, processing bank checks
- Scanner captures an image of the text
- Image is converted into constituent characters



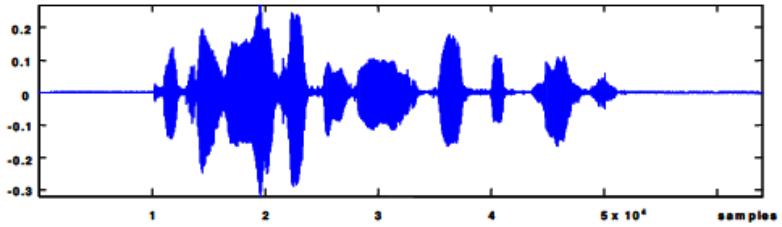
Computer aided diagnosis

- Medical imaging, EEG, ECG signal analysis
- Designed to assist (not replace) physicians
- Example: X-ray mammography
  - 10-30% false negatives in x-ray mammograms
  - 2/3 of these could be prevented with proper analysis



Speech recognition

- Human Computer Interaction, Universal Access
- Microphone records acoustic signal
- Speech signal is classified into phonemes and/or words



Related Fields and Application Areas

Related fields

- Adaptive signal processing
- Machine learning
- Artificial neural networks
- Robotics and vision
- Cognitive sciences
- Mathematical statistics
- Nonlinear optimization
- Exploratory data analysis
- Fuzzy and genetic systems
- Detection and estimation theory
- Formal languages
- Structural modeling
- Biological cybernetics
- Computational neuroscience

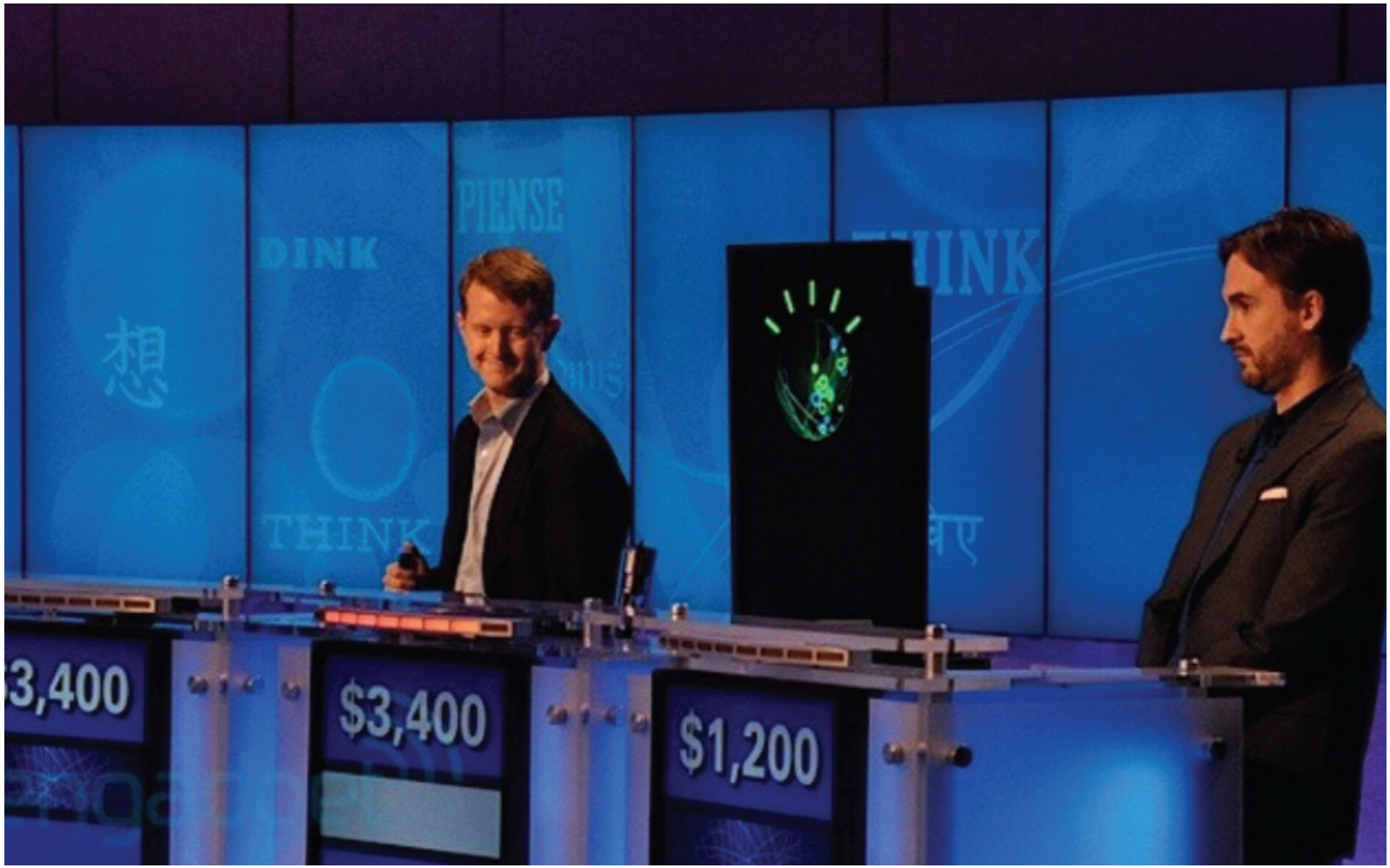
Applications

- Image processing
- Computer vision
- Speech recognition
- Multimodal interfaces
- Automated target recognition
- Optical character recognition
- Seismic analysis
- Man and machine diagnostics
- Fingerprint identification
- Industrial inspection
- Financial forecast
- Medical diagnosis
- ECG signal analysis

More Success Stories

- object recognition/localization/detection
- Speech/speaker/language recognition
- Text to speech
- Natural language understanding
- Detect skin cancer / melanoma
- DL for classification of star systems
- Consumer protection against fraud
- Self-driving cars and intelligent roads
- real-time machine translation
- Weather forecast

Intelligence in games: the beginning


PIENSE
DINK
想
THINK
THINK
$3,400
$1,200

Recent progress in AI

The 2 best examples of the success of new ML

- AlphaGo
- Mobileye

FINANCIAL TIMES

ft.com/comment

Home World Companies Markets Global Economy Lex Comment Management Life & Arts

PERSON IN THE NEWS

Demis Hassabis, master of the new machine age

Murad Ahmed

The creator of the AI game-playing program makes all the right moves, writes Murad Ahmed

The victories have a human mastermind in Demis Hassabis, co-founder and chief executive of DeepMind. He describes Mr Lee as the "Roger Federer of Go", and for some the computer program's achievement is akin to a robot taking to the lawns of Wimbledon and beating the legendary tennis champion.

"I think it is pretty huge but, ultimately, it will be for

Real Engineering: Mobileye

Real Engineering: Mobileye

Math Foundations

- Functional analysis (~45mins)

Linear Algebra

Basic notion and definitions: matrix and vectors norms, positive, symmetric, invertible matrices, linear systems, condition number.

Functional Analysis:

Linear and Euclidean spaces
scalar product, orthogonality
orthonormal bases, norms and semi-norms,
Cauchy sequence and complete spaces
Hilbert spaces, function spaces
and linear functional, Riesz representation
theorem, convex functions, functional calculus.

- Probability (~45mins)

Probability Theory:

Random Variables (and related concepts), Law of Large Numbers, Probabilistic Convergence, Concentration Inequalities.

Learning

Learning process:

Learner (a computer program) processes data ***D*** representing past experiences and tries to either develop an appropriate response to future data, or describe in some meaningful way the data seen

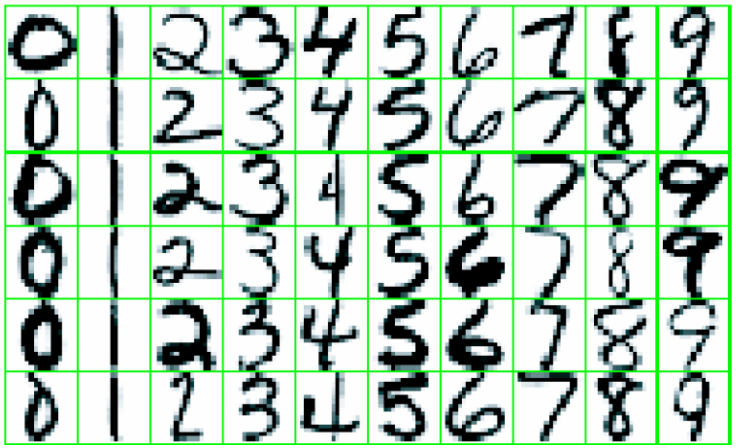
Example:

Learner sees a set of patient cases (patient records) with corresponding diagnoses. It can either try:

- to predict the presence of a disease for future patients
- describe the dependencies between diseases, symptoms

Machine Learning?

