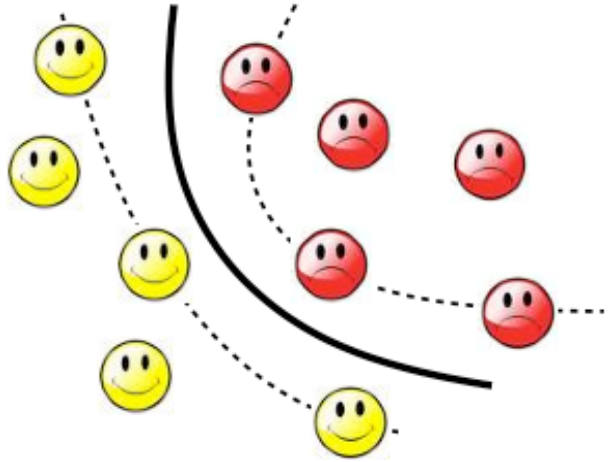
THE UNIVERSITY OF SYDNEY



Machine Learning and Data Mining (COMP 5318)

Review

Fabio Ramos

Conditional Probabilities

- One of the most important concepts in all of Data Mining and Machine Learning
- P(A|B) = P(A,B)/P(B) …assuming P(B) not equal 0.
  – Conditional probability of A given B has occurred.
- Probability it will rain tomorrow given it has rained today.
  – P(A|B) = P(AB)/P(B) = 0.1/0.4 = 1/4 = 0.25
  – In general P(A|B) is not equal to P(B|A)

THE UNIVERSITY OF SYDNEY

Bayes' Rule

Posterior

Prior

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

$$P(hypothesis \mid Data) = \frac{P(data \mid hypothesis)P(hypothesis)}{P(data)}$$

THE UNIVERSITY OF SYDNEY

Bayes' Rule

- P(A|B) = P(AB)/P(B); P(B|A) = P(BA)|P(A)
- Now P(AB) = P(BA)
- Thus P(A|B)P(B) = P(B|A)P(A)
- Thus P(A|B) = [P(B|A)P(A)]/[P(B)]
  - This is called Bayes Rule
  - Basis of almost all prediction
  - Latest theories hypothesise that human memory and action is Bayes' rule in action.

THE UNIVERSITY OF SYDNEY

Example

The ASX market goes up 60% of the days of a year. 40% of the time it stays the same or goes down. The day the ASX is up, there is a 50% chance that the Shanghai Index is up. On other days there is 30% chance that Shanghai goes up. Suppose The Shanghai market is up. What is the probability that ASX was up.

- Define A1 as "ASX is up"; A2 is "ASX is not up"
  Define S1 as "Shanghai is up"; S2 is "Shanghai is not up"
- We want to calculate P(A1|S1) ?
- P(A1) = 0.6; P(A2) = 0.4;
  P(S1|A1) = 0.5; P(S1|A2) = 0.3
- P(S2|A1) = 1 – P(S1|A1) = 0.5;
  P(S2|A2) = 1 – P(S1|A2) = 0.7;

THE UNIVERSITY OF SYDNEY

Example cont.

- We want to calculate P(A1|S1) ?
- P(A1) = 0.6; P(A2) = 0.4;
  P(S1|A1) = 0.5; P(S1|A2) = 0.3
  P(S2|A1) = 1 – P(S1|A1) = 0.5;
  P(S2|A2) = 1 – P(S1|A2) = 0.7;
- P(A1|S1) = P(S1|A1)P(A1)/(P(S1))
- How do we calculate P(S1) ?

THE UNIVERSITY OF SYDNEY

Example cont.

- P(S1) = P(S1,A1) + P(S1,A2) [Key Step]
  = P(S1|A1)P(A1) + P(S1|A2)P(A2)
  = 0.5 x 0.6 + 0.3 x 0.4
  = 0.42
- Finally,
  P(A1|S1) = P(S1|A1)P(A1)/P(S1)
  = (0.5 x 0.6)/0.42
  = 0.71

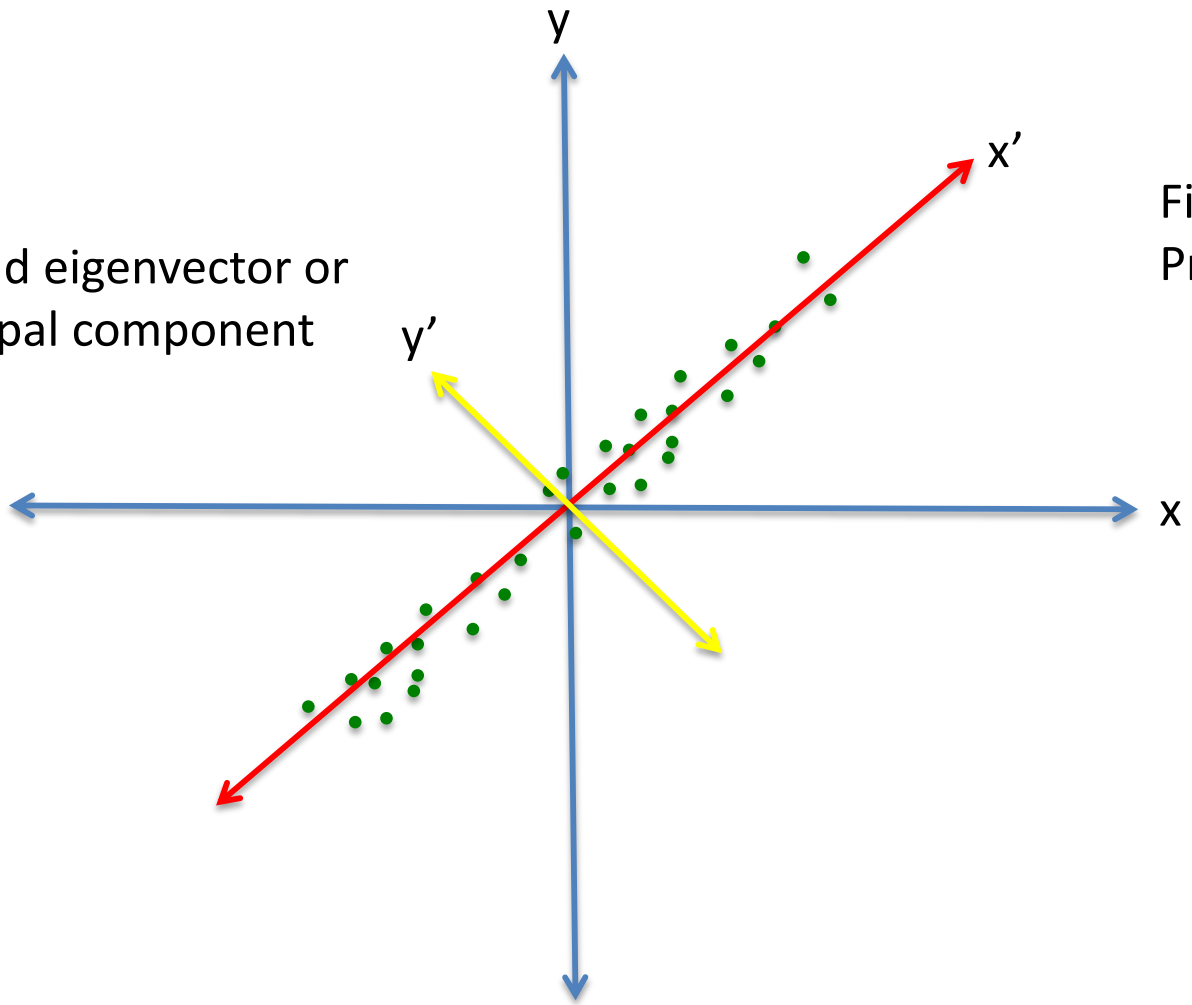
THE UNIVERSITY OF SYDNEY

PCA

The central idea of principal component analysis (PCA) is to reduce the dimensionality of a data set consisting of a large number of interrelated variables, while retaining as much as possible of the variation present in the data set. This is achieved by transforming to a new set of variables, the principal components (PCs), which are uncorrelated, and which are ordered so that the first few *retain most of the variation present in* all *of the original variables.*
[Jolliffe, Pricipal Component Analysis, 2nd edition]

THE UNIVERSITY OF SYDNEY

Geometric intuition



Second eigenvector or
Principal component

First eigenvector or
Principal component

THE UNIVERSITY OF SYDNEY

PCA Theorem

Theorem:

Each x_j can be written as: $$\mathbf{x}_j = \bar{\mathbf{x}} + \sum_{i=1}^{i=n} g_{ji}\mathbf{e}_i$$

where e_i are the n eigenvectors of Q with non-zero eigenvalues.

