CS60020: Foundations of Algorithm Design and Machine Learning

Sourangshu Bhattacharya

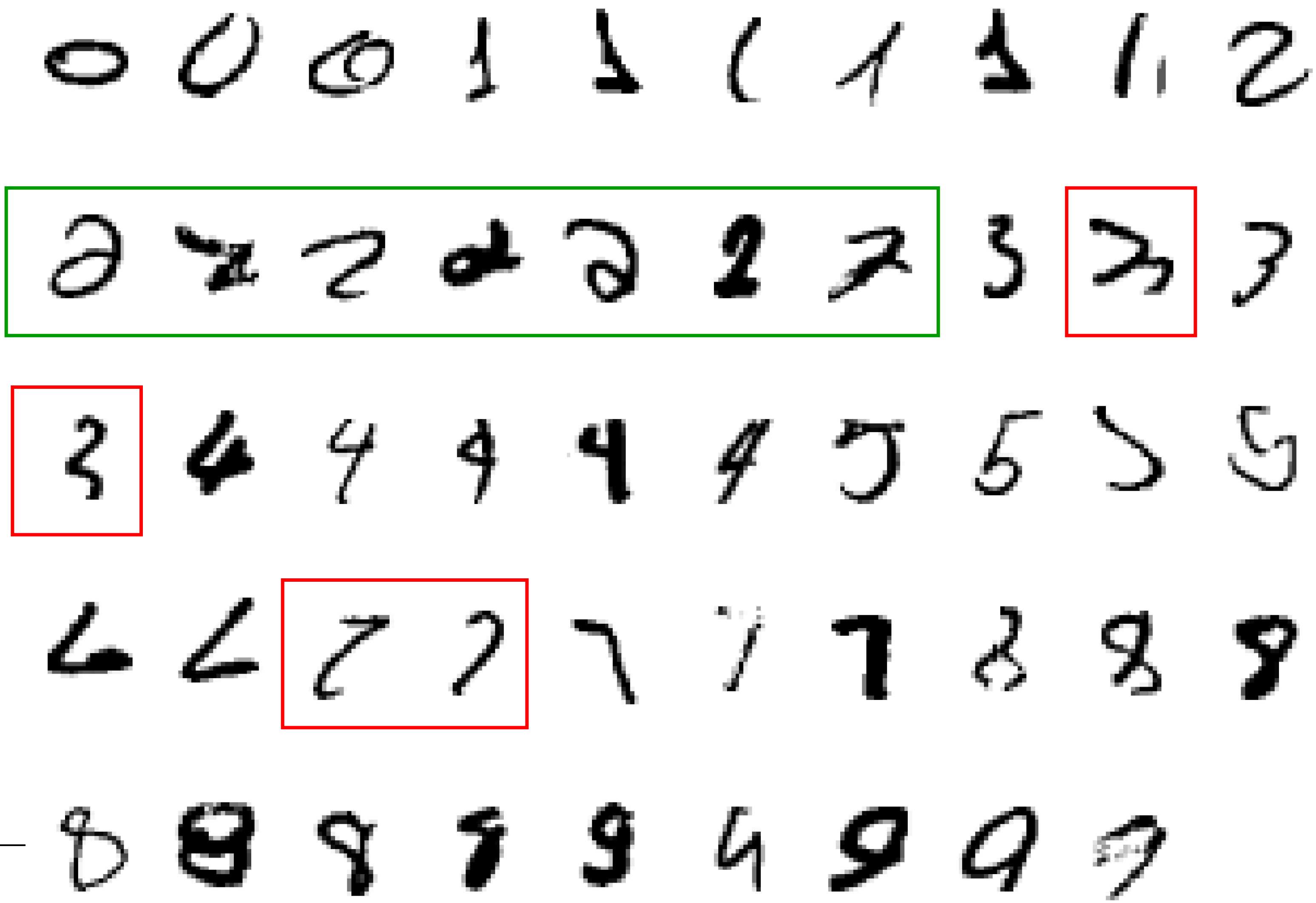
Some slides are taken from Christopher Bishop and Geoffrey Hinton's courses

OVERVIEW

What is Machine Learning?

- It is very hard to write programs that solve problems like recognizing a face.
  - We don't know what program to write because we don't know how our brain does it.
  - Even if we had a good idea about how to do it, the program might be horrendously complicated.
- Instead of writing a program by hand, we collect lots of examples that specify the correct output for a given input.
- A machine learning algorithm then takes these examples and produces a program that does the job.
  - The program produced by the learning algorithm may look very different from a typical hand-written program. It may contain millions of numbers.
  - If we do it right, the program works for new cases as well as the ones we trained it on.

A classic example of a task that requires machine learning: It is very hard to say what makes a 2



Some more examples of tasks that are best solved by using a learning algorithm

Recognizing patterns:

- Facial identities or facial expressions
- Handwritten or spoken words
- Medical images

Generating patterns:

- Generating images or motion sequences

Recognizing anomalies:

- Unusual sequences of credit card transactions
- Unusual patterns of sensor readings in a nuclear power plant or unusual sound in your car engine.

Prediction:

- Future stock prices or currency exchange rates

Some web-based examples of machine learning

The web contains a lot of data. Tasks with very big datasets often use machine learning

especially if the data is noisy or non-stationary.

Spam filtering, fraud detection:

The enemy adapts so we must adapt too.

Recommendation systems:

Lots of noisy data. Million dollar prize!

Information retrieval:

Find documents or images with similar content.

Data Visualization:

Display a huge database in a revealing way

Displaying the structure of a set of documents using Latent Semantic Analysis (a form of PCA)

Each document is converted to a vector of word counts. This vector is then mapped to two coordinates and displayed as a colored dot. The colors represent the hand-labeled classes.

When the documents are laid out in 2-D, the classes are not used. So we can judge how good the algorithm is by seeing if the classes are separated.

Machine Learning & Symbolic AI

Knowledge Representation works with facts/assertions and develops rules of logical inference. The rules can handle quantifiers. Learning and uncertainty are usually ignored.

Expert Systems used logical rules or conditional probabilities provided by “experts” for specific domains.

Graphical Models treat uncertainty properly and allow learning (but they often ignore quantifiers and use a fixed set of variables)

- Set of logical assertions → values of a subset of the variables and local models of the probabilistic interactions between variables.
- Logical inference → probability distributions over subsets of the unobserved variables (or individual ones)
- Learning = refining the local models of the interactions.

Machine Learning & Statistics

A lot of machine learning is just a rediscovery of things that statisticians already knew.

But the emphasis is very different:

- A good piece of statistics: Clever proof that a relatively simple estimation procedure is asymptotically unbiased.
- A good piece of machine learning: Demonstration that a complicated algorithm produces impressive results on a specific task.

Data-mining: Using very simple machine learning techniques on very large databases because computers are too slow to do anything more interesting with ten billion examples.

A spectrum of machine learning tasks

Statistics------------------------------Artificial Intelligence

Low-dimensional data (e.g. less than 100 dimensions)

Lots of noise in the data

There is not much structure in the data, and what structure there is, can be represented by a fairly simple model.

The main problem is distinguishing true structure from noise.

High-dimensional data (e.g. more than 100 dimensions)

The noise is not sufficient to obscure the structure in the data if we process it right.

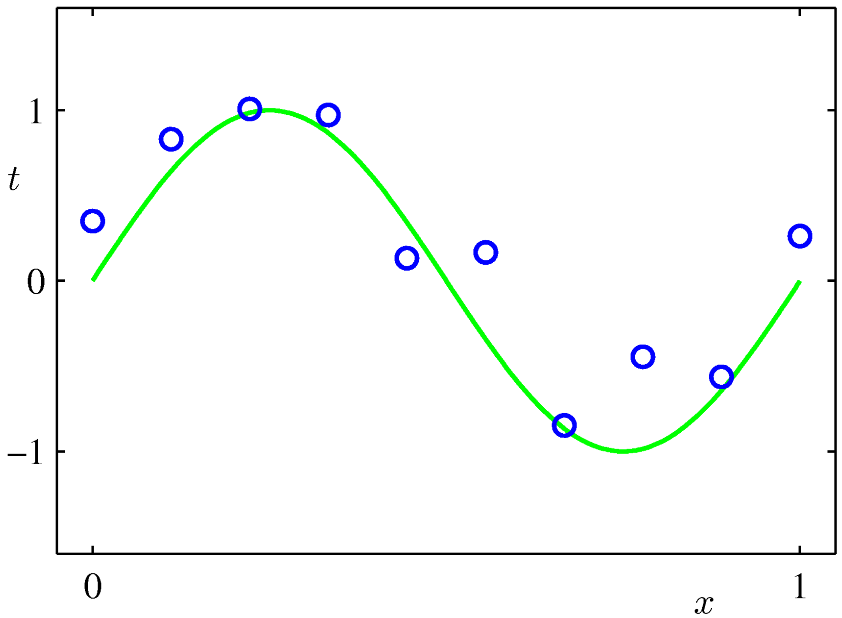
There is a huge amount of structure in the data, but the structure is too complicated to be represented by a simple model.

The main problem is figuring out a way to represent the complicated structure that allows it to be learned.

REGRESSION

Linear Basis Function Models (1)

Example: Polynomial Curve Fitting



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

Linear Basis Function Models (2)

Generally

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

where $\phi_j(\mathbf{x})$ are known as *basis functions*.

Typically, $\phi_0(\mathbf{x}) = 1$, so that w_0 acts as a bias.

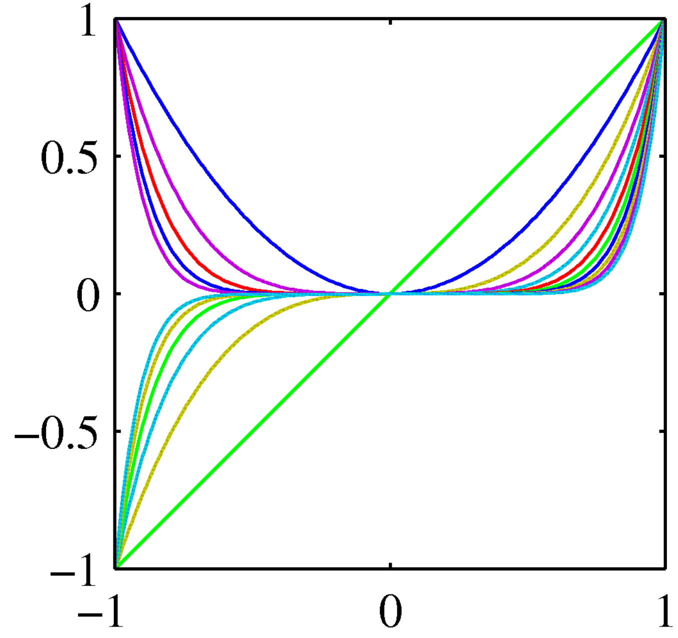
In the simplest case, we use linear basis functions : $\phi_d(\mathbf{x}) = x_d$.