- Programming computers to optimize a performance criterion using example data or past experimence
- Learning is not memorizing!
- Crucial: Ability to generalize to novel situations
- Often implies an element of uncertainity
  (e.g. spreadsheet calculations do not require learning)
- Uncertainity could be due to
  - Noise in the input or measurement (e.g. EEG)
  - Generative process is complex (e.g. speech recognition)
  - High-level of variability (e.g. handwriting recognition)



Types of learning

- **Supervised learning**
  - Learning mapping between input **x** and desired output **y**
  - Teacher gives me y's for the learning purposes
- **Unsupervised learning**
  - Learning relations between data components
  - No specific outputs given by a teacher
- **Reinforcement learning**
  - Learning mapping between input **x** and desired output y
  - Critic does not give me y's but instead a signal (reinforcement) of how good my answer was
- **Other types of learning:**
  - **Concept learning, Active learning**

Supervised learning

Data: $D = \{d_1, d_2, .., d_n\}$ **a set of *n* examples**

$d_i = < \mathbf{x}_i, y_i >$

$\mathbf{x}_i$ is input vector, and y is desired output (given by a teacher)

Objective: learn the mapping $f : X \to Y$

s.t. $y_i \approx f(x_i)$ for all $i = 1, .., n$

Two types of problems:

- **Regression:** X discrete or continuous → Y is **continuous**
- **Classification:** X discrete or continuous → Y is **discrete**

Supervised learning examples

- **Regression:** Y is **continuous**

Debt/equity
Earnings
Future product orders

→ company stock price

- **Classification:** Y is **discrete**

→ Label "3"

Handwritten digit (array of 0,1s)

Unsupervised learning

- **Data:** $D = \{d_1, d_2, .., d_n\}$

  $d_i = \mathbf{x}_i$ vector of values

  No target value (output) y

- **Objective:**
  - learn relations between samples, components of samples

Types of problems:

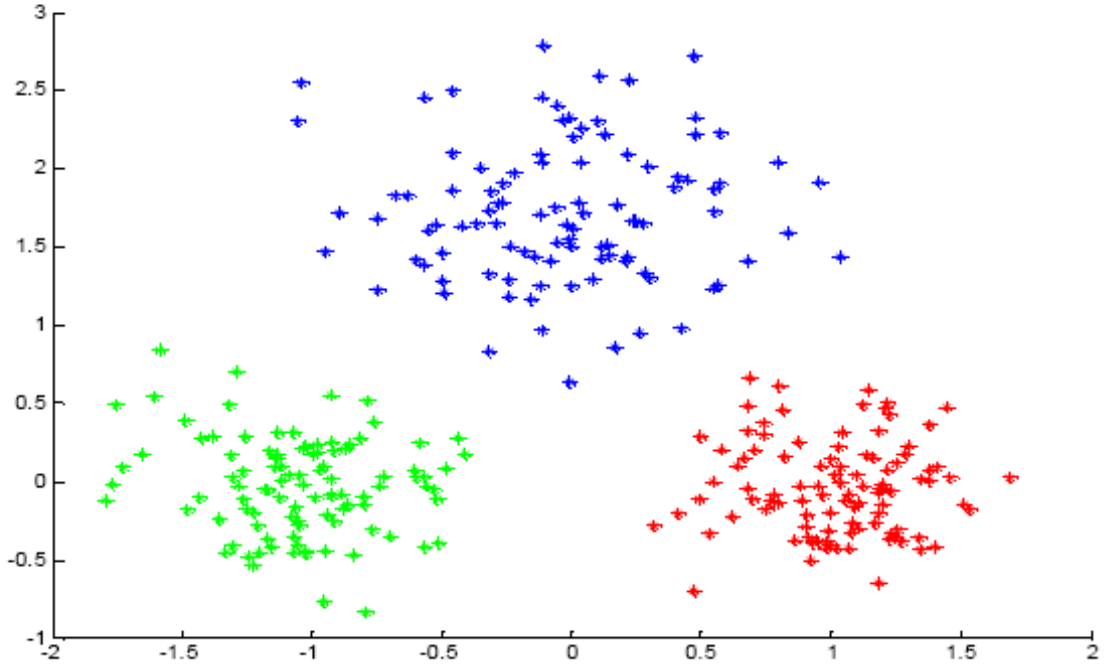
- **Clustering**

  Group together "similar" examples, e.g. patient cases

- **Density estimation**
  - Model probabilistically the population of samples

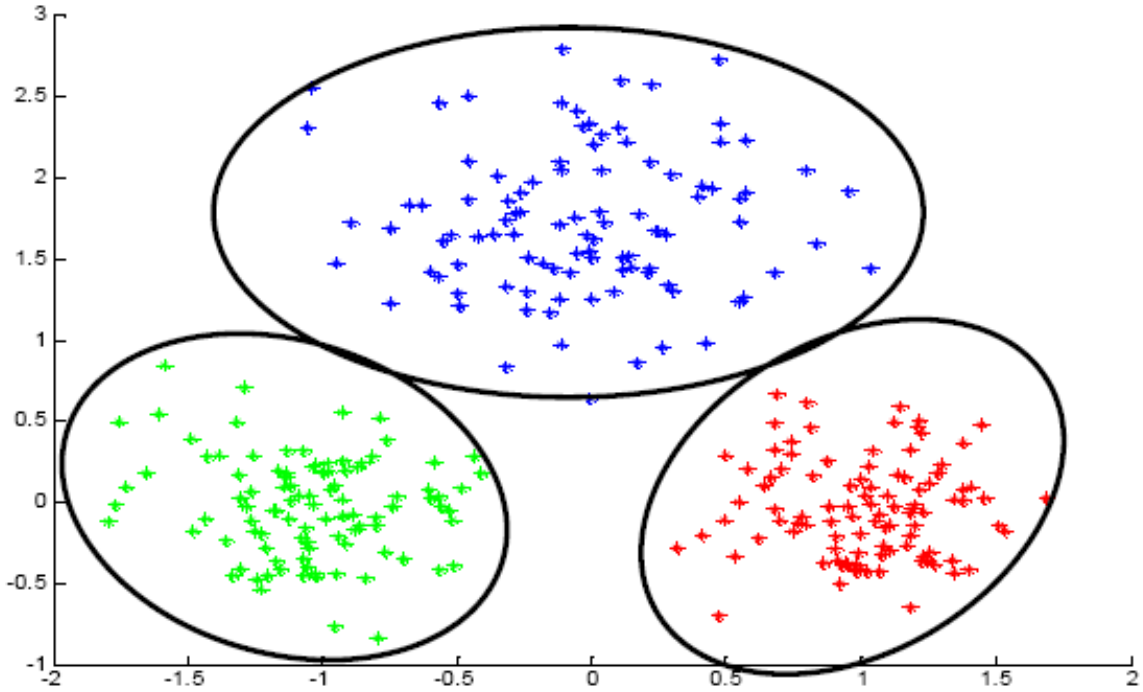
Unsupervised learning example

- **Clustering.** Group together similar examples $d_i = \mathbf{x}_i$



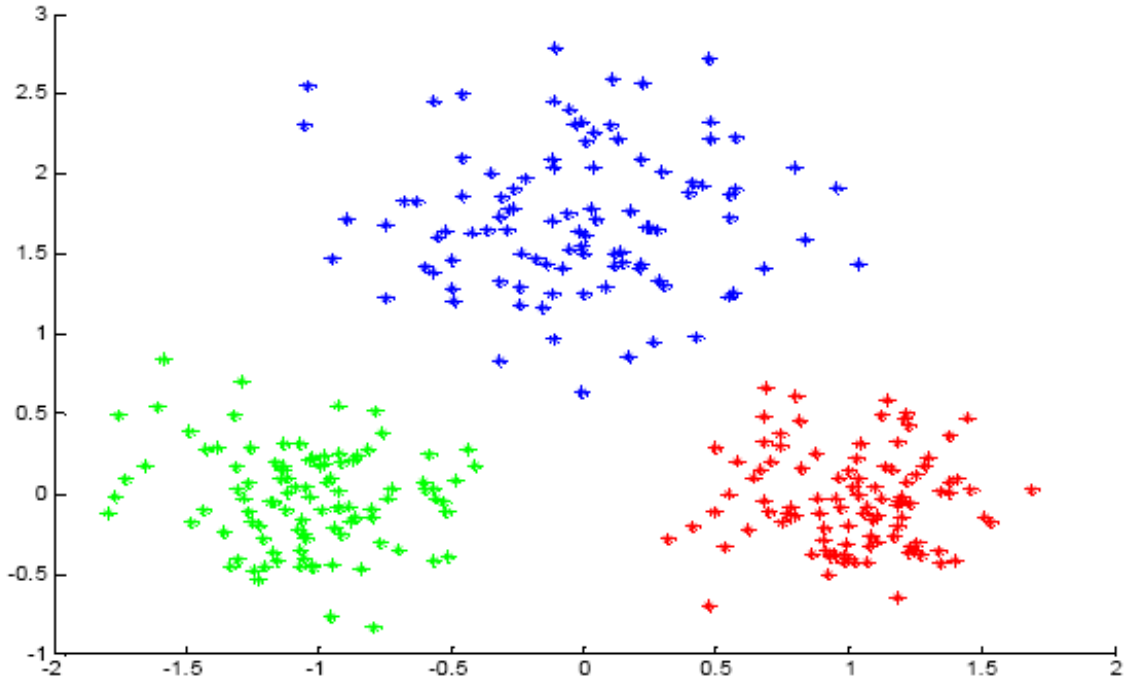
Unsupervised learning example

- **Clustering.** Group together similar examples $d_i = \mathbf{x}_i$



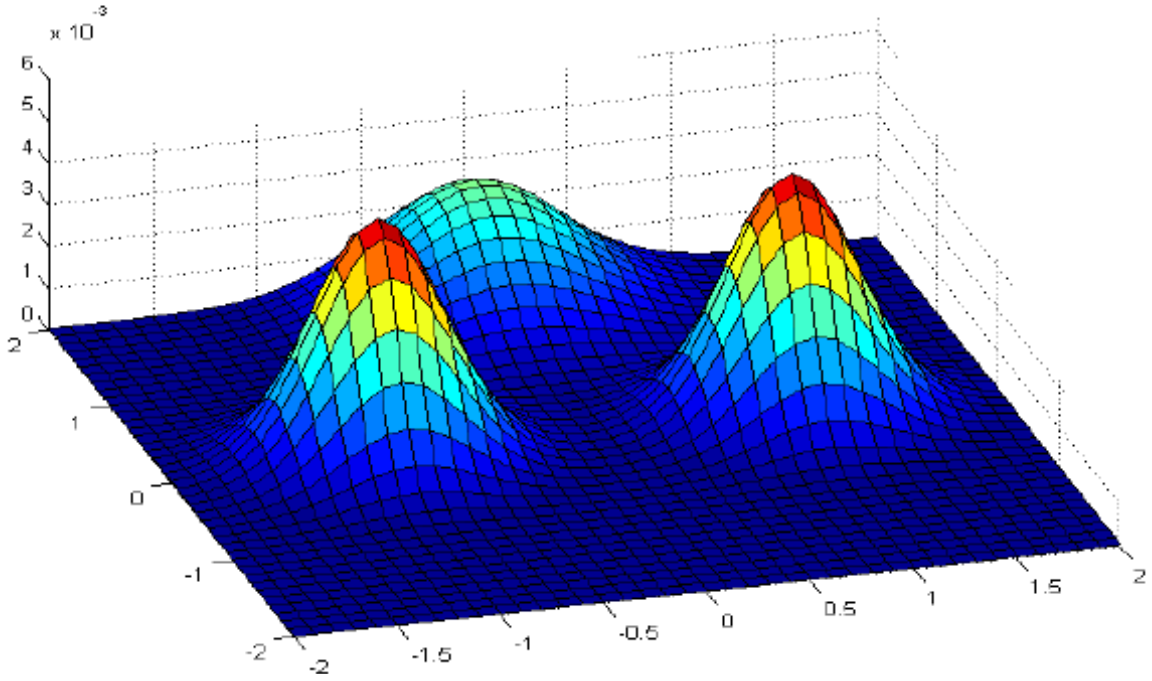
Unsupervised learning example

- **Density estimation.** We want to build the probability model P(**x**) of a population from which we draw examples $d_i = \mathbf{x}_i$