Notes:

1. The eigenvectors e_1 e_2 … e_n span an ***eigenspace***
2. e_1 e_2 … e_n are N x 1 orthonormal vectors (directions in N-Dimensional space)
3. The scalars g_{ji} are the coordinates of x_j in the space.

$$g_{ji} = (\mathbf{x}_j - \bar{\mathbf{x}}).\mathbf{e}_i$$

PCA to compress data

- Expressing x in terms of e_1 … e_n has not changed the size of the data
- However, if the points are highly correlated many of the coordinates of x will be zero or closed to zero.

note: this means they lie in a lower-dimensional linear subspace

PCA to compress data

- Sort the eigenvectors e_i according to their eigenvalue:

$$\lambda_1 \geq \lambda_2 \geq \ldots \lambda_n$$

- Assuming that $\lambda_i \approx 0$ if $i > k$

- Then

$$\mathbf{x}_j \approx \bar{\mathbf{x}} + \sum_{i=1}^{i=k} g_{ji}\mathbf{e}_i$$

THE UNIVERSITY OF SYDNEY

Loss functions

- Squared error, 0-1 loss

$$
\begin{aligned}
L(y, \hat{y}) &= (y - \hat{y})^2 \\
L(y, \hat{y}) &= I(y \neq \hat{y})
\end{aligned}
$$

- Minimize risk (expected loss, empirical loss)

$$
\begin{aligned}
R(\hat{f}) &= E_{\mathbf{x},y} L(f(\mathbf{x}), \hat{f}(\mathbf{x})) \\
\hat{R}(\hat{f}) &= \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}(\mathbf{x}_i))
\end{aligned}
$$

Generative vs Discriminative

Generative approach:

Model $p(t, \mathbf{x}) = p(\mathbf{x}|t)p(t)$

Use Bayes' theorem $p(t|\mathbf{x}) = \dfrac{p(\mathbf{x}|t)p(t)}{p(\mathbf{x})}$

Discriminative approach:

Model $p(t|\mathbf{x})$ directly

Naïve Bayes Classifier

Generative model:

$$p(\mathbf{x}, \mathcal{C}_k) = p(\mathcal{C}_k)p(\mathbf{x}_n|\mathcal{C}_k) = \pi\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \Sigma_k)$$

class posterior

class conditional density

class prior

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_{\mathcal{C}_j} p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)}$$

normalising constant

Example of Naïve Bayes Classifier

THE UNIVERSITY OF SYDNEY

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

A: attributes

M: mammals

N: non-mammals

$$P(A \mid M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A \mid N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A \mid M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(A \mid N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

P(A|M)P(M) > P(A|N)P(N)

=> Mammals

Logistic Regression

- Assumes a parametric form for directly estimating P($Y \mid X$). For binary concepts, this is:

$$P(Y=1 \mid X) = \frac{1}{1+\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

$$P(Y=0 \mid X) = 1 - P(Y=1 \mid X)$$

$$= \frac{\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}{1+\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

- Equivalent to a one-layer backpropagation neural net.
- Logistic regression is the source of the sigmoid function used in backpropagation.
- Objective function for training is somewhat different.

Logistic Regression Training

- Weights are set during training to maximise the **conditional data likelihood** :

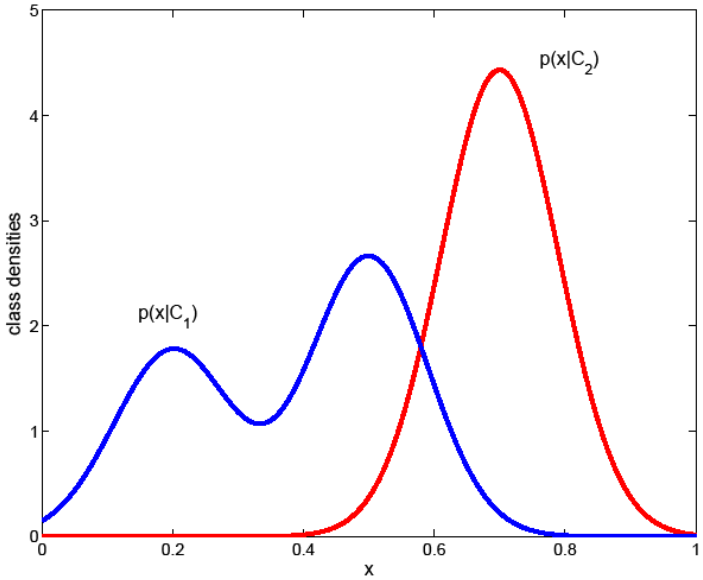
$$W \leftarrow \underset{W}{\operatorname{argmax}} \prod_{d \in D} P(Y^d \mid X^d, W)$$

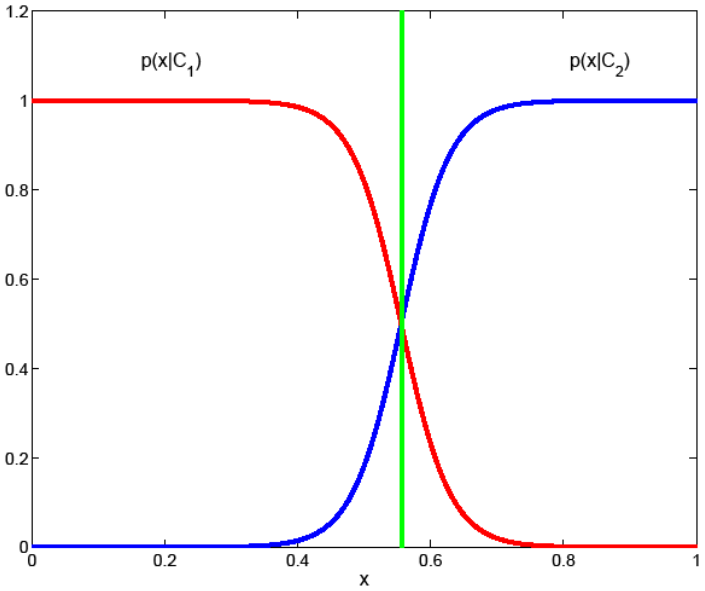
where D is the set of training examples and Y^d and X^d denote, respectively, the values of Y and X for example d.