Linear Basis Function Models (3)

Polynomial basis functions:

$$\phi_j(x) = x^j.$$

These are global; a small change in x affect all basis functions.

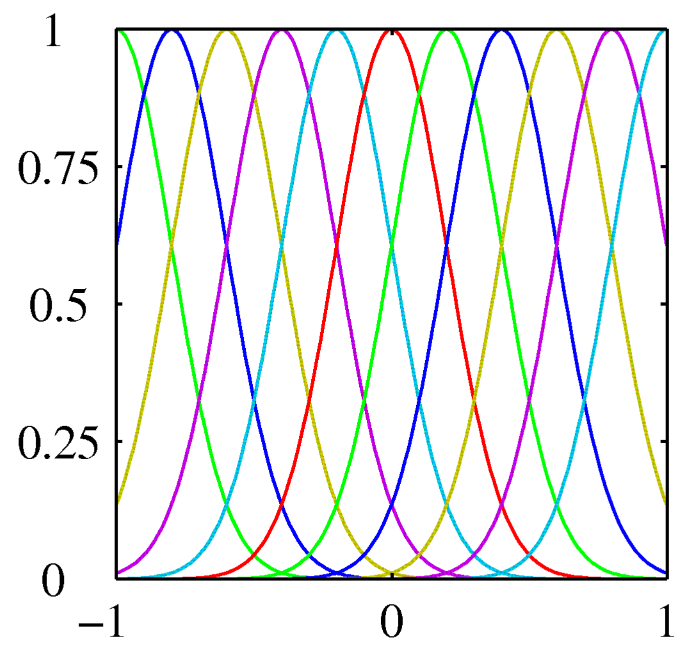


Linear Basis Function Models (4)

Gaussian basis functions:

$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$

These are local; a small change in x only affect nearby basis functions. μ_j and s control location and scale (width).



Linear Basis Function Models (5)

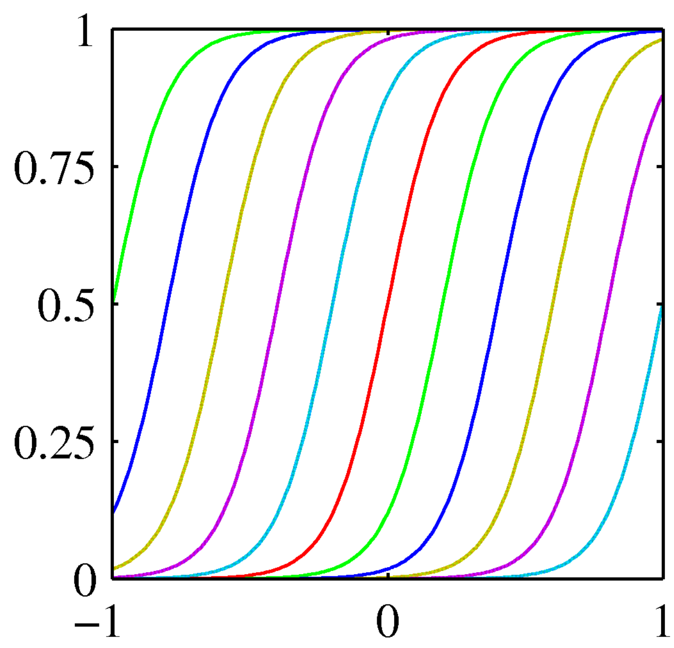
Sigmoidal basis functions:

$$\phi_j(x) = \sigma\left(\frac{x - \mu_j}{s}\right)$$

where

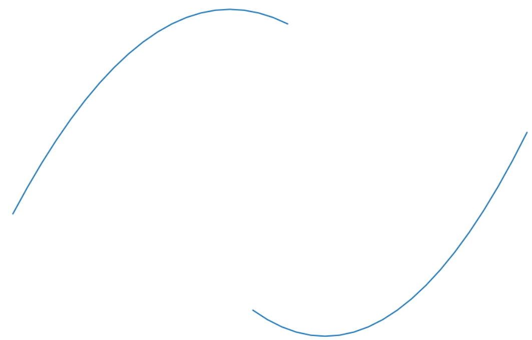
$$\sigma(a) = \frac{1}{1 + \exp(-a)}.$$

Also these are local; a small change in x only affect nearby basis functions. μ_j and s control location and scale (slope).



Least Squares Estimation

A a polynomial curve is represented by the parameters w.



$f(x) = x - x^2$

$f(x) = x + x^2$

Error (loss) function for a given parameter:

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

Estimate $w^* = \min_w E(w)$

Maximum Likelihood and Least Squares (1)

Assume observations from a deterministic function with added Gaussian noise:

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \qquad \text{where} \qquad p(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1})$$

which is the same as saying,

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}).$$

Given observed inputs, $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, and targets, $\mathbf{t} = [t_1, \ldots, t_N]^{\mathrm{T}}$, we obtain the likelihood function

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}).$$

Maximum Likelihood and Least Squares (2)

Taking the logarithm, we get

$$\begin{aligned}\ln p(\mathbf{t}|\mathbf{w},\beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n),\beta^{-1}) \\ &= \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) - \beta E_D(\mathbf{w})\end{aligned}$$

where

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

is the sum-of-squares error.

Maximum Likelihood and Least Squares (3)

Computing the gradient and setting it to zero yields

$$\nabla_{\mathbf{w}} \ln p(\mathbf{t}|\mathbf{w}, \beta) = \beta \sum_{n=1}^{N} \left\{ t_n - \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n) \right\} \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} = \mathbf{0}.$$

Solving for $\mathbf{w}$, we get

$$\mathbf{w}_{\mathrm{ML}} = \underbrace{\left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}}}_{\text{The Moore-Penrose pseudo-inverse, } \boldsymbol{\Phi}^{\dagger}.} \mathbf{t}$$

where

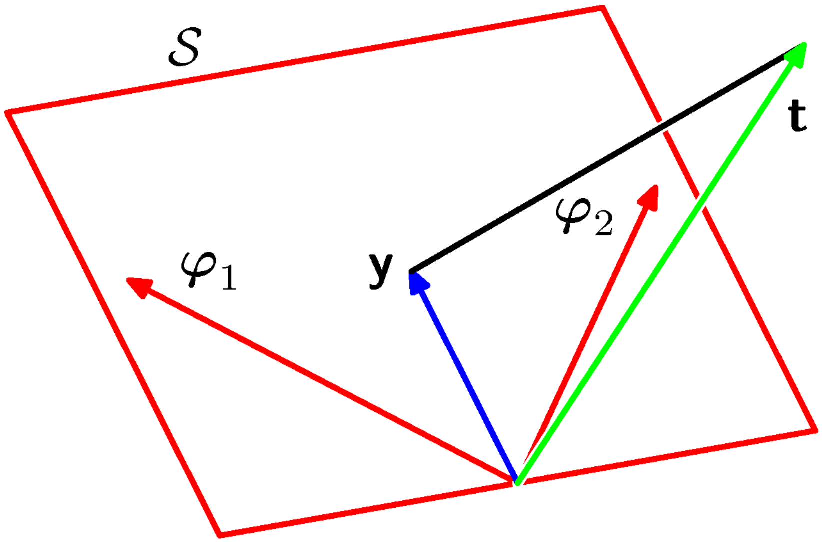
$$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}.$$

Geometry of Least Squares

Consider

$$\mathbf{y} = \boldsymbol{\Phi}\mathbf{w}_{\mathrm{ML}} = [\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_M]\,\mathbf{w}_{\mathrm{ML}}.$$

$$\mathbf{y} \in \underset{M\text{-dimensional}}{\mathcal{S}} \subseteq \underset{N\text{-dimensional}}{\mathcal{T}} \qquad \mathbf{t} \in \mathcal{T}$$



$\mathcal{S}$ is spanned by $\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_M$.

$\mathbf{w}_{\mathrm{ML}}$ minimizes the distance between **t** and its orthogonal projection on $\mathcal{S}$, i.e. **y**.

Normal Equations

$$(\mathbf{A}^T\mathbf{A})\widehat{\beta} = \mathbf{A}^T\mathbf{Y}$$

p xp p x1 p x1

If $(\mathbf{A}^T\mathbf{A})$ is invertible,

$$\boxed{\widehat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \widehat{f}_n^L(X) = X\widehat{\beta}}$$

When is $(\mathbf{A}^T\mathbf{A})$ invertible ?
Recall: Full rank matrices are invertible.

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

Gradient Descent

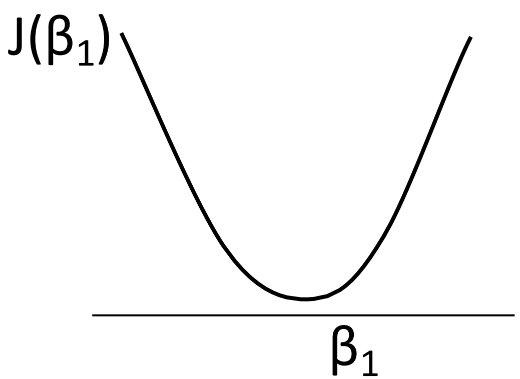
Even when $(\mathbf{A}^T\mathbf{A})$ is invertible, might be computationally expensive if **A** is huge.