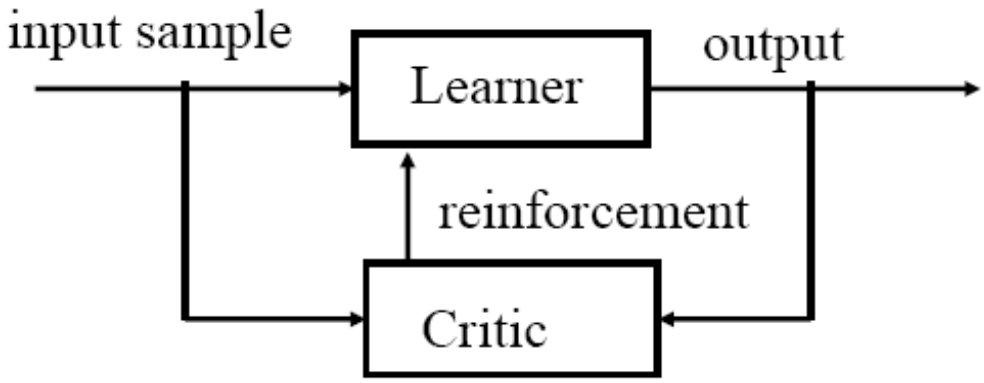
Unsupervised learning. Density estimation

- A probability density of a point in the two dimensional space
  - Model used here: **Mixture of Gaussians**



Reinforcement learning

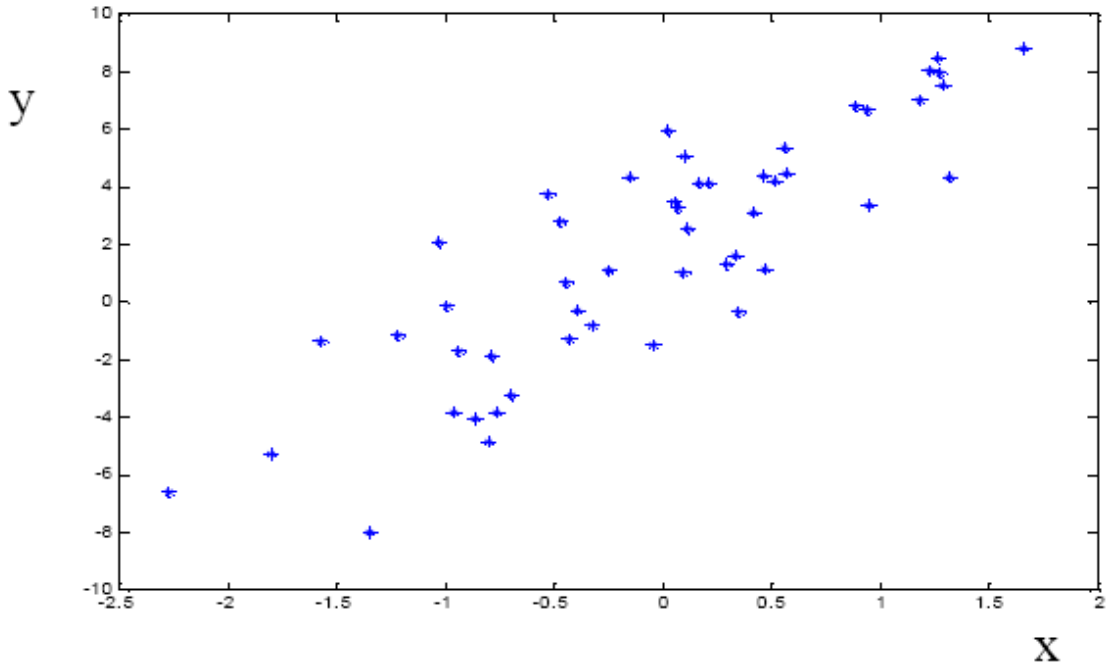
- We want to learn: $f : X \rightarrow Y$
- We see samples of **x** but not y
- Instead of y we get a feedback (reinforcement) from a **critic** about how good our output was



- The goal is to select outputs that lead to the best reinforcement

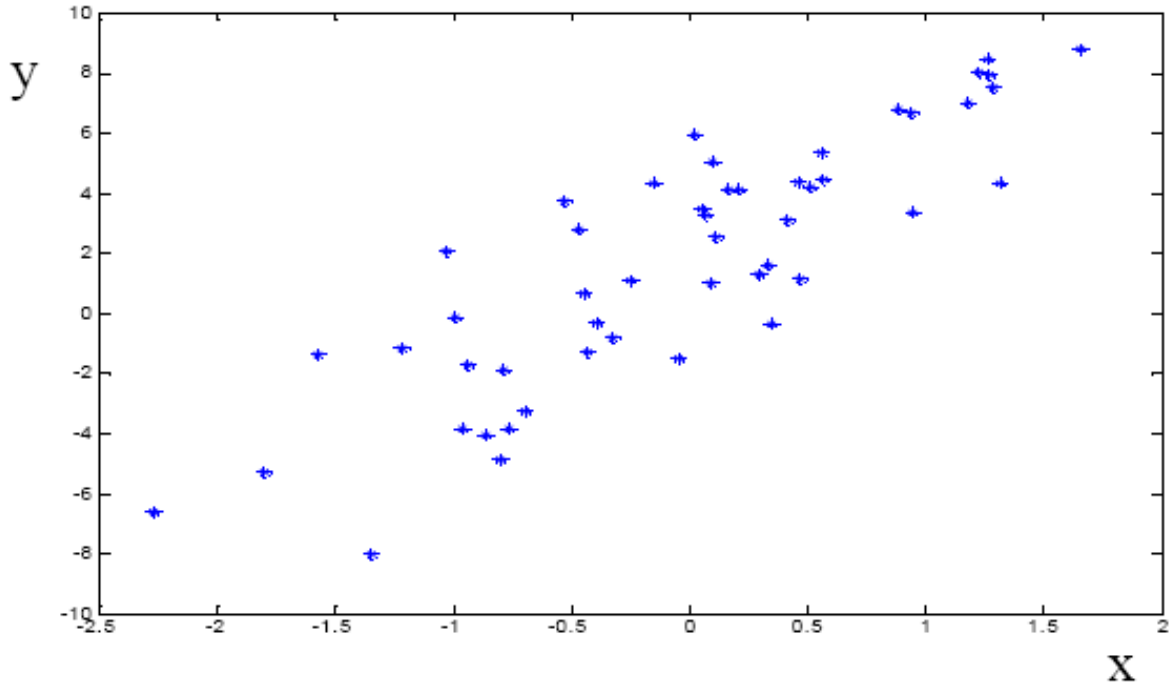
Learning: first look

- Assume we see examples of pairs (**x** , y) in D and we want to learn the mapping $f : X \rightarrow Y$ to predict y for some future **x**
- We get the data D - what should we do?



Learning: first look

- **Problem:** many possible functions $f : X \to Y$ exists for representing the mapping between **x** and y
- Which one to choose? Many examples still unseen!