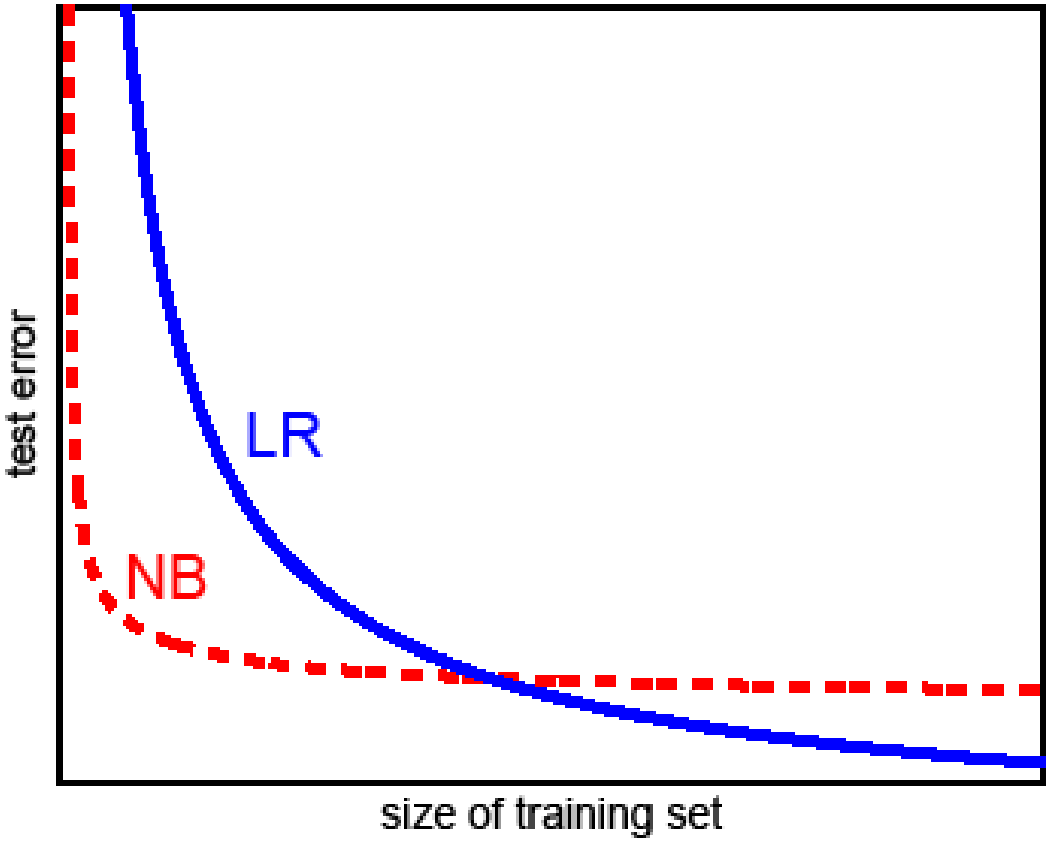
- Equivalently viewed as maximising the **conditional log likelihood** (CLL)

$$W \leftarrow \underset{W}{\operatorname{argmax}} \sum_{d \in D} \ln P(Y^d \mid X^d, W)$$

Logistic Regression vs NB





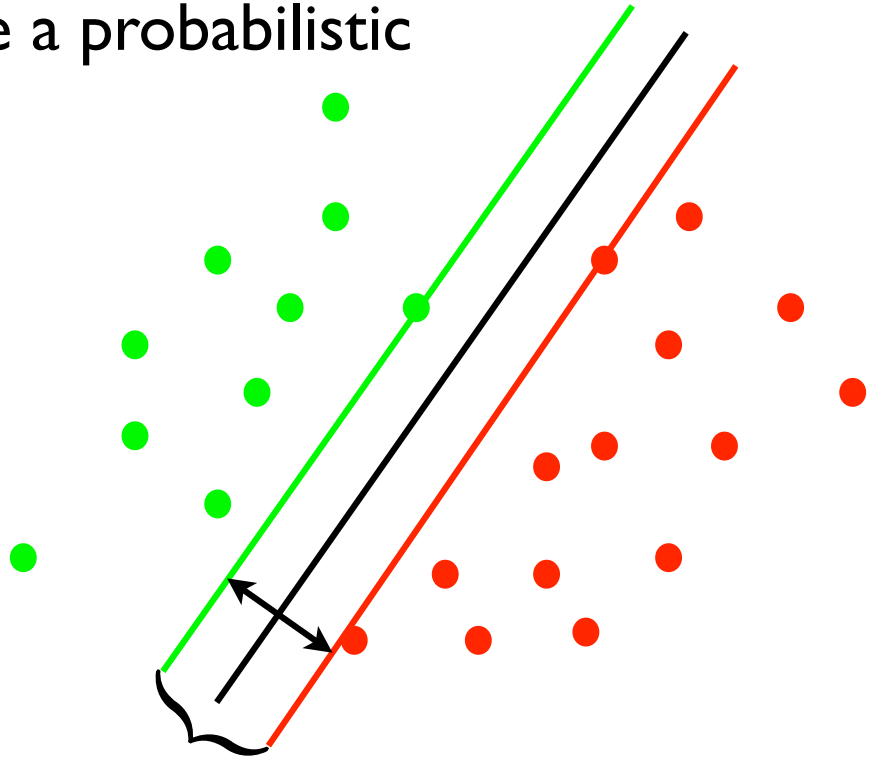


THE UNIVERSITY OF SYDNEY

Maximum margin

Things to note

- The boundary is defined by only a few points (support vectors)
- Difficult to define a probabilistic explanation



Maximum margin classifiers (1)

Given a training set

Inputs $\mathbf{x}_1, \ldots, \mathbf{x}_N$

Targets

$t_1, \ldots, t_N$ where $t_n \in \{-1, 1\}$

Linear classifier

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}) + b$$

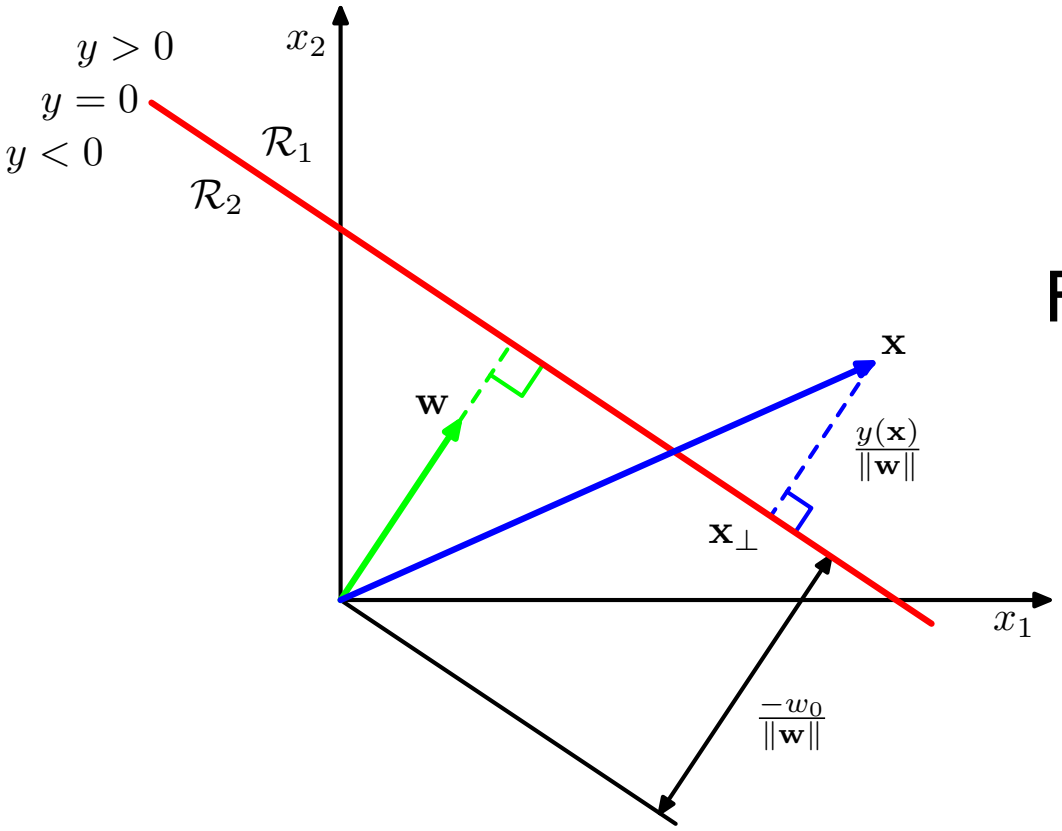
Basis function

Output sign of $y(\mathbf{x})$

so that $t_n y(\mathbf{x}_n) > 0$

for all points in the training set

Maximum margin classifiers(2)