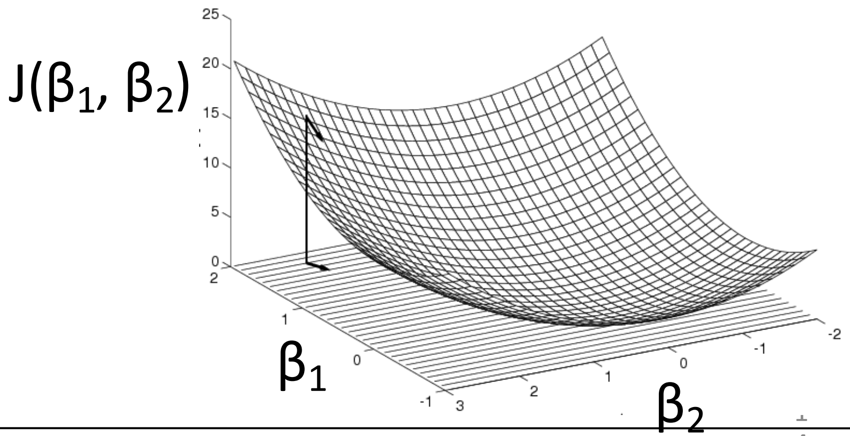
$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) = \arg\min_{\beta} J(\beta)$$

Treat as optimization problem

Observation: J(β) is convex in β.

How to find the minimizer?





Gradient Descent

Even when $(\mathbf{A}^T\mathbf{A})$ is invertible, might be computationally expensive if **A** is huge.

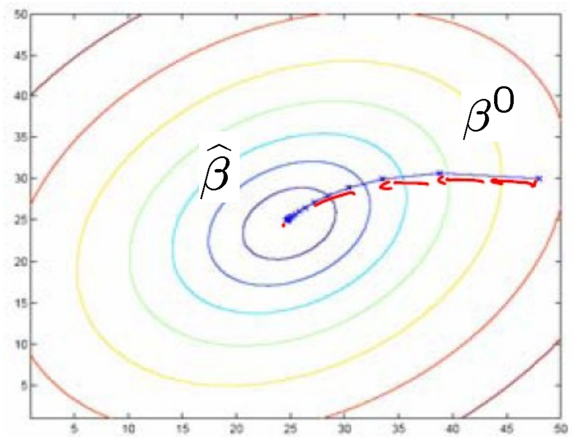
$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) = \arg\min_{\beta} J(\beta)$$

Since J(β) is convex, move along negative of gradient

Initialize: β^0

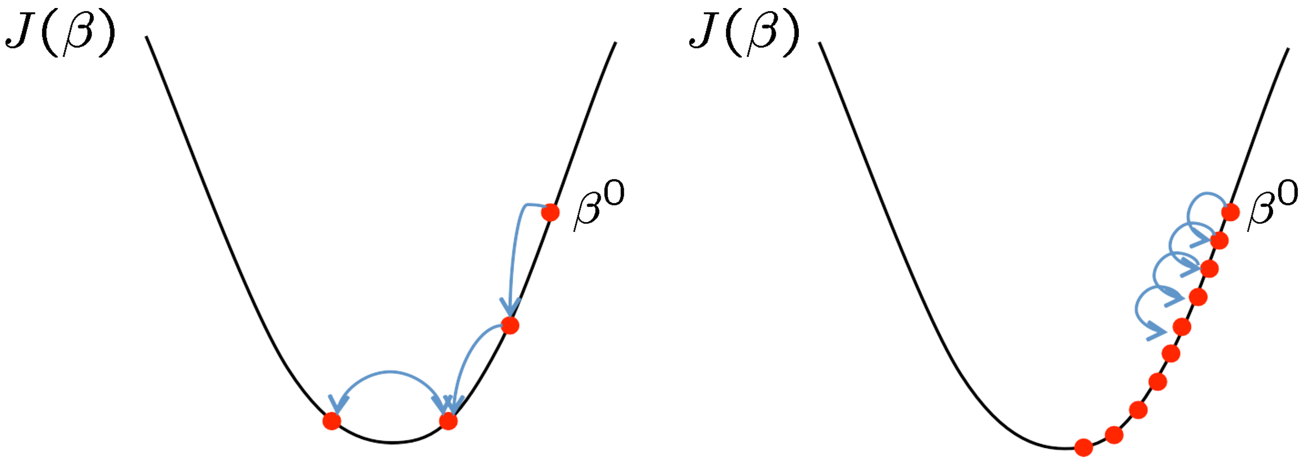
step size

$$\text{Update:} \quad \beta^{t+1} = \beta^t - \frac{\overset{\text{step size}}{\alpha}}{2}\left.\frac{\partial J(\beta)}{\partial \beta}\right|_t = \beta^t - \alpha\, \mathbf{A}^T\underbrace{(\mathbf{A}\beta^t - Y)}_{0 \text{ if } \widehat{\beta} = \beta^t}$$



Stop: when some criterion met e.g. fixed # iterations, or $\left.\frac{\partial J(\beta)}{\partial \beta}\right|_{\beta^t} < \varepsilon$.

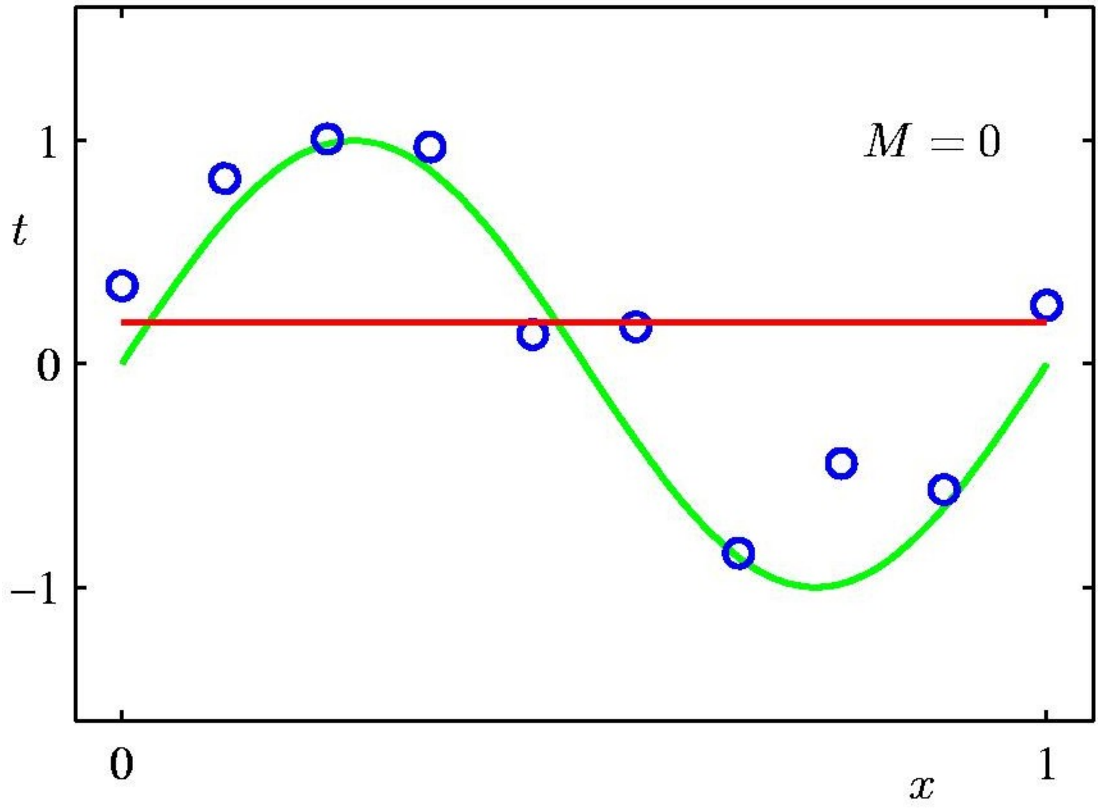
Effect of step-size α



Large α => Fast convergence but larger residual error
Also possible oscillations

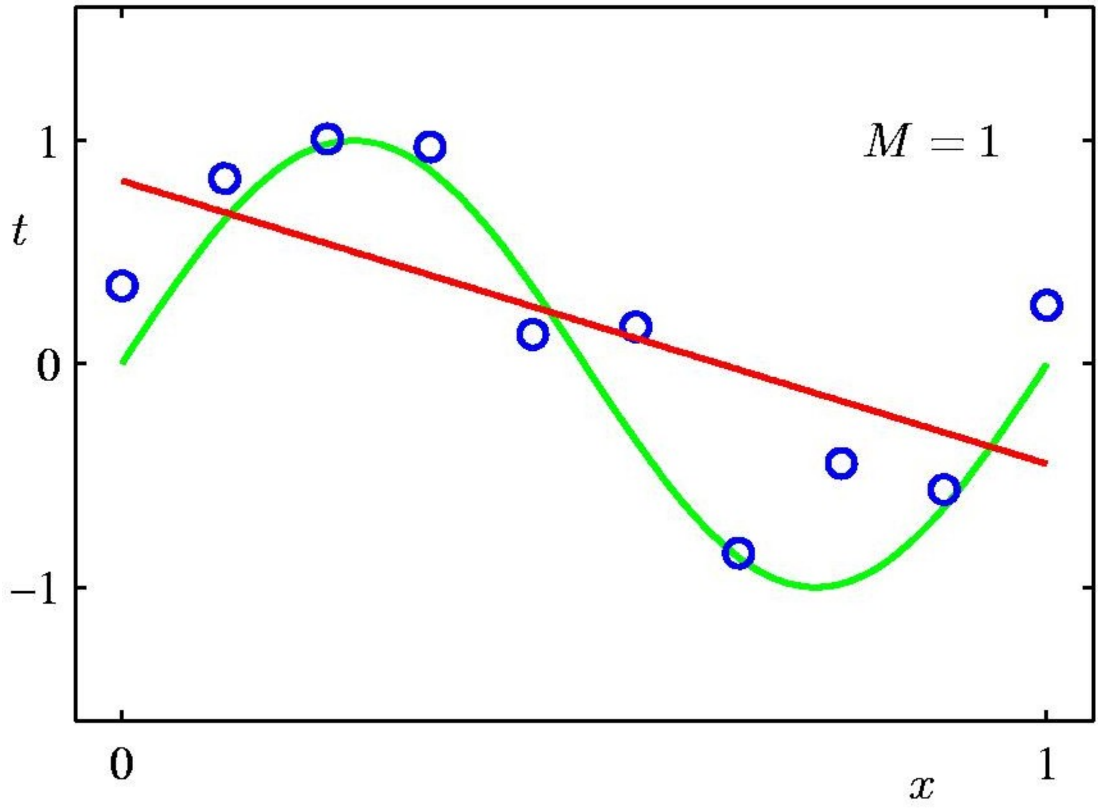
Small α => Slow convergence but small residual error