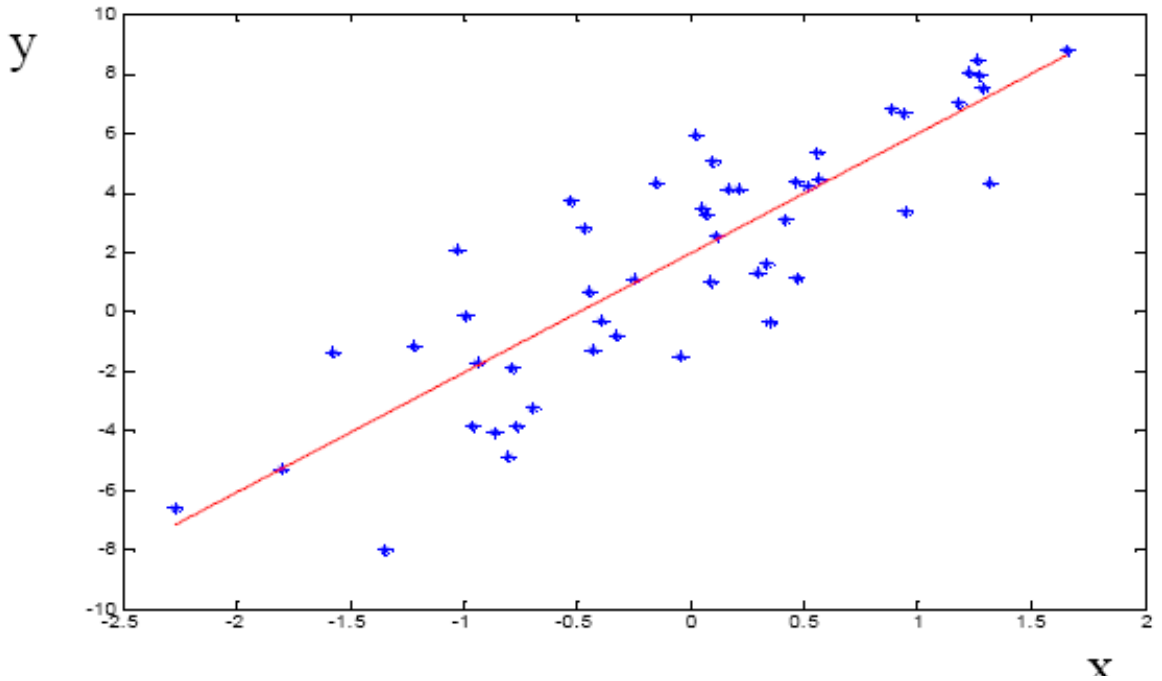
Learning: first look

- **Solution:** make an assumption about the model, say,

$$f(x) = ax + b + \varepsilon$$

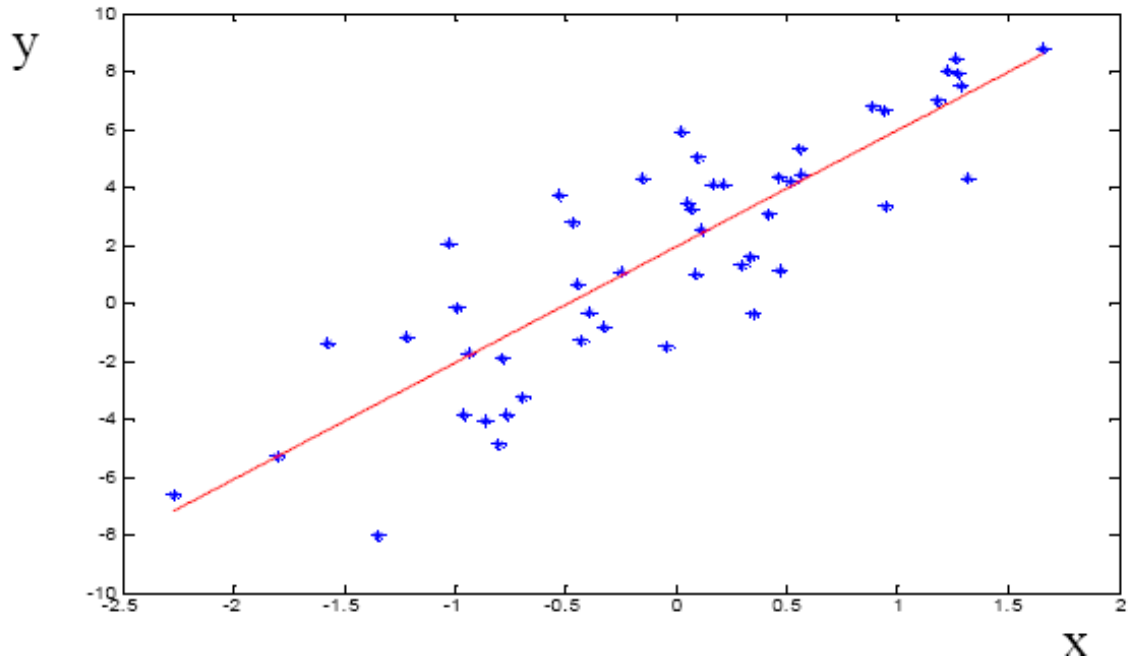
$\varepsilon = N(0, \sigma)$ - random (normally distributed) noise

- Restriction to a linear model is an example of learning bia



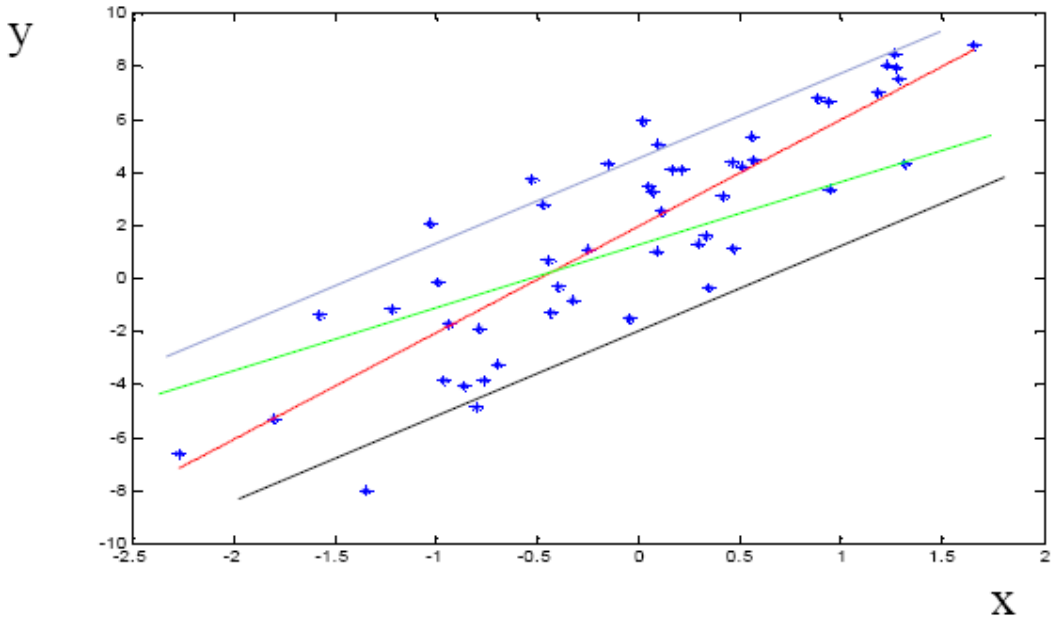
Learning: first look

- **Bias** provides the learner with some basis for choosing among possible representations of the function.
- **Forms of bias:** constraints, restrictions, model preferences
- **Important: There is no learning without a bias!**



Learning: first look

- Choosing a parametric model or a set of models is not enough
  Still too many functions $f(x) = ax + b + \varepsilon \quad \varepsilon = N(0, \sigma)$
  - One for every pair of parameters $a,\ b$



Fitting the data to the model

- We want the **best set** of model parameters

Objective: Find parameters that:

- reduce the misfit between the model **M** and observed data ***D***
- Or, (in other words) explain the data the best

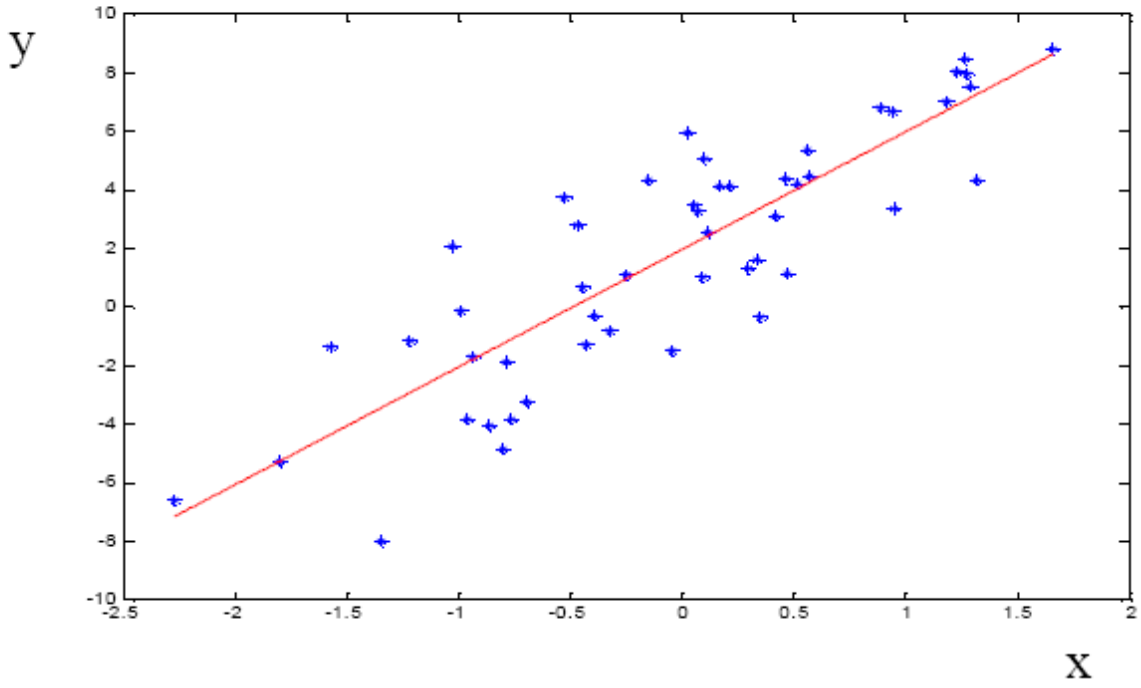
Objective function:

- **Error function: Measures the misfit between *D* and M**
- **Examples of error functions:**
  - Average Square Error $\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$
  - Average misclassification error $\frac{1}{n}\sum_{i=1}^{n} 1_{y_i \neq f(x_i)}$

Average # of misclassified cases

Fitting the data to the model

- **Linear regression problem**
  - Minimizes the squared error function for the linear model
  - minimizes $\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$



Learning: summary

Three basic steps:

- **Select a model** or a set of models (with parameters)

  E.g. $y = ax + b$

- **Select the error function** to be optimized

  E.g. $\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$

- **Find the set of parameters optimizing the error function**
  - The model and parameters with the smallest error represent the best fit of the model to the data

But there are problems one must be careful about …

Learning

Problem

- We fit the model based on past experience (past examples seen)
- But ultimately we are interested in learning the mapping that performs well on the whole population of examples

Training data: **Data used to fit the parameters of the model**

Training error: $\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$

True (generalization) error (over the whole unknown population):