Assuming $\phi(\mathbf{x}) = \mathbf{x}$

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0$$

For points in the boundary

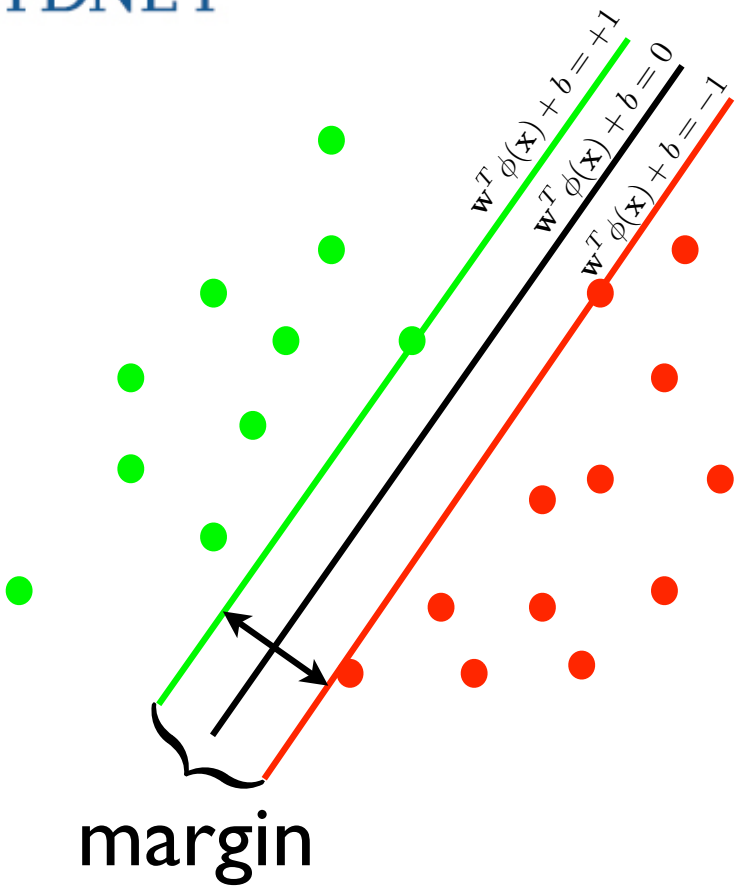
$$y(\mathbf{x}) = 0 \quad \text{and}$$

$$\frac{\mathbf{w}^{\mathrm{T}}\mathbf{x}}{\|\mathbf{w}\|} = -\frac{w_0}{\|\mathbf{w}\|}$$

For arbitrary $\mathbf{x}$

$$\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{\|\mathbf{w}\|} \quad \text{and} \quad r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}$$

Support Vector Machines



Quadratic programming

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$$

s.t.

$$t_n\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)+b\right)\geqslant 1, \quad n=1,\ldots,N$$

- Solve efficiently by quadratic programming (QP)
- Hyperplane defined by support vectors

The Dual SVM formulation

Quadratic programming

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$$

s.t.

$$t_n\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)+b\right) \geqslant 1, \quad n = 1, \ldots, N$$

Lagrangian function

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \left\{t_n(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)+b) - 1\right\}$$

setting the derivatives w.r.t. $\mathbf{w}$ and b equal to zero:

$$\begin{aligned} \mathbf{w} &= \sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}(\mathbf{x}_n) \\ 0 &= \sum_{n=1}^{N} a_n t_n. \end{aligned}$$

$$\widetilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$$

Kernel trick

$$k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}')$$

s.t.

$$\begin{aligned} a_n &\geqslant 0, \qquad n = 1, \ldots, N, \\ \sum_{n=1}^{N} a_n t_n &= 0. \end{aligned}$$

THE UNIVERSITY OF SYDNEY

Kernel SVMs

- Solve the QP problem

$$\widetilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$$

with respect to $\mathbf{a}$ subject to the constraints

$$\begin{aligned} a_n &\geqslant 0, \qquad n = 1, \ldots, N, \\ \sum_{n=1}^{N} a_n t_n &= 0. \end{aligned}$$

- Support vectors satisfy

$$t_n \left(\sum_{m \in \mathcal{S}} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m) + b \right) = 1$$

- For query points compute

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n) + b.$$

THE UNIVERSITY OF SYDNEY

Think about kernels not features

Polynomial kernel

$$
\begin{aligned}
k(\mathbf{x}, \mathbf{z}) &= \left(1 + \mathbf{x}^{\mathrm{T}}\mathbf{z}\right)^2 = (1 + x_1 z_1 + x_2 z_2)^2 \\
&= 1 + 2x_1 z_1 + 2x_2 z_2 + x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2 \\
&= (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, \sqrt{2}x_1 x_2, x_2^2)(1, \sqrt{2}z_1, \sqrt{2}z_2, z_1^2, \sqrt{2}z_1 z_2, z_2^2)^{\mathrm{T}} \\
&= \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{z}).
\end{aligned}
$$

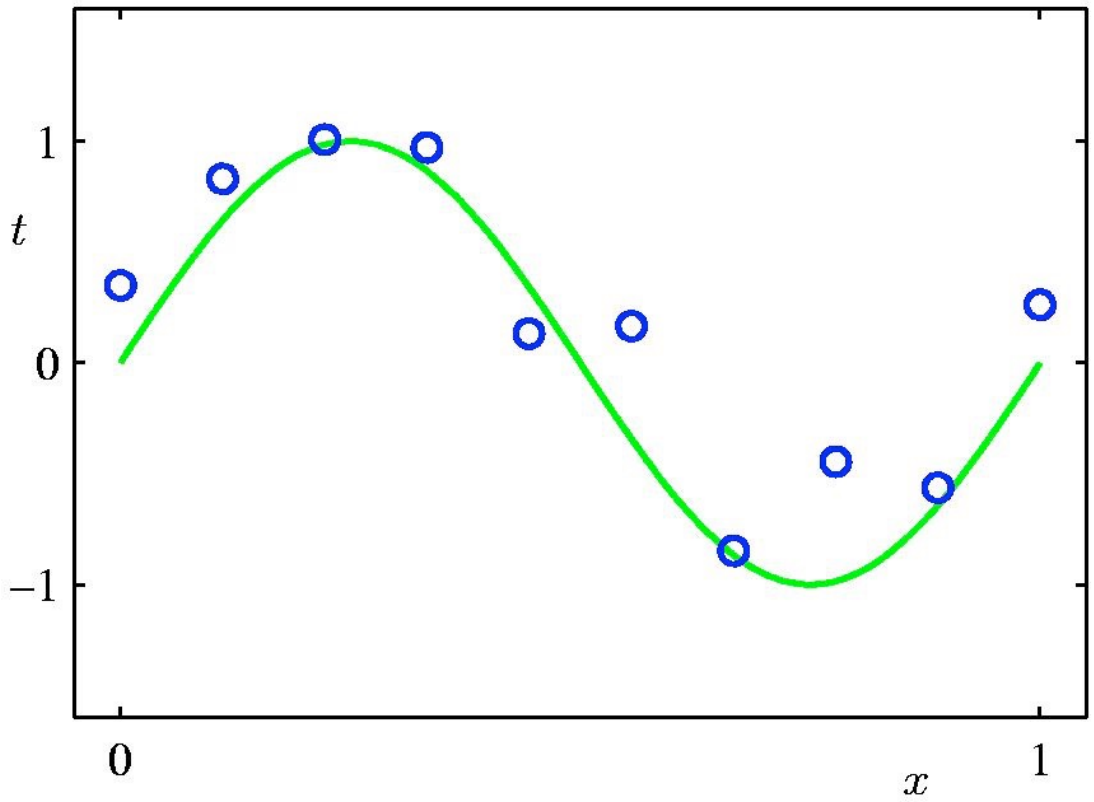
Gaussian kernel

$$k(\mathbf{x}, \mathbf{z}) = \exp(-\gamma\|\mathbf{x} - \mathbf{z}\|^2)$$

Infinite dimensional feature space!