0th Order Polynomial



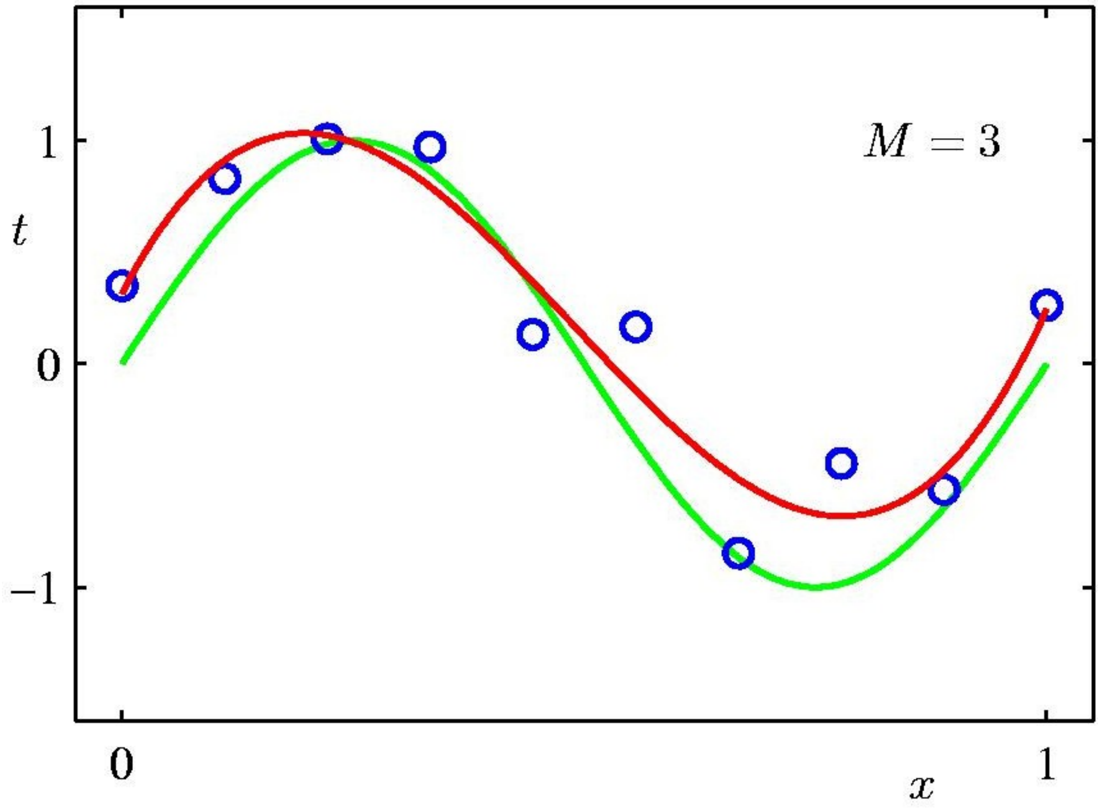
n=10

1st Order Polynomial

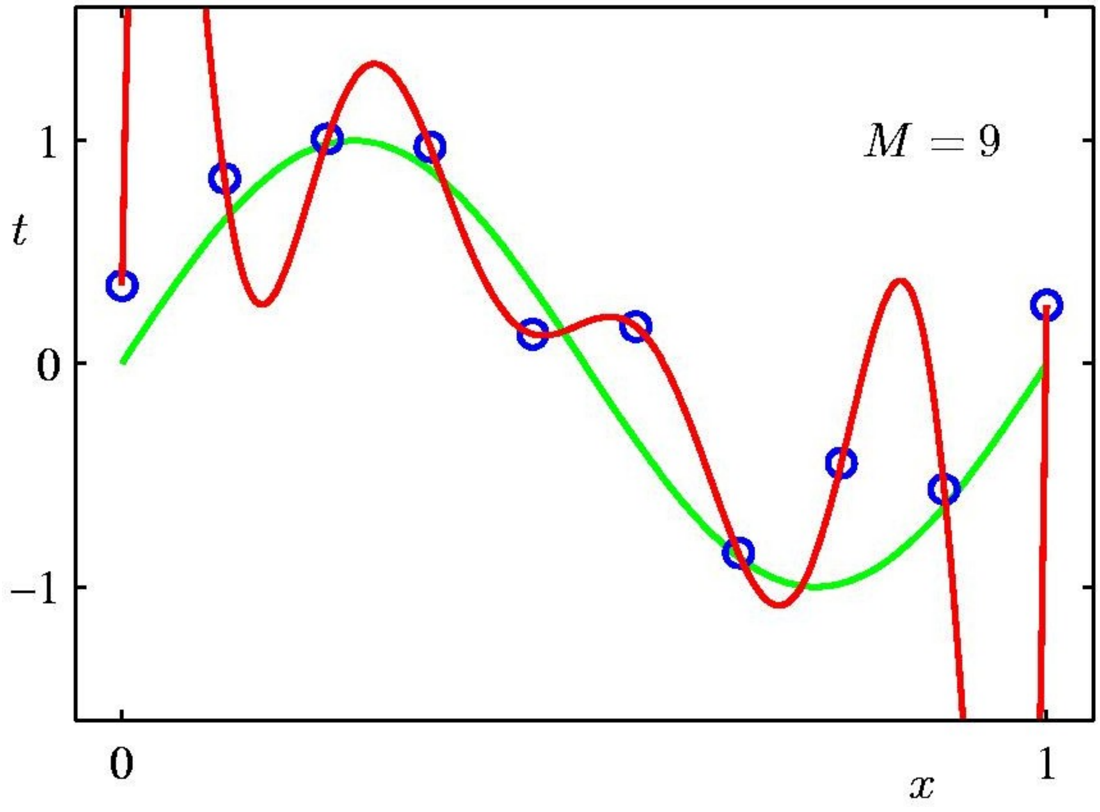


Slide courtesy of William Cohen

3^{rd} Order Polynomial

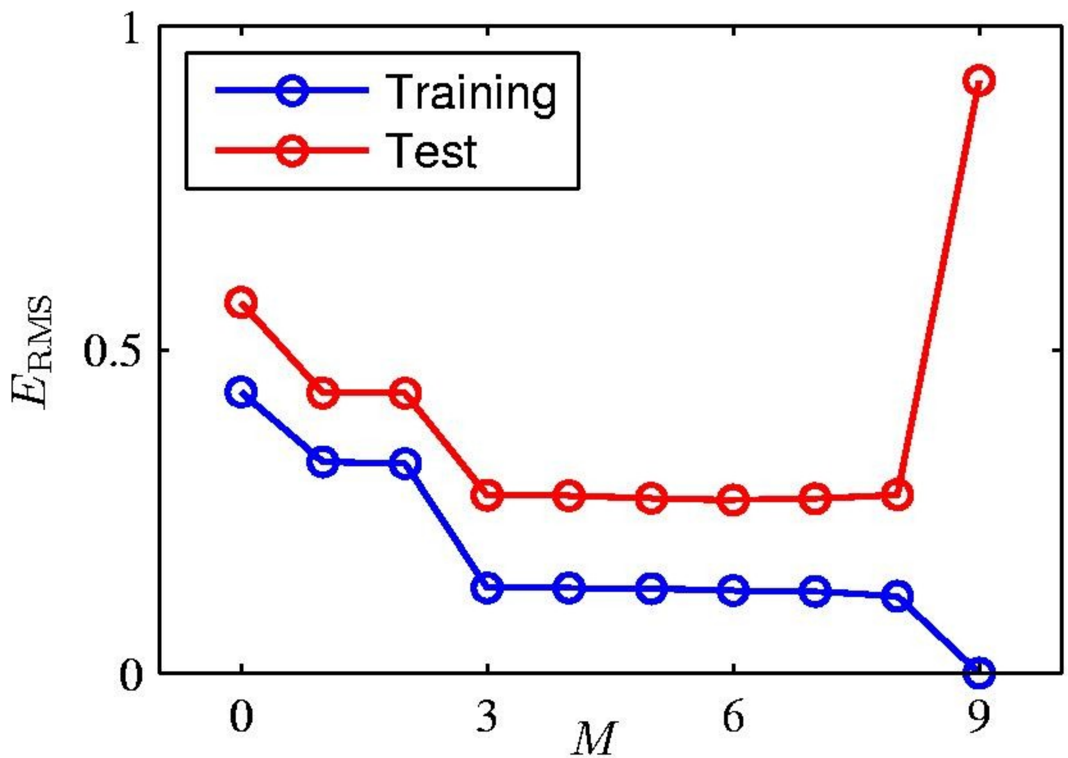


9^{th} Order Polynomial



Slide courtesy of William Cohen

Over-fitting



Root-Mean-Square (RMS) Error

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

Slide courtesy of William Cohen

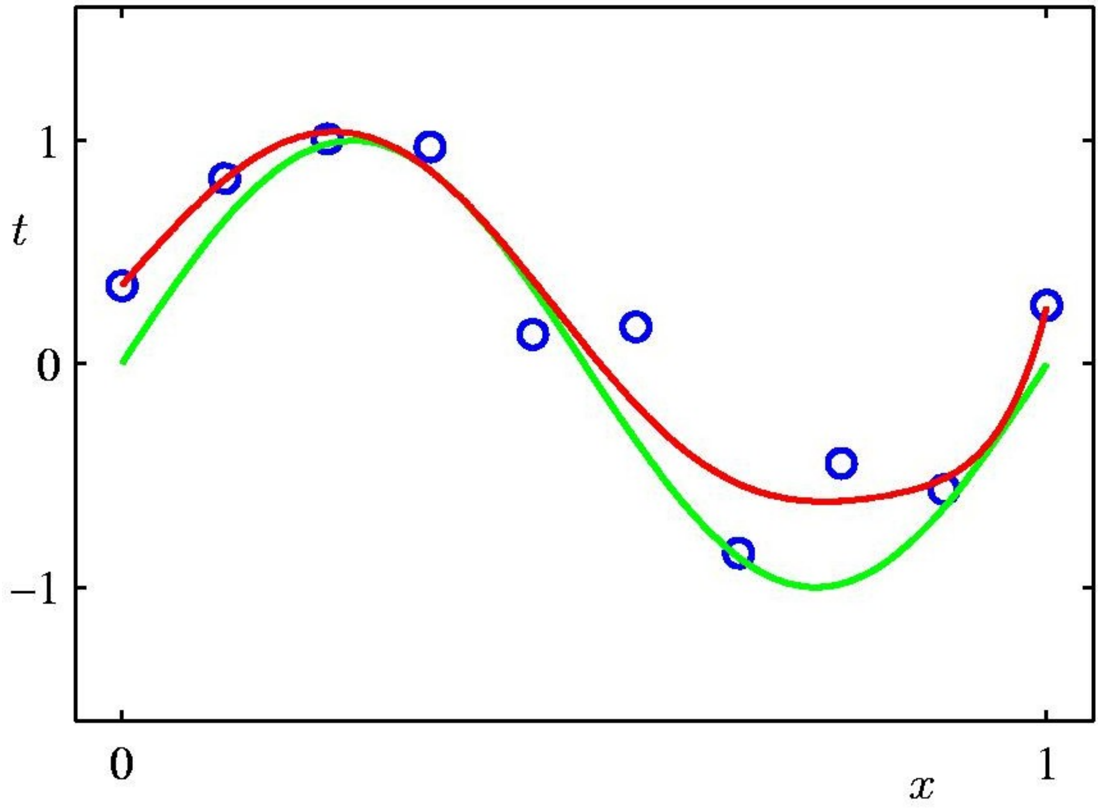
Regularization

Penalize large coefficient values

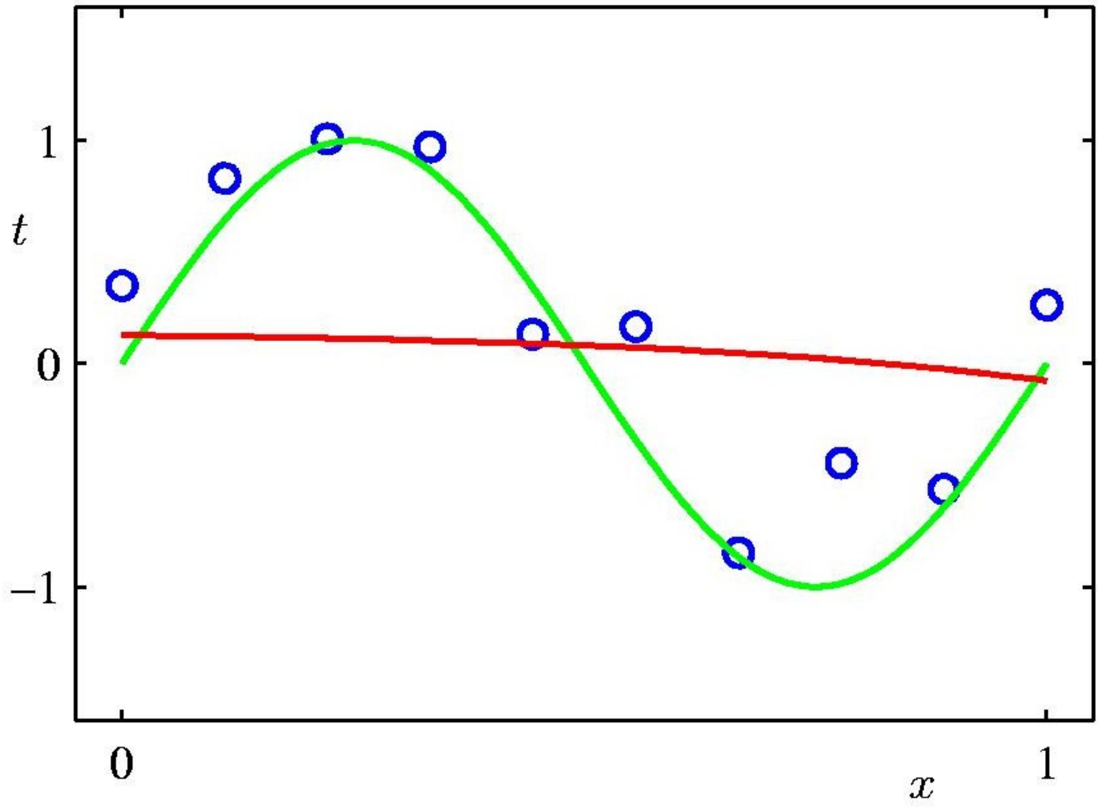
$$J_{\mathbf{X},\mathbf{y}}(\mathbf{w}) = \frac{1}{2}\sum_i \left(y^i - \sum_j w_j \phi_j(\mathbf{x}^i) \right)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