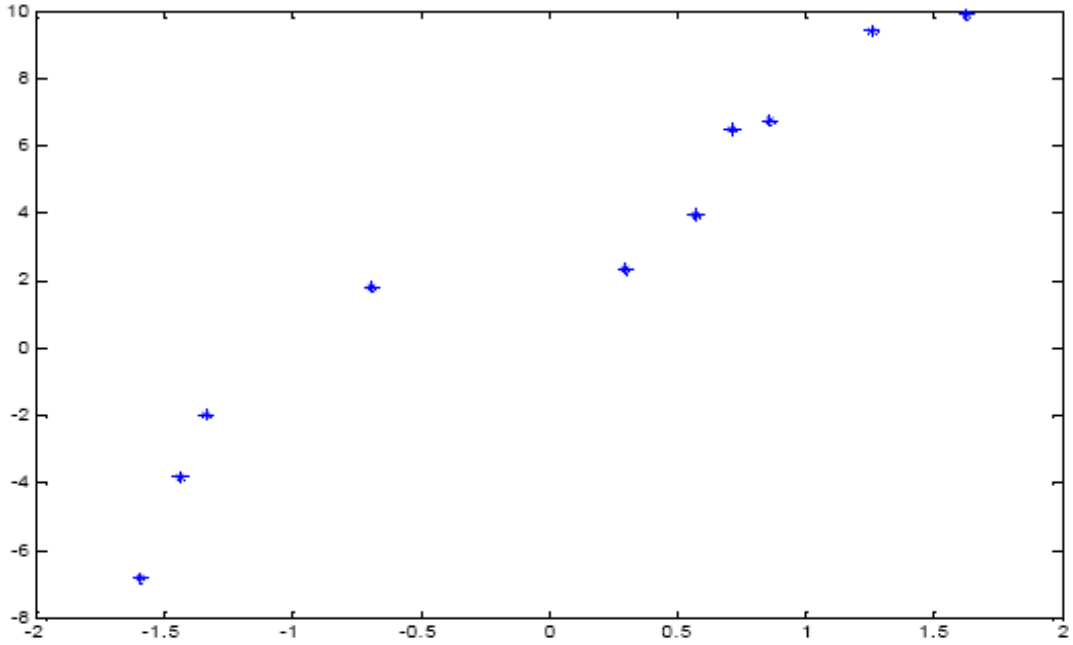
$E_{(x,y)}[(y - f(x))^2]$ Mean squared error

Training error tries to approximate the true error !!!!

Does a good training error imply a good generalization error ?

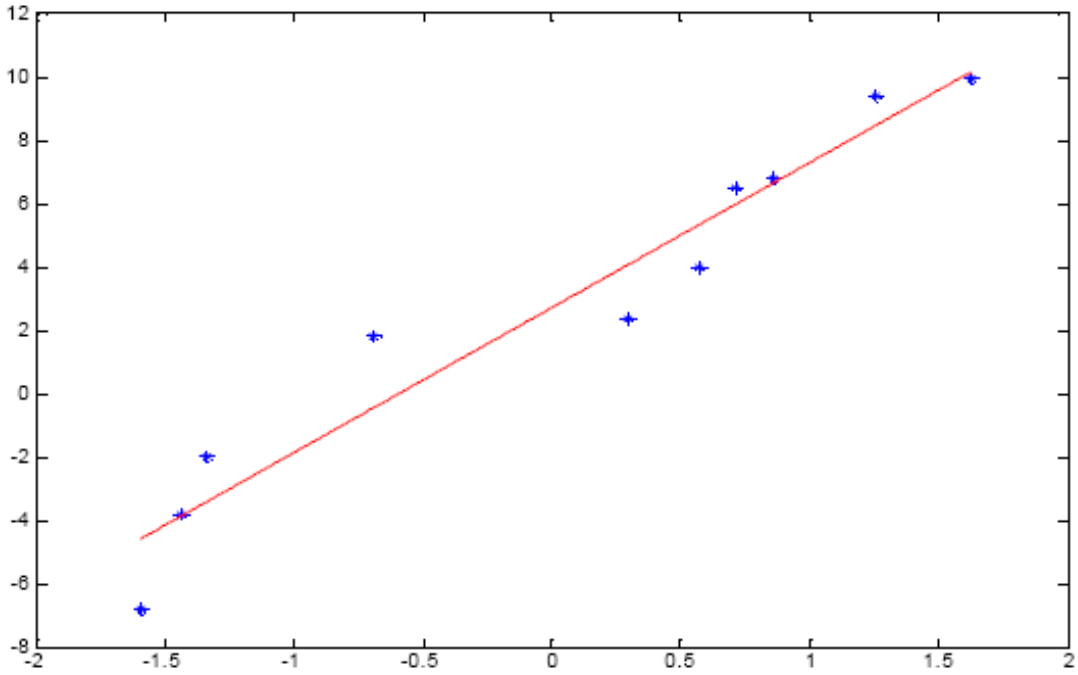
Overfitting

- Assume we have a set of 10 points and we consider polynomial functions as our possible models



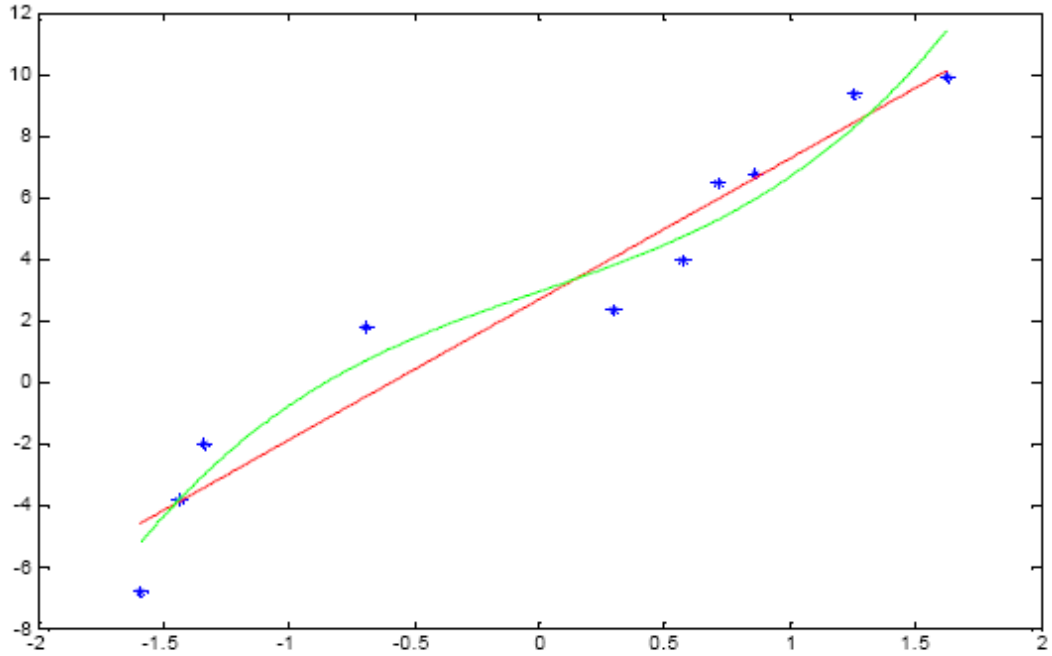
Overfitting

- Fitting a linear function with the square error
- Error is nonzero



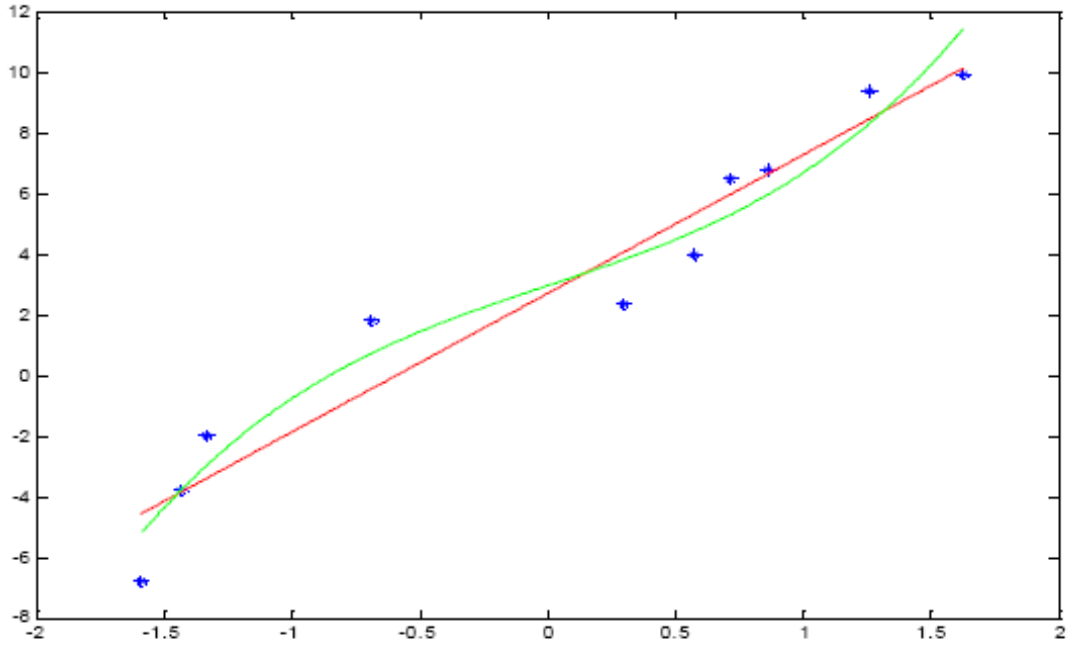
Overfitting

- Linear vs. cubic polynomial
- Higher order polynomial leads to a better fit, smaller error



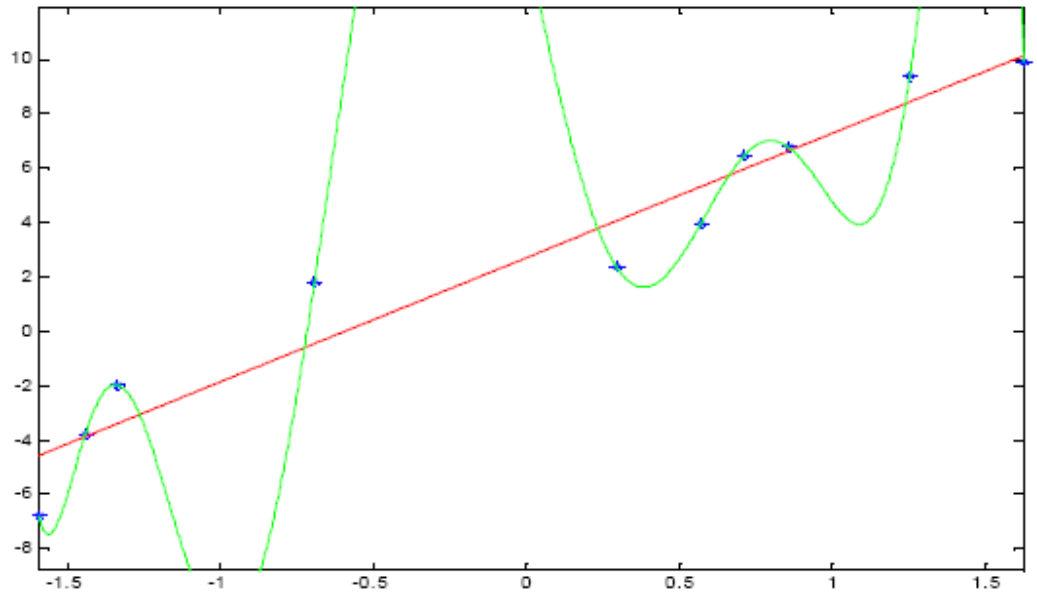
Overfitting

- Is it always good to minimize the error of the observed data?