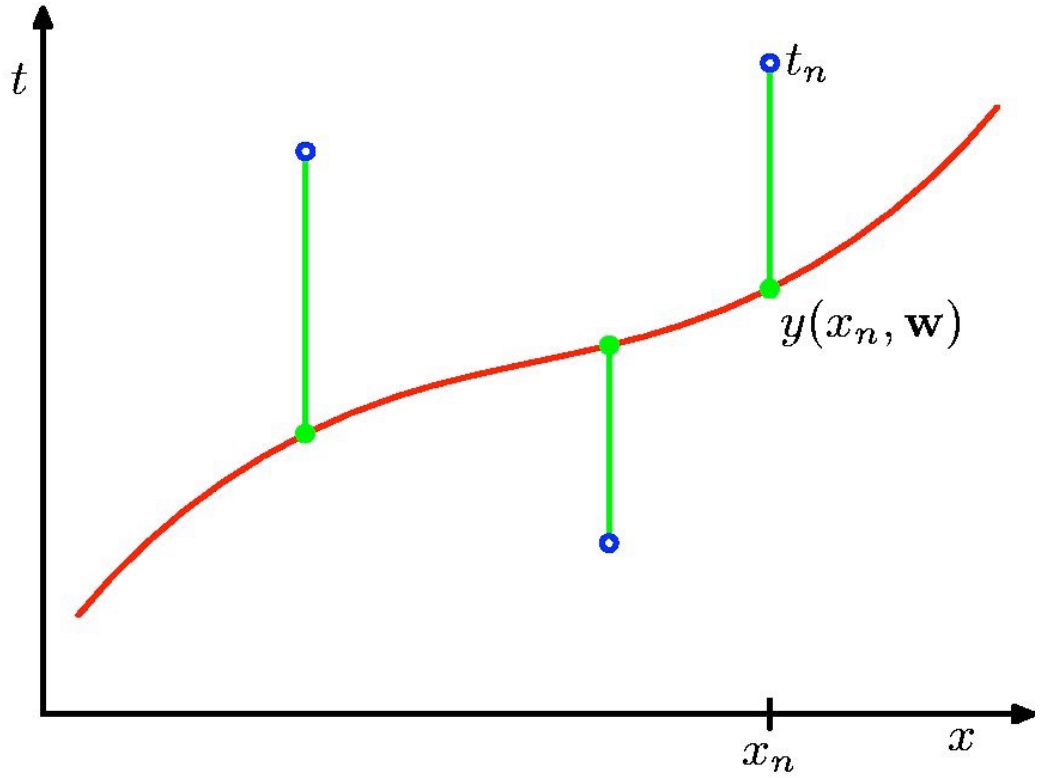
THE UNIVERSITY OF SYDNEY

Polynomial Curve Fitting



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

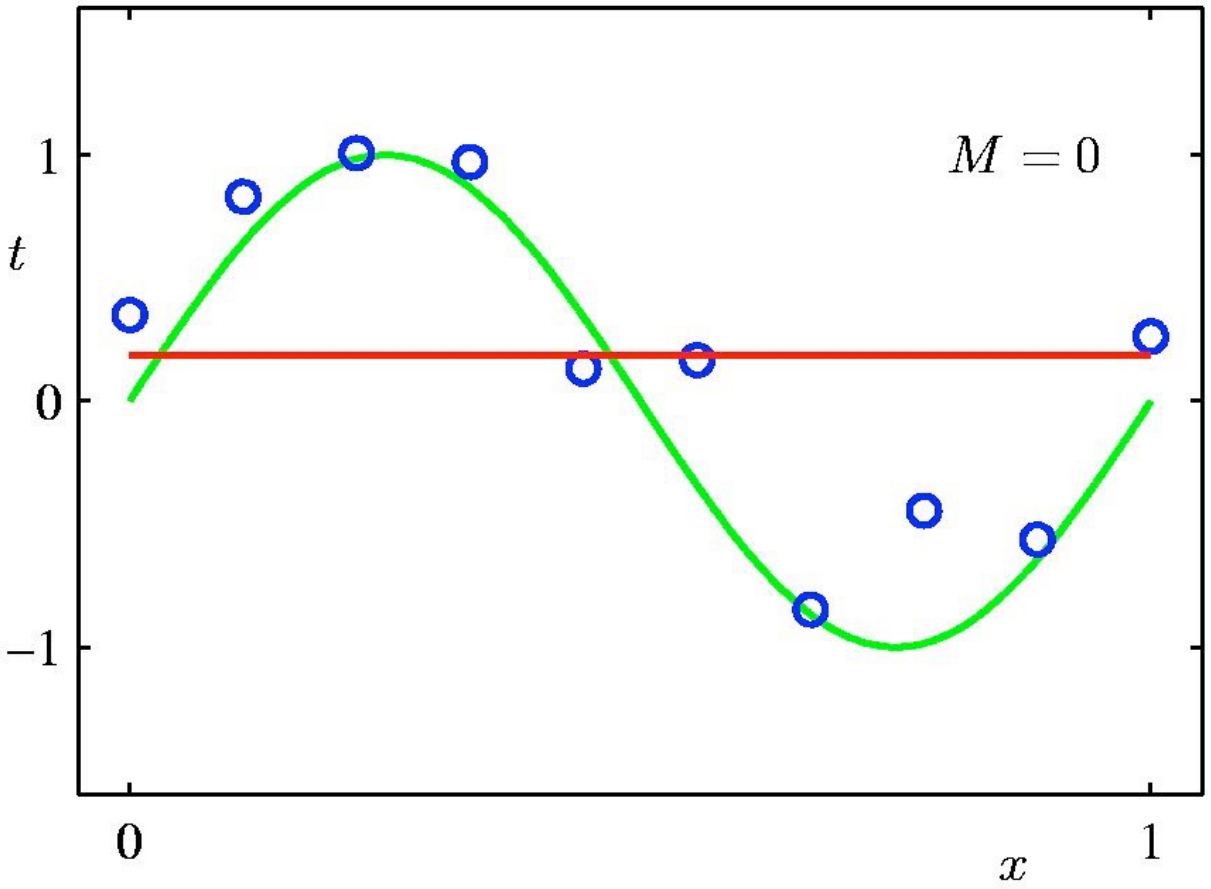
Sum-of-Squares Error Function



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