Regularization:

$$\ln \lambda = -18$$



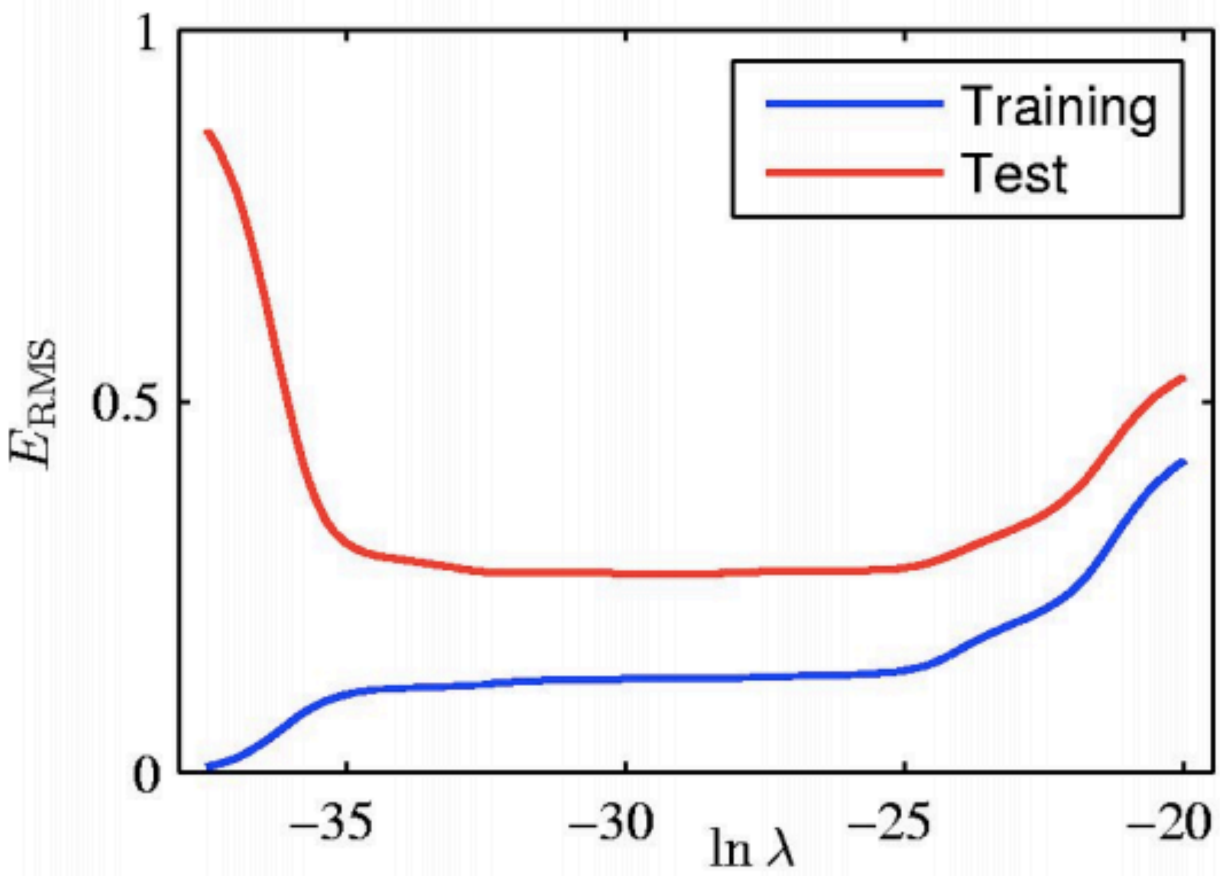
Over Regularization



Slide courtesy of William Cohen

Regularization

9th Order Polynomial



Regularized Least Squares (1)

Consider the error function:

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

Data term + Regularization term

With the sum-of-squares error function and a quadratic regularizer, we get

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$$

which is minimized by

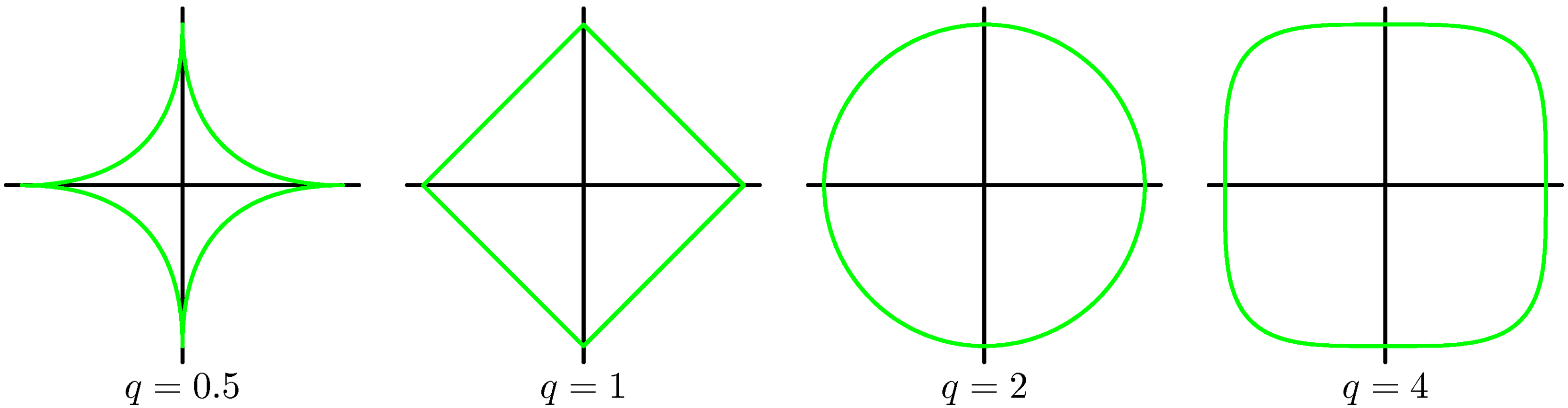
$$\mathbf{w} = \left(\lambda\mathbf{I} + \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}.$$

λ is called the regularization coefficient.