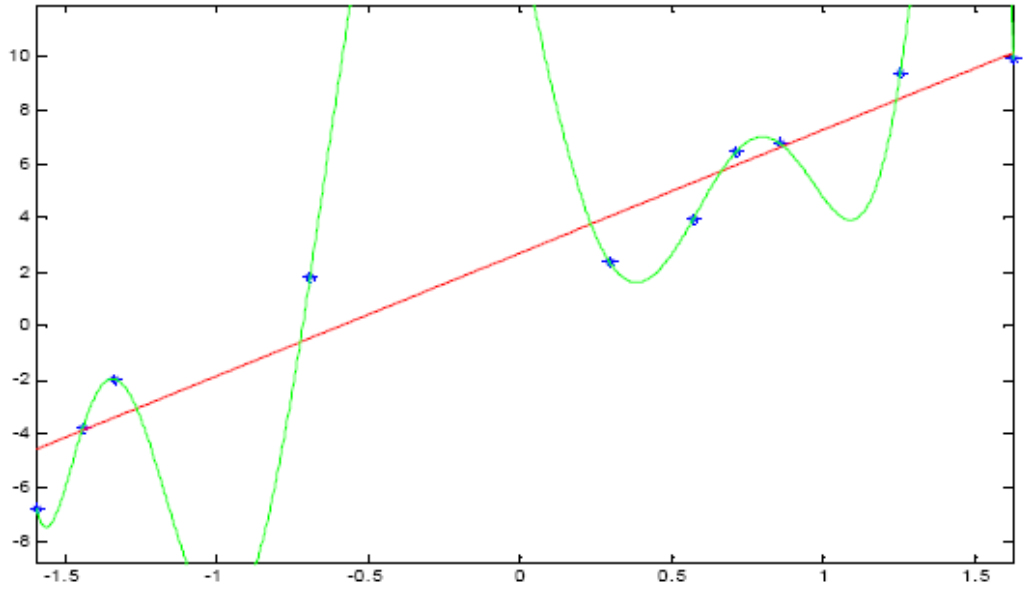
Overfitting

- For 10 data points, the degree 9 polynomial gives a perfect fit (Lagrange interpolation). Error is zero.
- Is it always good to minimize the training error?



Overfitting

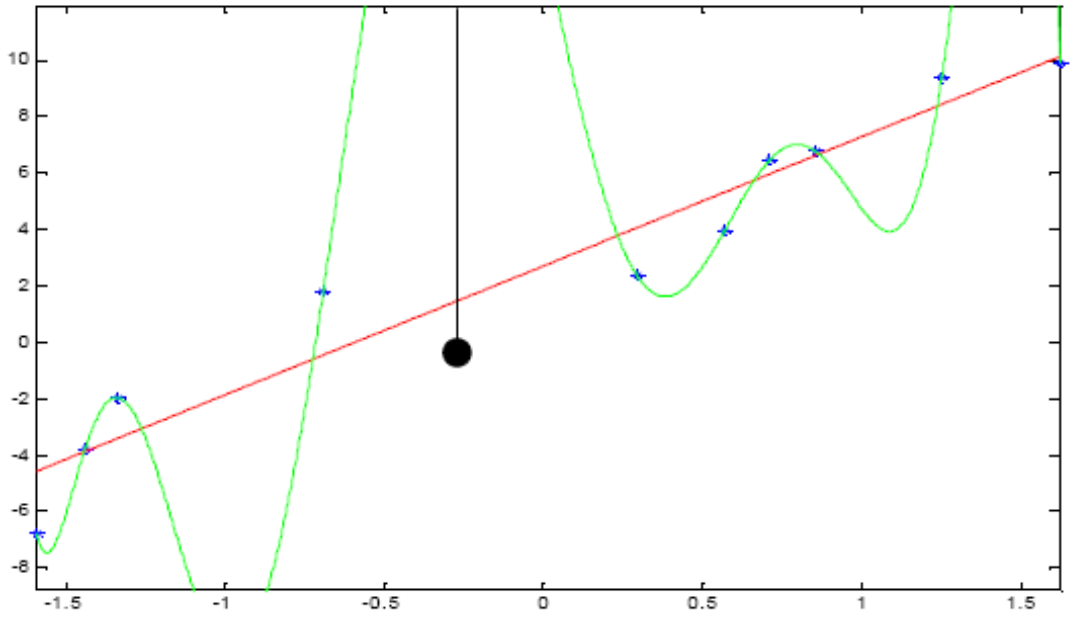
- For 10 data points, degree 9 polynomial gives a perfect fit (Lagrange interpolation). Error is zero.
- Is it always good to minimize the training error? NO !!
- **More important:** How do we perform on the unseen data?



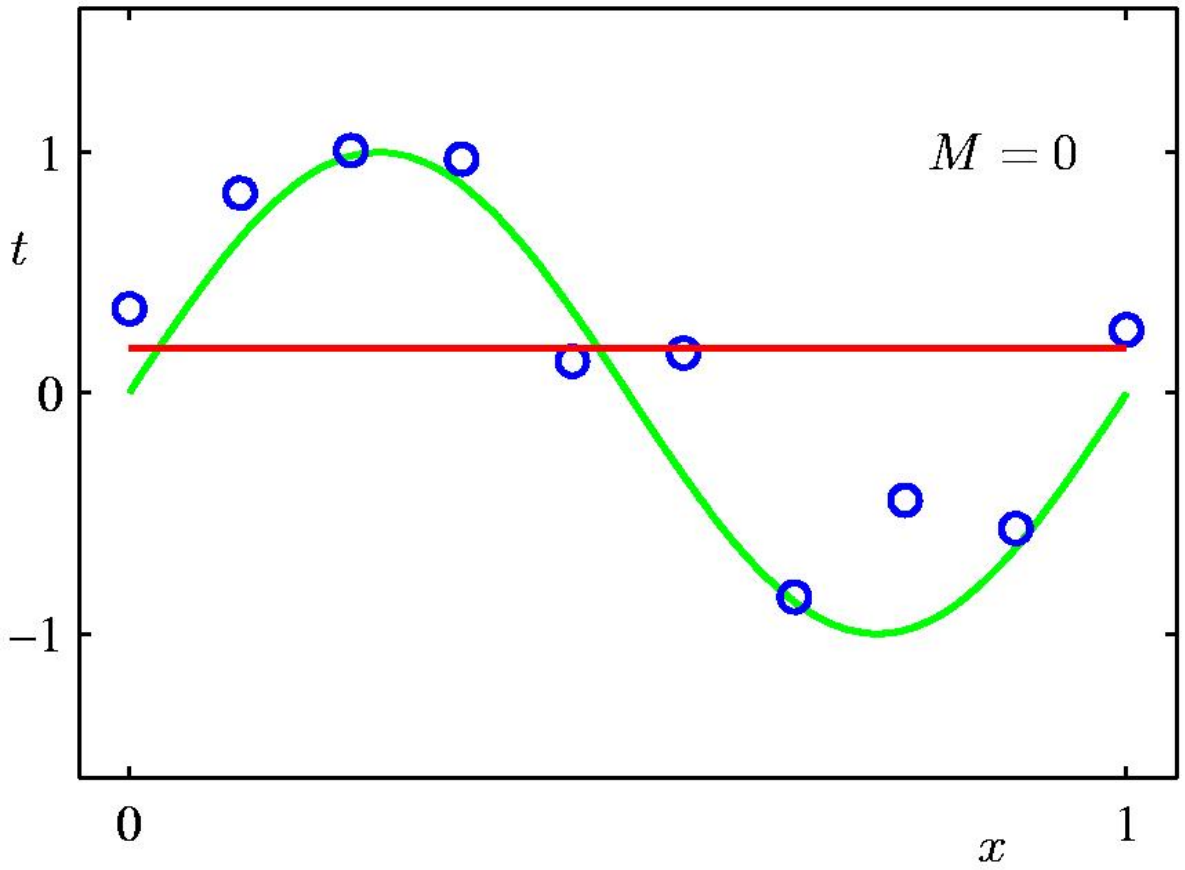
Overfitting

Situation when the training error is low and the generalization error is high. Causes of the phenomenon:

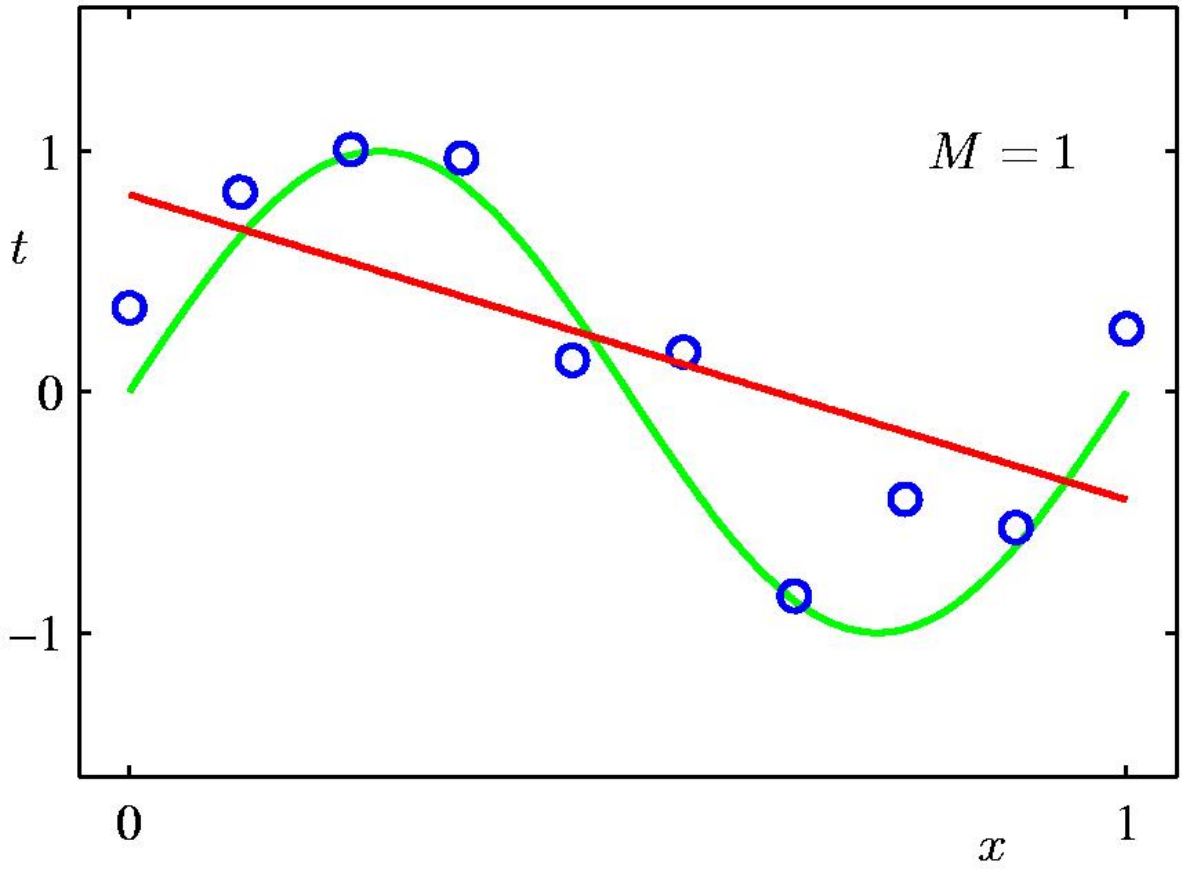
- Model with a large number of parameters (degrees of freedom)
- Small data size (as compared to the complexity of the model)