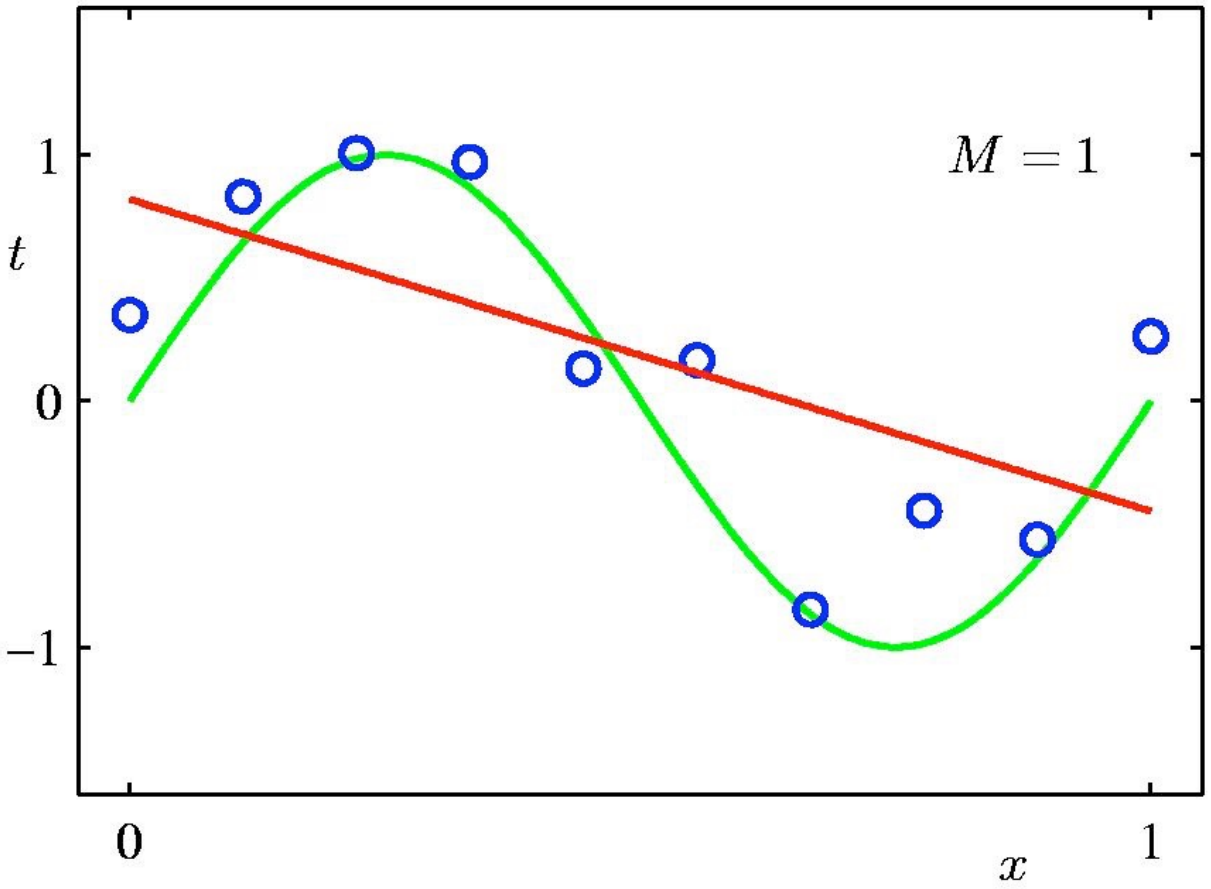
THE UNIVERSITY OF SYDNEY

0th Order Polynomial



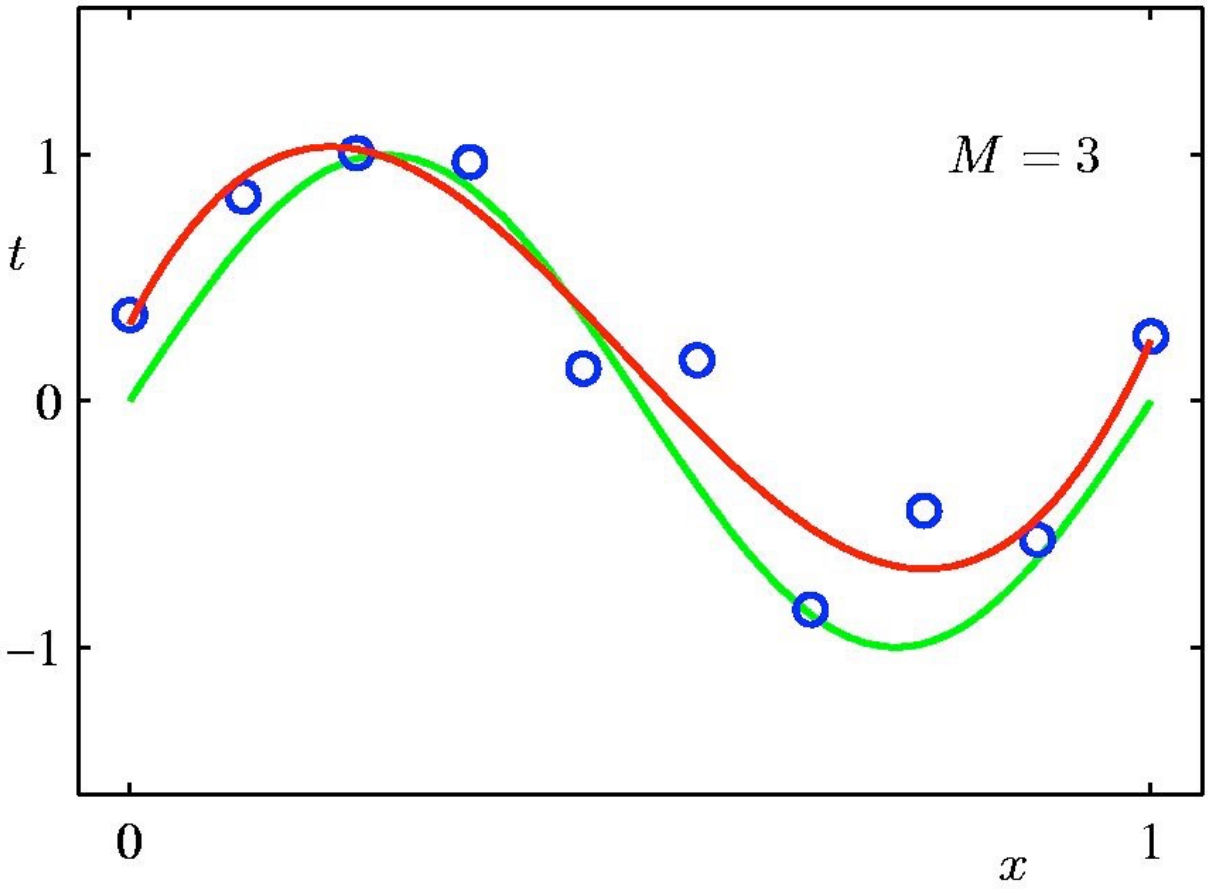
THE UNIVERSITY OF SYDNEY

1st Order Polynomial



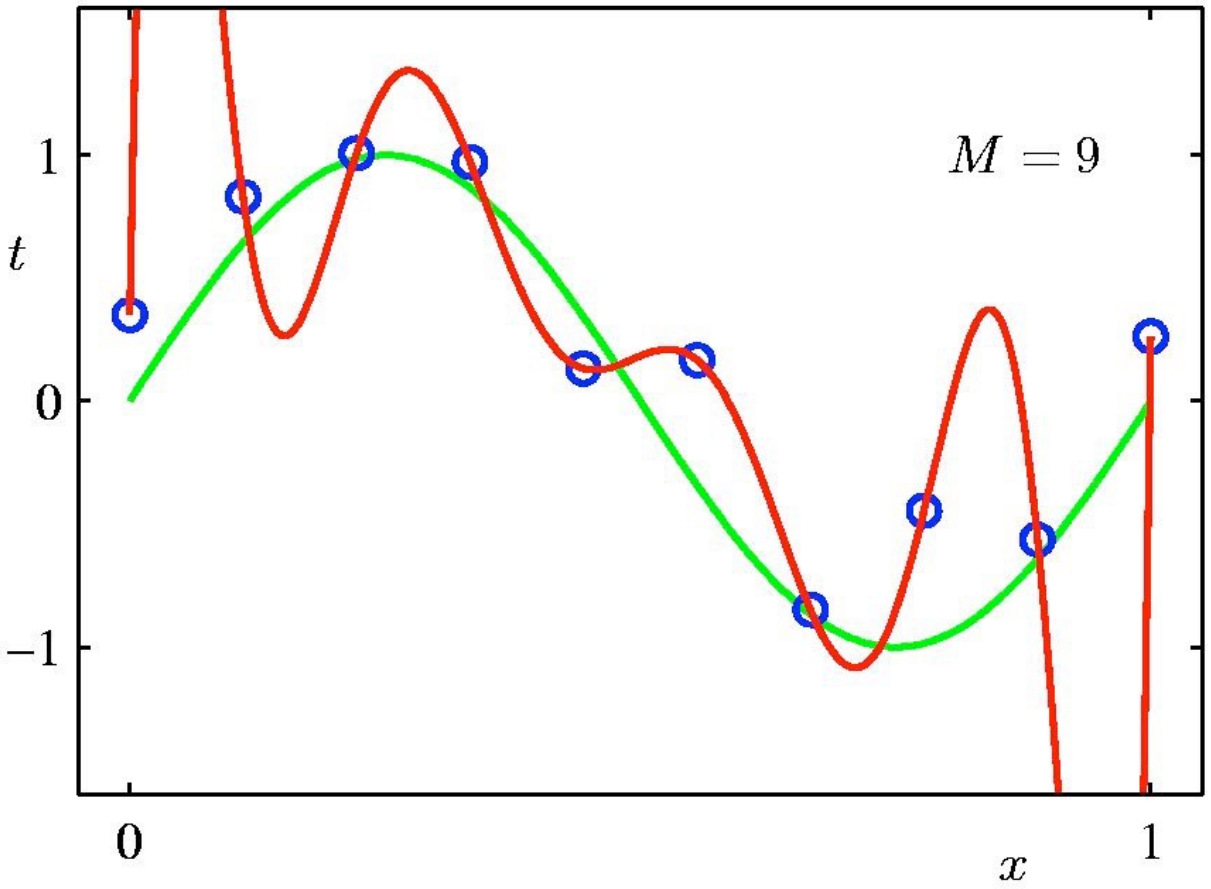
THE UNIVERSITY OF SYDNEY

3rd Order Polynomial

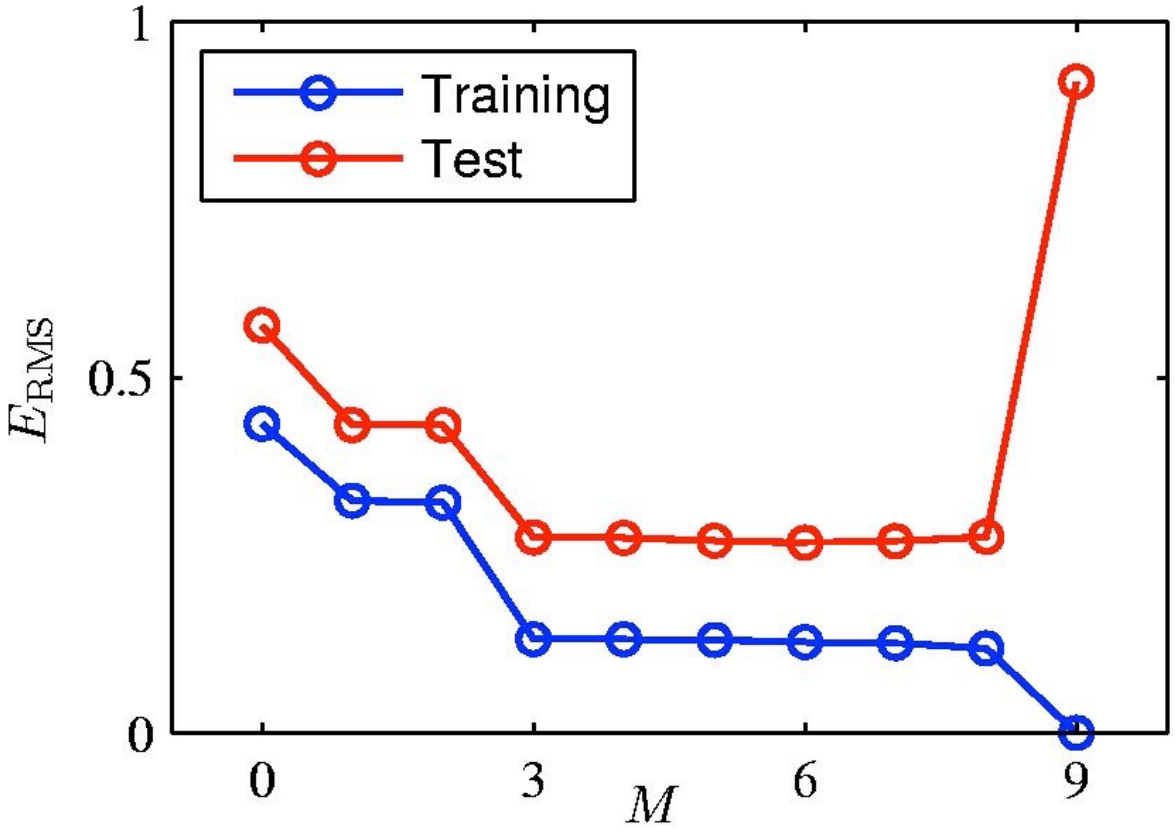