Regularized Least Squares (2)

With a more general regularizer, we have

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$$

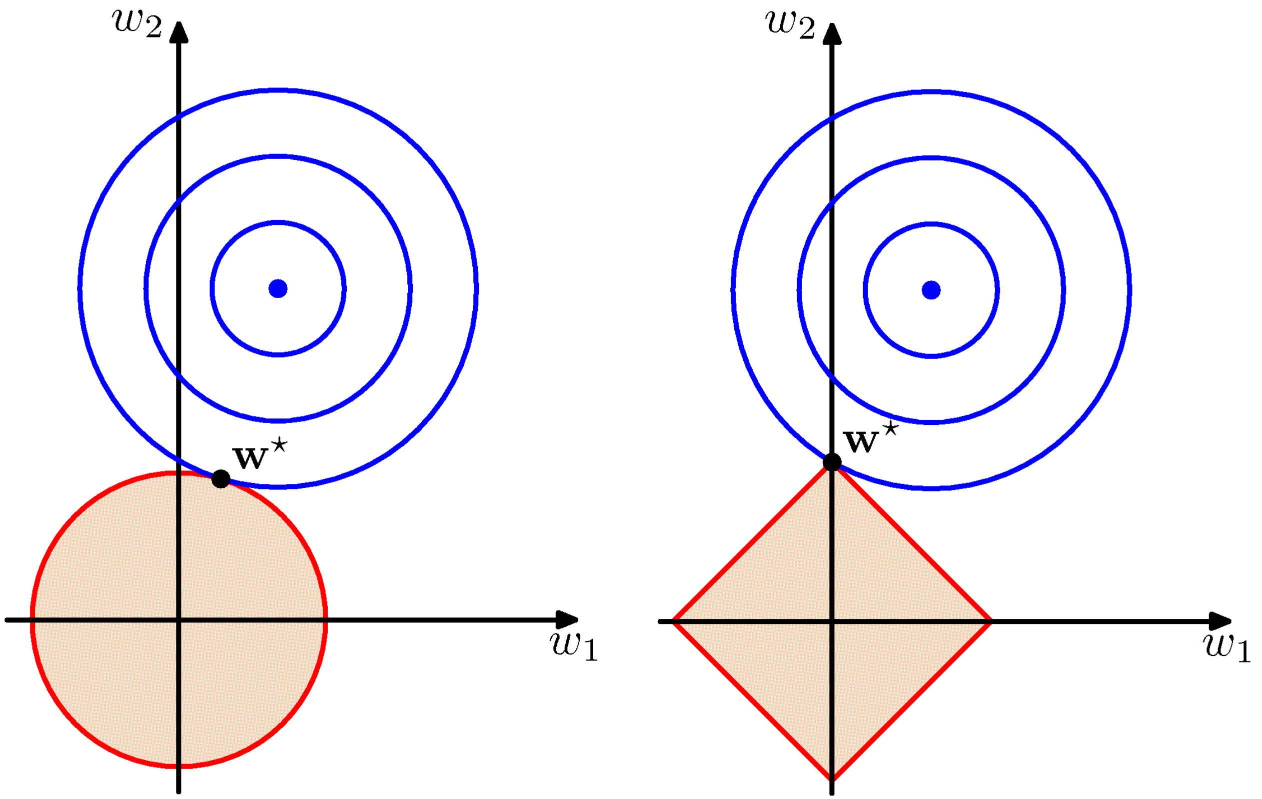


Lasso

Quadratic

Regularized Least Squares (3)

Lasso tends to generate sparser solutions than a quadratic regularizer.



Multiple Outputs (1)

Analogously to the single output case we have:

$$\begin{aligned} p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta) &= \mathcal{N}(\mathbf{t}|\mathbf{y}(\mathbf{W}, \mathbf{x}), \beta^{-1}\mathbf{I}) \\ &= \mathcal{N}(\mathbf{t}|\mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}), \beta^{-1}\mathbf{I}). \end{aligned}$$

Given observed inputs, $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, and targets, $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_N]^{\mathrm{T}}$, we obtain the log likelihood function

$$\begin{aligned} \ln p(\mathbf{T}|\mathbf{X}, \mathbf{W}, \beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{t}_n|\mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}\mathbf{I}) \\ &= \frac{NK}{2} \ln\left(\frac{\beta}{2\pi}\right) - \frac{\beta}{2} \sum_{n=1}^{N} \left\| \mathbf{t}_n - \mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n) \right\|^2. \end{aligned}$$

Multiple Outputs (2)

Maximizing with respect to $\mathbf{W}$, we obtain

$$\mathbf{W}_{\mathrm{ML}} = \left(\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{T}.$$

If we consider a single target variable, t_k, we see that

$$\mathbf{w}_k = \left(\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{t}_k = \mathbf{\Phi}^{\dagger}\mathbf{t}_k$$

where $\mathbf{t}_k = [t_{1k}, \ldots, t_{Nk}]^{\mathrm{T}}$, which is identical with the single output case.

CLASSIFICATION

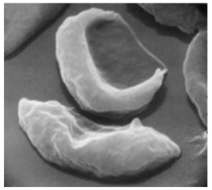
Discrete and Continuous Labels

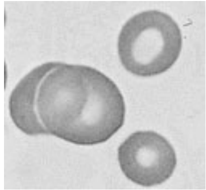
Classification

Sports
Science
News

X = Document

Y = Topic





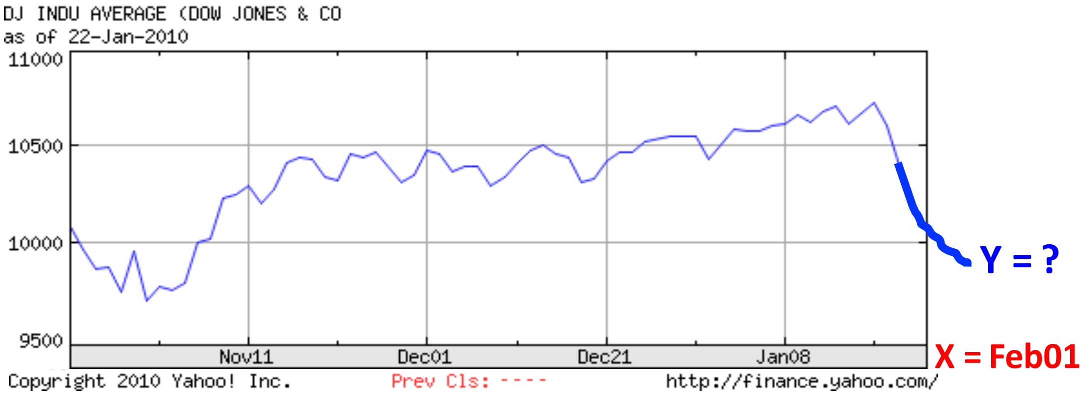
Anemic cell
Healthy cell

X = Cell Image

Y = Diagnosis

Regression

Stock Market Prediction



An example application

An emergency room in a hospital measures 17 variables (e.g., blood pressure, age, etc) of newly admitted patients.

A decision is needed: whether to put a new patient in an intensive-care unit.

Due to the high cost of ICU, those patients who may survive less than a month are given higher priority.

Problem: to predict high-risk patients and discriminate them from low-risk patients.

Another application

A credit card company receives thousands of applications for new cards. Each application contains information about an applicant,

- age
- Marital status
- annual salary
- outstanding debts
- credit rating
- etc.

Problem: to decide whether an application should approved, or to classify applications into two categories, approved and not approved.

The data and the goal

Data: A set of data records (also called examples, instances or cases) described by

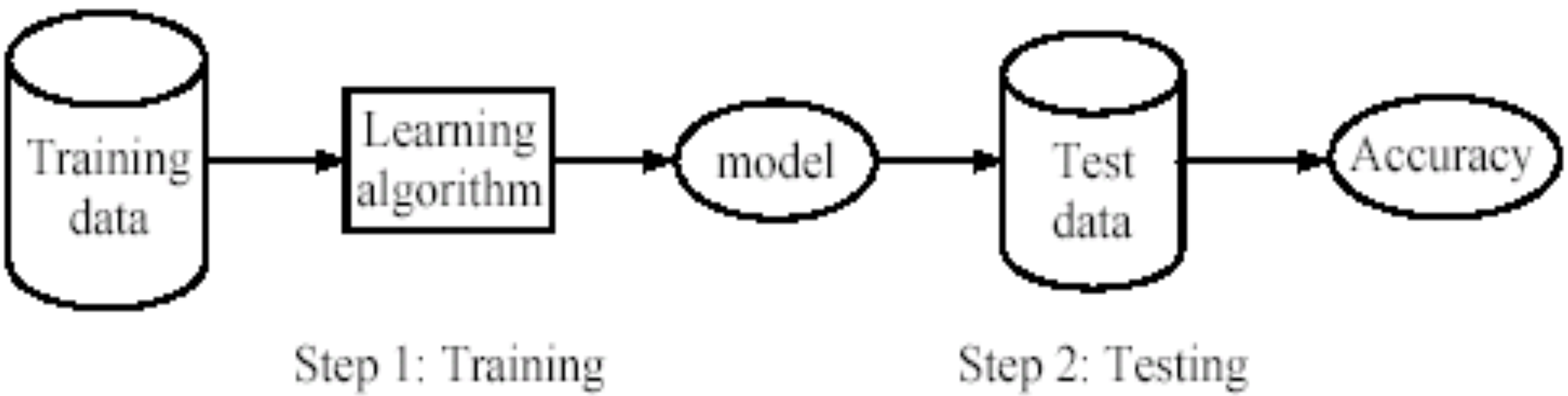
- *k* attributes: A_1, A_2, … A_k.
- a class: Each example is labelled with a pre-defined class.

Goal: To learn a classification model from the data that can be used to predict the classes of new (future, or test) cases/instances.

Supervised learning process: two steps

- Learning (training): Learn a model using the training data
- Testing: Test the model using unseen test data to assess the model accuracy

$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$



Least squares classification

Binary classification.