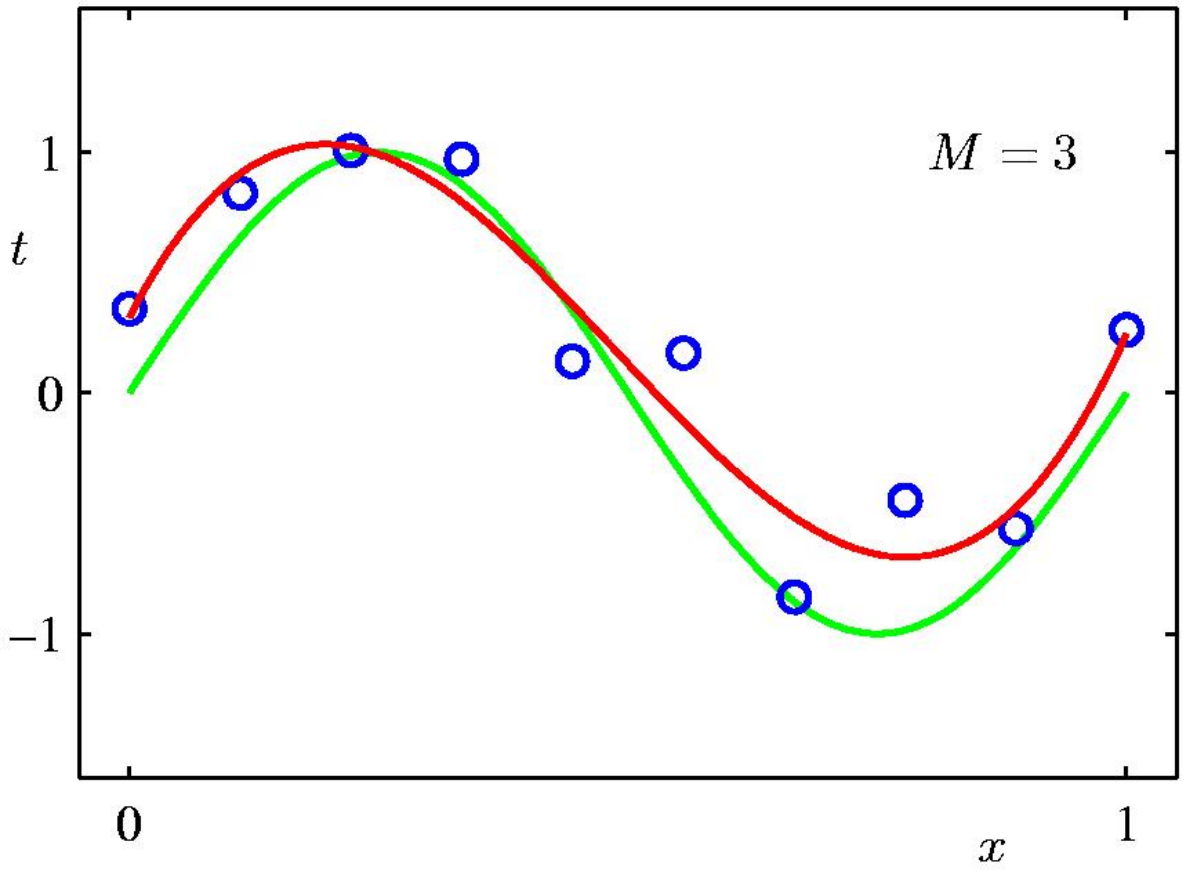
0^{th} Order Polynomial



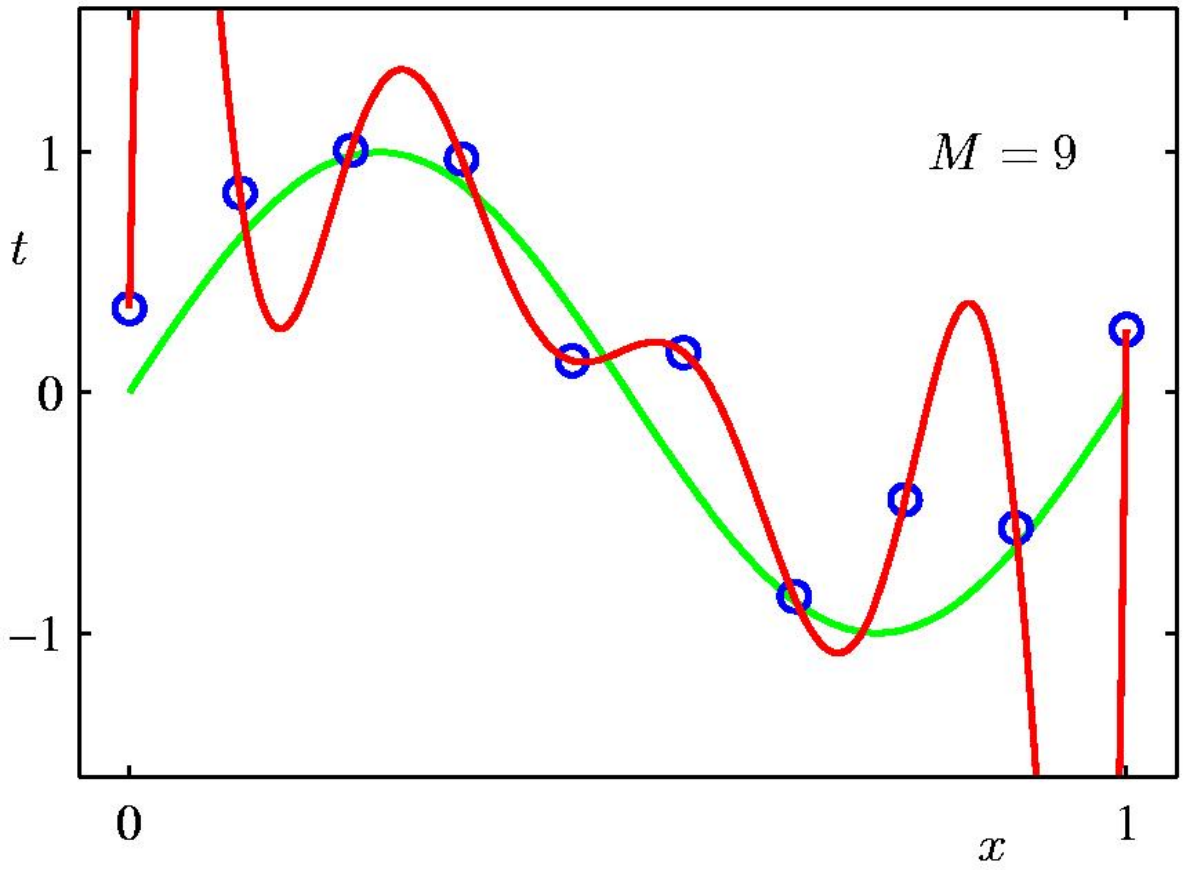
1st Order Polynomial



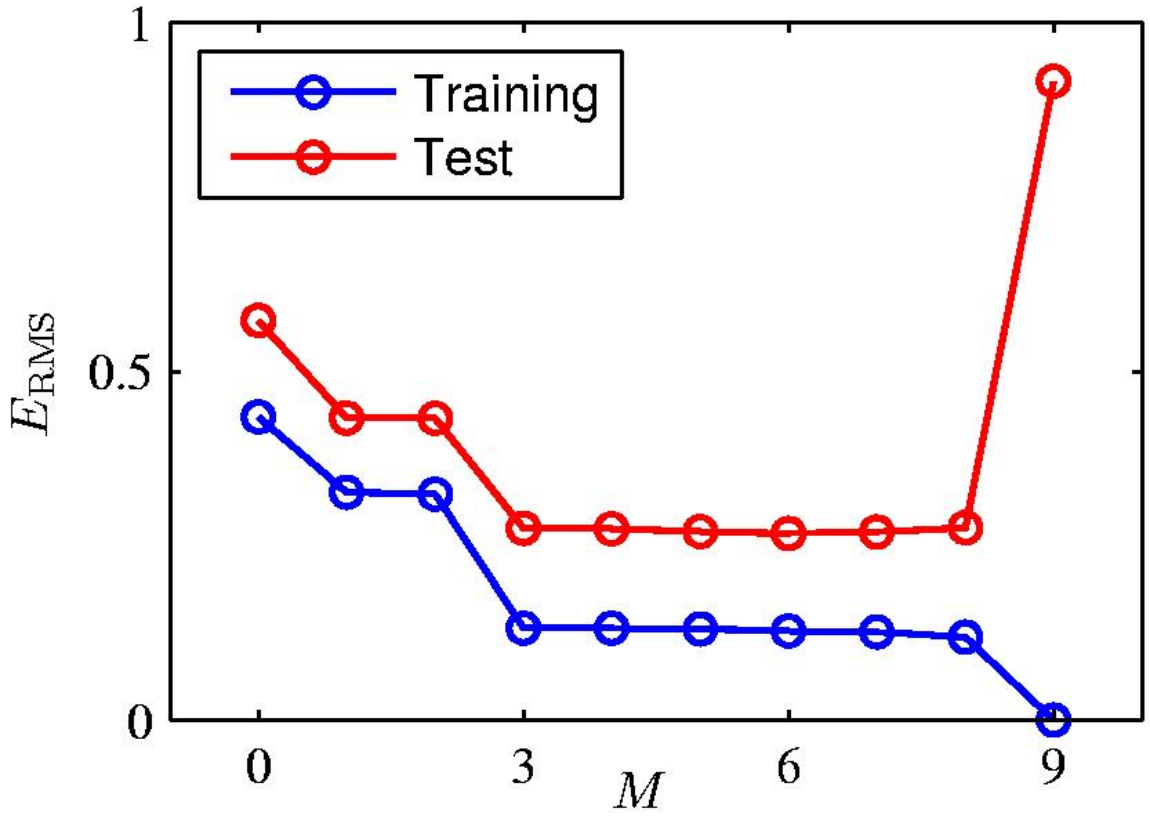
3rd Order Polynomial



9th Order Polynomial



Over-fitting



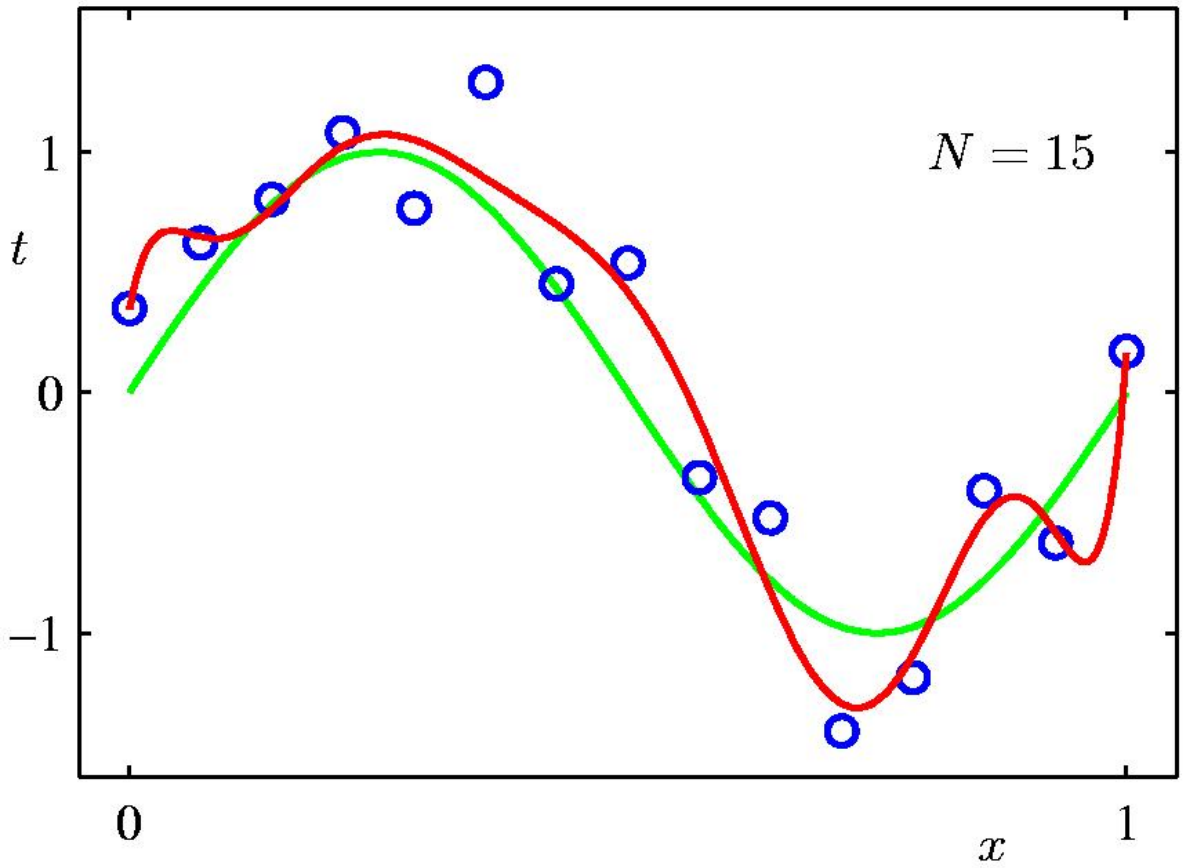
Root-Mean-Square (RMS) Error: $E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$

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

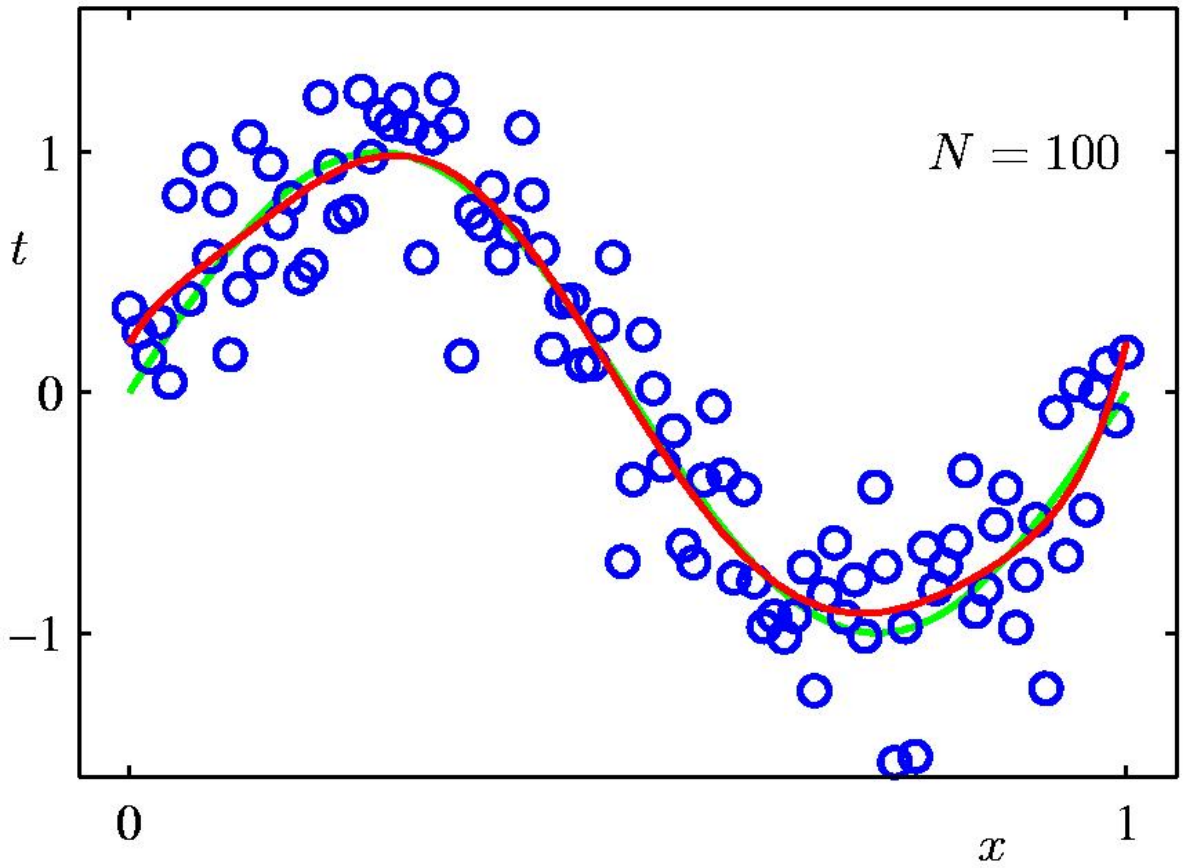
Data Set Size: $N = 15$

9th Order Polynomial



Data Set Size: $N = 100$

9th Order Polynomial



How to evaluate the learner's performance?

- **Generalization error** is the true error for the population of examples we would like to optimize

$$E_{(x,y)}[(y - f(x))^2]$$

- But it cannot be computed exactly
- **Sample mean only approximates the true mean**

- **Optimizing (mean) training error can lead to the overfit, i.e.** training error may not reflect properly the generalization error

$$\frac{1}{n}\sum_{i=1,..n}(y_i - f(x_i))^2$$

- So how to test the generalization error?

How to evaluate the learner's performance?

- **Generalization error** is the true error for the population of examples we would like to optimize
- **Sample mean only approximates it**
- **Two ways to assess the generalization error is:**
  - **Theoretical: Law of Large numbers**
    - statistical bounds on the difference between true and sample mean errors
  - **Practical:** Use a separate data set with m data samples to test the model
    - **(Mean) test error** $\frac{1}{m}\sum_{j=1,..m}(y_j - f(x_j))^2$

Basic experimental setup to test the learner's performance

1. Take a dataset D and divide it into:

- **Training data set**
- **Testing data set**

2. Use the training set and your favorite ML algorithm to train the learner

3. Test (evaluate) the learner on the testing data set

- The results on the testing set can be used to compare different learners powered with different models and learning algorithms