THE UNIVERSITY OF SYDNEY

9th Order Polynomial



Over-fitting



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

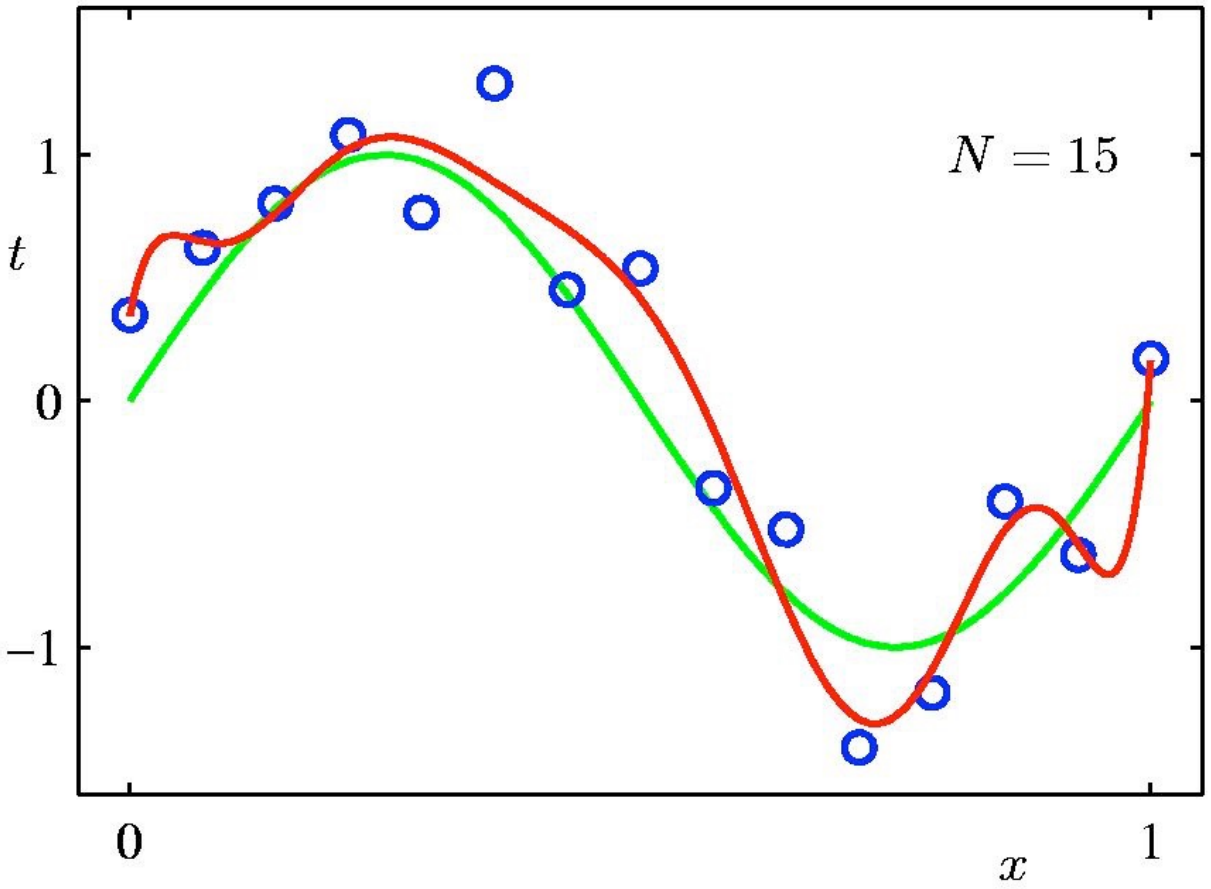
Polynomial Coefficients

	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	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