Each class is described by it's own linear model:

$$y(x) = w^T x + w_0$$

Compactly written as:

$$\mathrm{y}(\boldsymbol{x}) = \boldsymbol{W}^T \boldsymbol{x}$$

W is $[w\ w_0]$.

$$E_D(\boldsymbol{W}) = {}^1\!/_2\, (\boldsymbol{XW} - \boldsymbol{t})^T (\boldsymbol{XW} - \boldsymbol{t})$$

n^{th} row of $\boldsymbol{X}$ is x_n**,** the n^{th} datapoint.

$\boldsymbol{t}$ is vector of +1, -1.

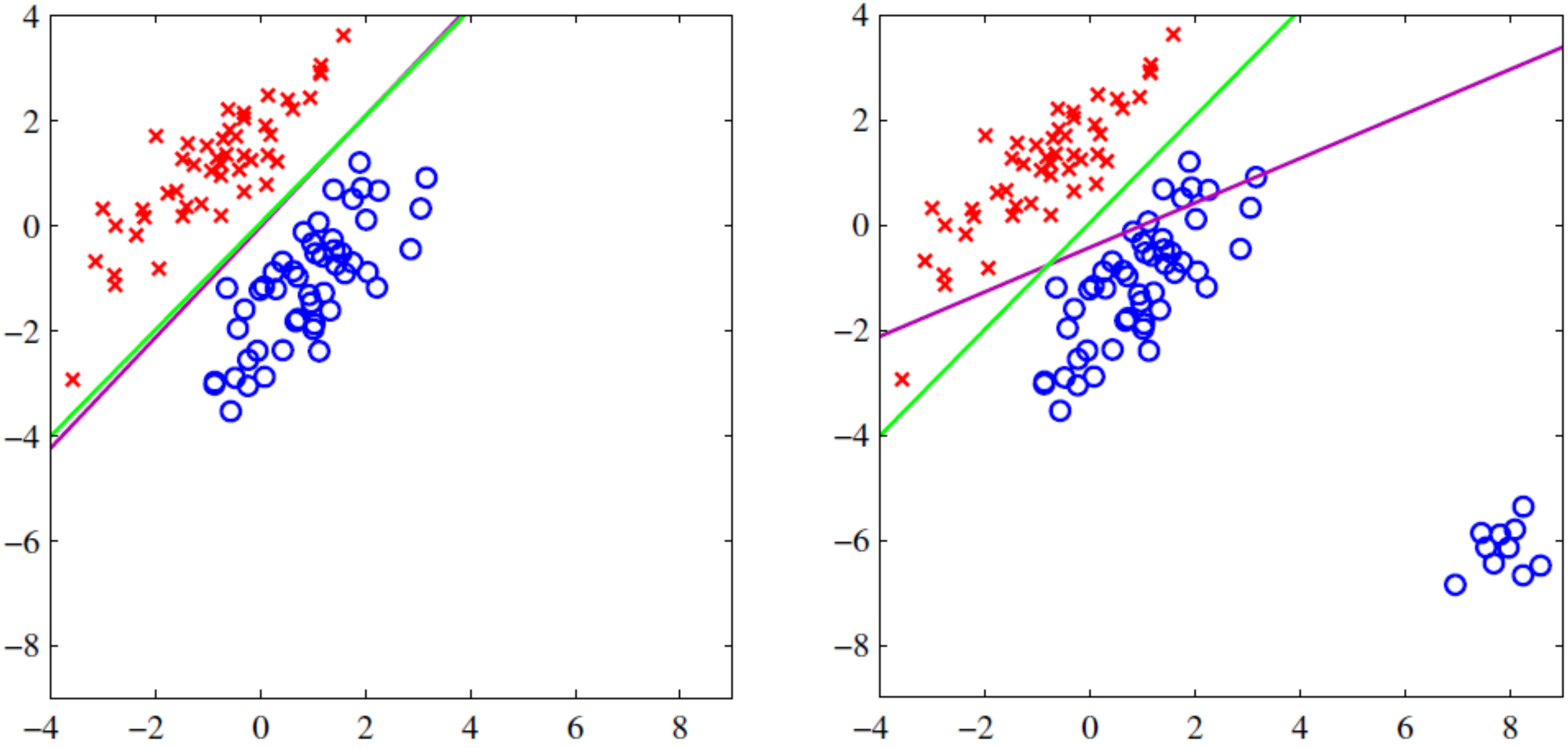
Least squares classification

Least squares $\boldsymbol{W}$ is:

$$\boldsymbol{W} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{t}$$

Problem is affected by outliers.

Least squares classification



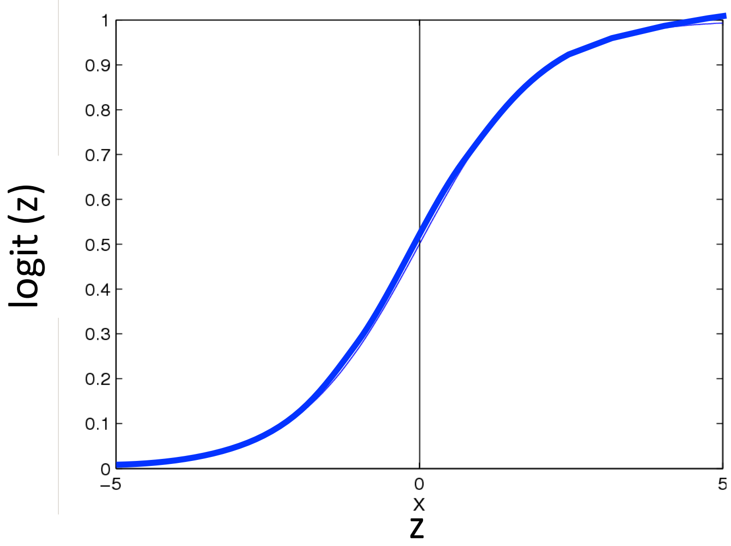
From Linear to Logistic Regression

Assumes the following functional form for P(Y|X):

$$P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_i w_i X_i)}$$

Logistic function applied to a linear function of the data

Logistic function (or Sigmoid): $\frac{1}{1+exp(-z)}$



Features can be discrete or continuous!

Logistic Regression is a Linear Classifier!

Assumes the following functional form for P(Y|X):

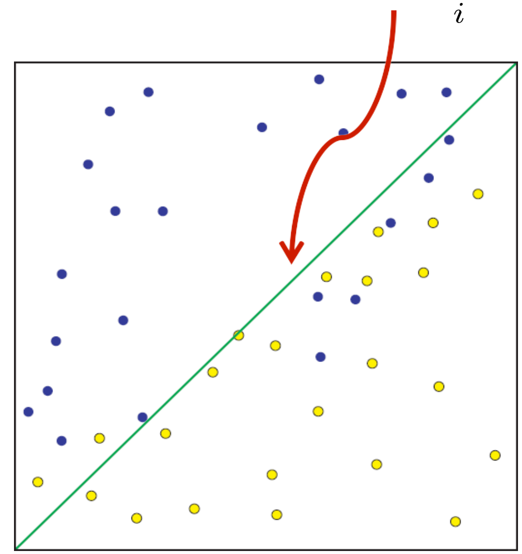
$$P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_i w_i X_i)}$$

$$w_0 + \sum_i w_i X_i = 0$$

Decision boundary:

$$P(Y=0|X) \overset{0}{\underset{1}{\gtrless}} P(Y=1|X)$$

$$w_0 + \sum_i w_i X_i \overset{0}{\underset{1}{\gtrless}} 0$$



(Linear Decision Boundary)

Logistic Regression is a Linear Classifier!

Assumes the following functional form for P(Y|X):

$$P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_i w_i X_i)}$$

$$\Rightarrow P(Y=0|X) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1+\exp(w_0 + \sum_i w_i X_i)}$$

$$\Rightarrow \frac{P(Y=0|X)}{P(Y=1|X)} = \exp(w_0 + \sum_i w_i X_i) \quad \underset{1}{\overset{0}{\gtrless}} \; 1$$

$$\Rightarrow \boxed{w_0 + \sum_i w_i X_i \quad \underset{1}{\overset{0}{\gtrless}} \; 0}$$

Logistic Regression

Label t $\in \{+1, -1\}$modeled as:

$$P(t = 1|x, w) = \sigma(w^T x)$$

$$P(y|x, w) = \sigma(y w^T x), y \in \{+1, -1\}$$

Given a set of parameters w, the probability or likelihood of a datapoint *(x,t):*

$$P(t|x, w) = \sigma(t w^T x)$$

Logistic Regression

Given a training dataset $\{(x_1, t_1), \ldots, (x_N, t_N)\}$, log likelihood of a model w is given by:

$$L(w) = \sum_n \ln(P(t_n | x_n, w))$$

Using principle of maximum likelihood, the best w is given by:

$$w^* = \arg\max_w L(w)$$

Logistic Regression

Final Problem:

$$\max_{w} \sum_{i=1}^{n} -\log(1 + \exp(-t_n w^T x_n))$$

Or, $\min_{w} \sum_{i=1}^{n} \log(1 + \exp(-t_n w^T x_n))$

Error function:

$$E(w) = \sum_{i=1}^{n} \log(1 + \exp(-t_n w^T x_n))$$

$E(w)$ is convex.

Logistic Regression

Final Problem:

$$\max_{w} \sum_{i=1}^{n} -\log(1 + \exp(-t_n w^T x_n))$$

Regularized Version:

$$max \sum_{i=1}^{n} -\log(1 + \exp(-t_n w^T x_n)) - \lambda w^T w$$

Or, $\min_{w} \sum_{i=1}^{n} \log(1 + \exp(-t_n w^T x_n)) + \lambda \left| |w| \right|^2$

Properties of Error function

Derivatives:

$$\nabla E(w) = \sum_{i=1}^{n} -\big(1 - \sigma(t_i w^T x_i)\big)(t_i x_i)$$

$$\nabla E(w) = \sum_{i=1}^{n} (\sigma(w^T x_i) - t_i) x_i$$

$$\nabla^2 E(w) = \sum_{i=1}^{n} \sigma(t_i w^T x_i)\big(1 - \sigma(t_i w^T x_i)\big) x_i x_i^T$$

Gradient Descent

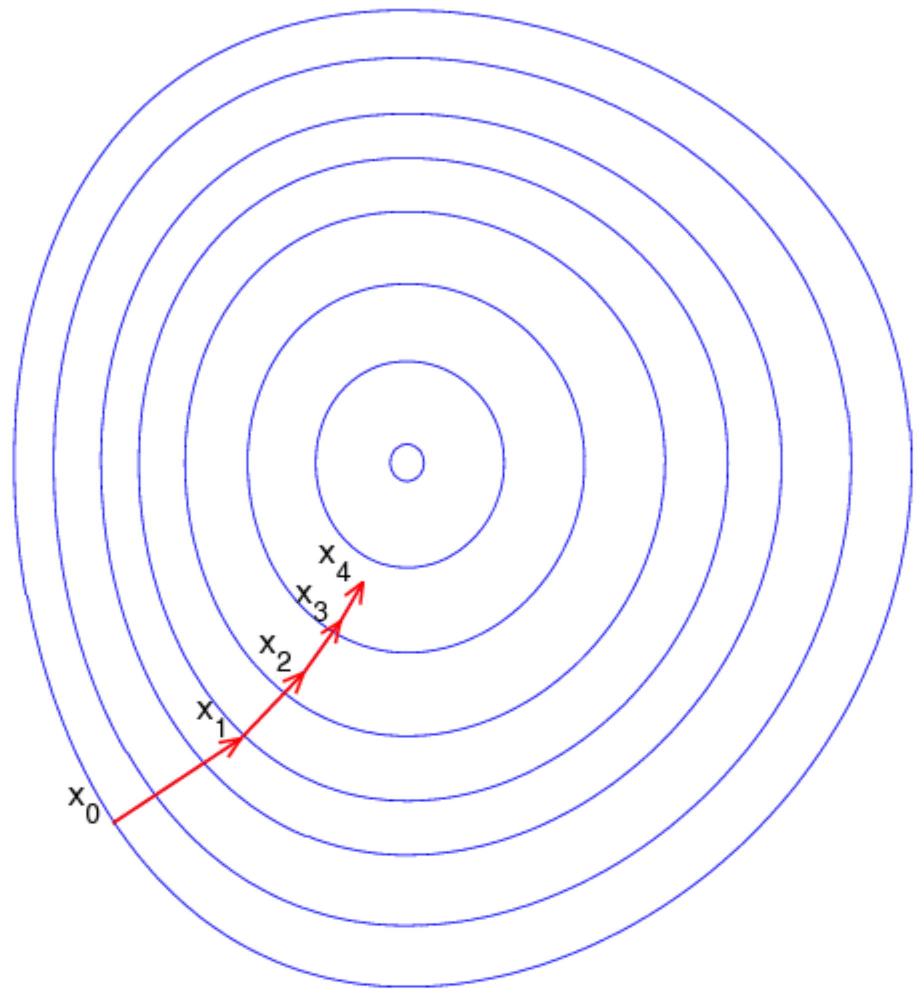
Problem: min f(x)

f(x): differentiable

g(x): gradient of f(x)

Negative gradient is steepest descent direction.

At each step move in the gradient directic
so that there is
"sufficient decrease



Gradient Descent

input : Function f, Gradient ∇f
output: Optimal solution w^*

Initialize $w_0 \leftarrow 0$, $k \leftarrow 0$
while $|\nabla f_k| > \epsilon$ **do**
  Compute $\alpha_k \leftarrow \text{linesearch}(f, -\nabla f_k, w_k)$
  Set $w_{k+1} \leftarrow w_k - \alpha_k \nabla f_k$
  Evaluate ∇f_{k+1}
  $k \leftarrow k + 1$
end
$w^* \leftarrow w_k$

Logistic Regression is a Linear Classifier!

Assumes the following functional form for P(Y|X):

$$P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_i w_i X_i)}$$

$$\Rightarrow P(Y=0|X) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1+\exp(w_0 + \sum_i w_i X_i)}$$

$$\Rightarrow \frac{P(Y=0|X)}{P(Y=1|X)} = \exp(w_0 + \sum_i w_i X_i) \quad \overset{0}{\underset{1}{\gtrless}} \; 1$$

$$\Rightarrow \boxed{w_0 + \sum_i w_i X_i \quad \overset{0}{\underset{1}{\gtrless}} \; 0}$$

Logistic Regression for more than 2 classes

- Logistic regression in more general case, where
  Y $\{y_1,\ldots,y_K\}$

for $k<K$

$$P(Y = y_k|X) = \frac{\exp(w_{k0} + \sum_{i=1}^{d} w_{ki}X_i)}{1 + \sum_{j=1}^{K-1} \exp(w_{j0} + \sum_{i=1}^{d} w_{ji}X_i)}$$

for $k=K$ (normalization, so no weights for this class)

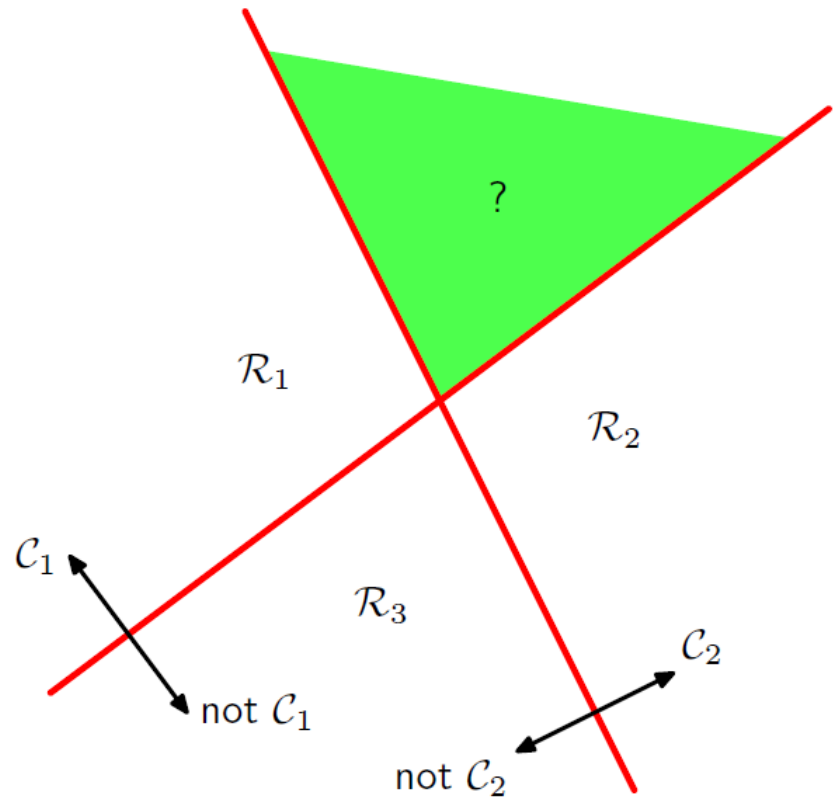
$$P(Y = y_K|X) = \frac{1}{1 + \sum_{j=1}^{K-1} \exp(w_{j0} + \sum_{i=1}^{d} w_{ji}X_i)}$$

Multiple classes

One-vs-all: $K-1$ hyperplanes each separating $C_1, \dots, C_{K-1}$ classes from rest.

Otherwise C_K

Low number of classifiers.

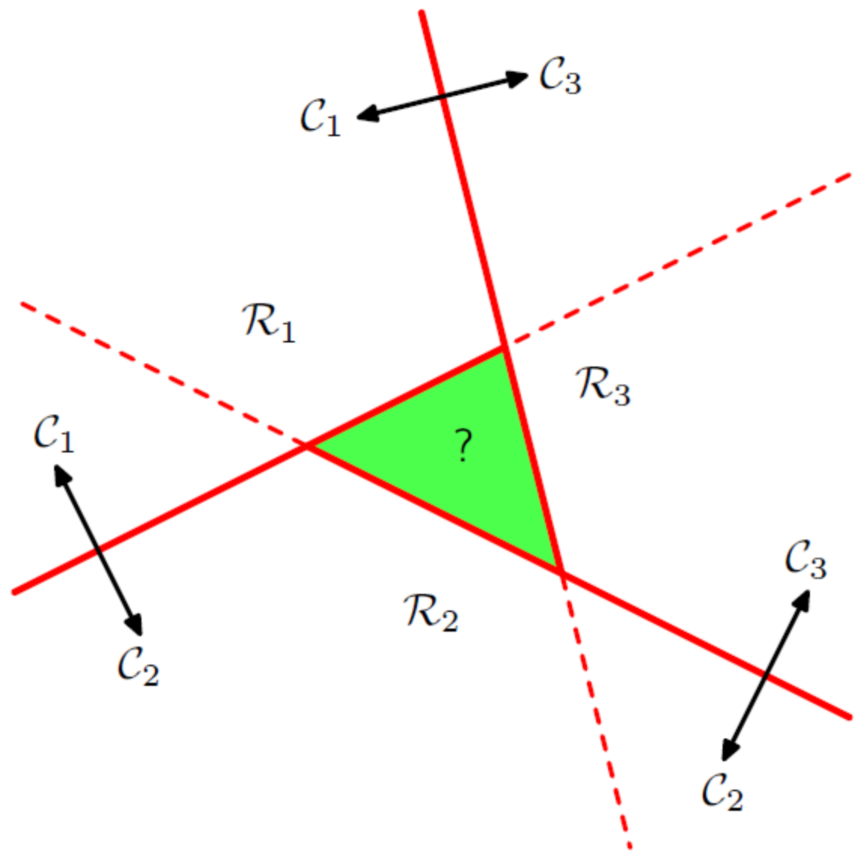


Multiple classes

One-vs-one: Every pair $C_i - C_j$ get a boundary.

Final by majority vote.

High number of classifiers.



Multiple classes

K-linear discriminant functions:

$$y_k(x) = w_k^T \boldsymbol{x} + w_{k0}$$

Assign x to C_k if $y_k(x) \geq y_j(x)$ for all $j \neq k$

Decision boundary:

$$\left(w_k - w_j\right)^T \boldsymbol{x} + \left(w_{k0} - w_{j0}\right) = 0$$

Decision region is singly connected:

$$x = \lambda x_A + (1 - \lambda) x_B$$

If x_A and x_B have same label, so does x.

Multiple Classes