THE UNIVERSITY OF SYDNEY

Data Set Size: $N = 15$

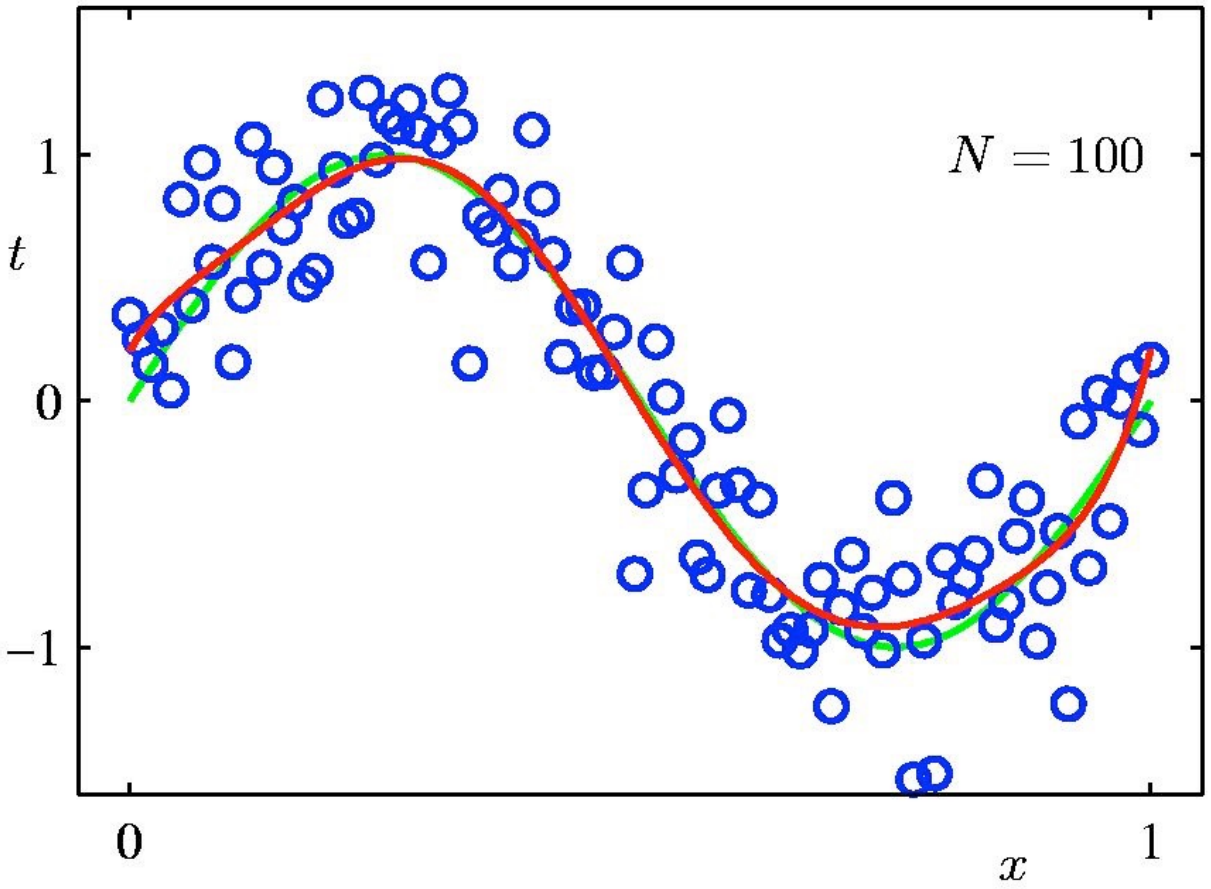
9th Order Polynomial



THE UNIVERSITY OF SYDNEY

Data Set Size: $N = 100$

9th Order Polynomial



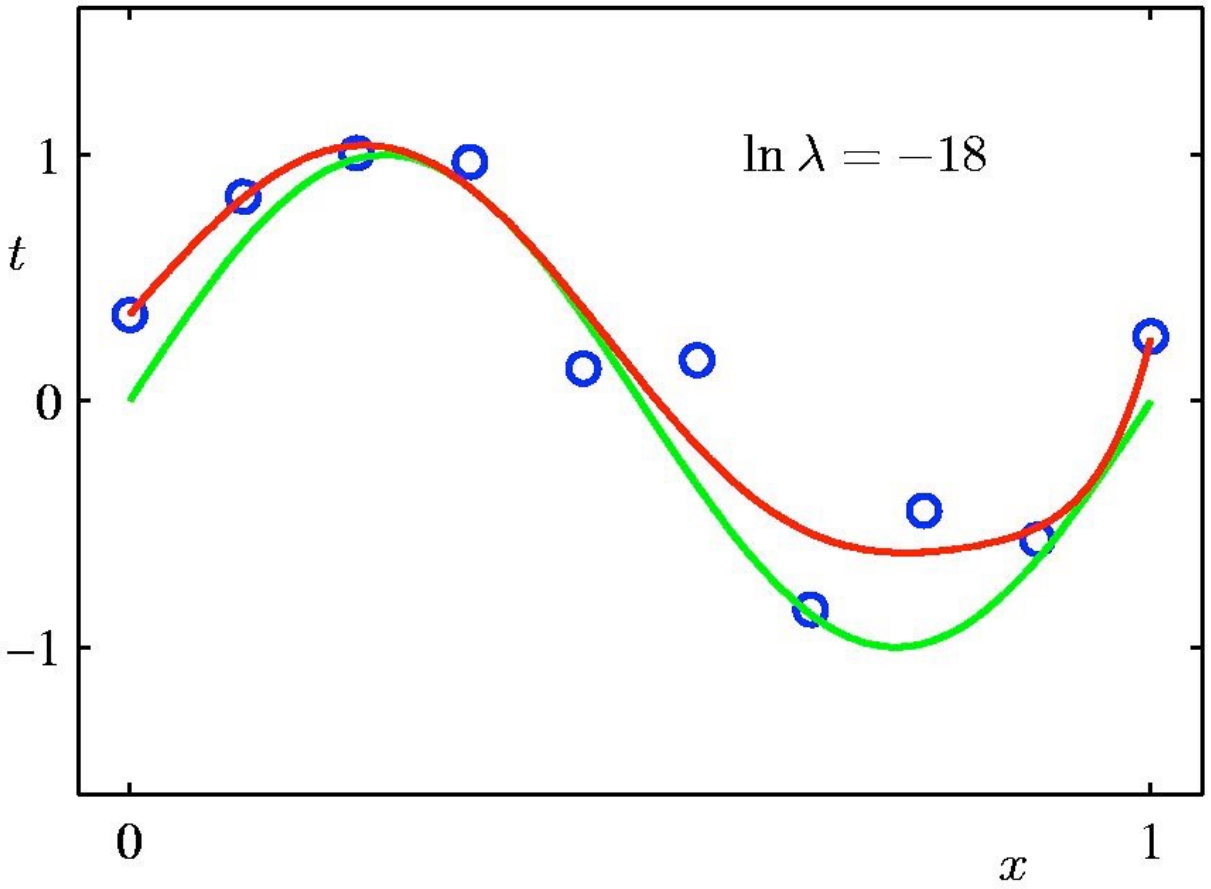
Regularisation

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

Penalise large coefficient values

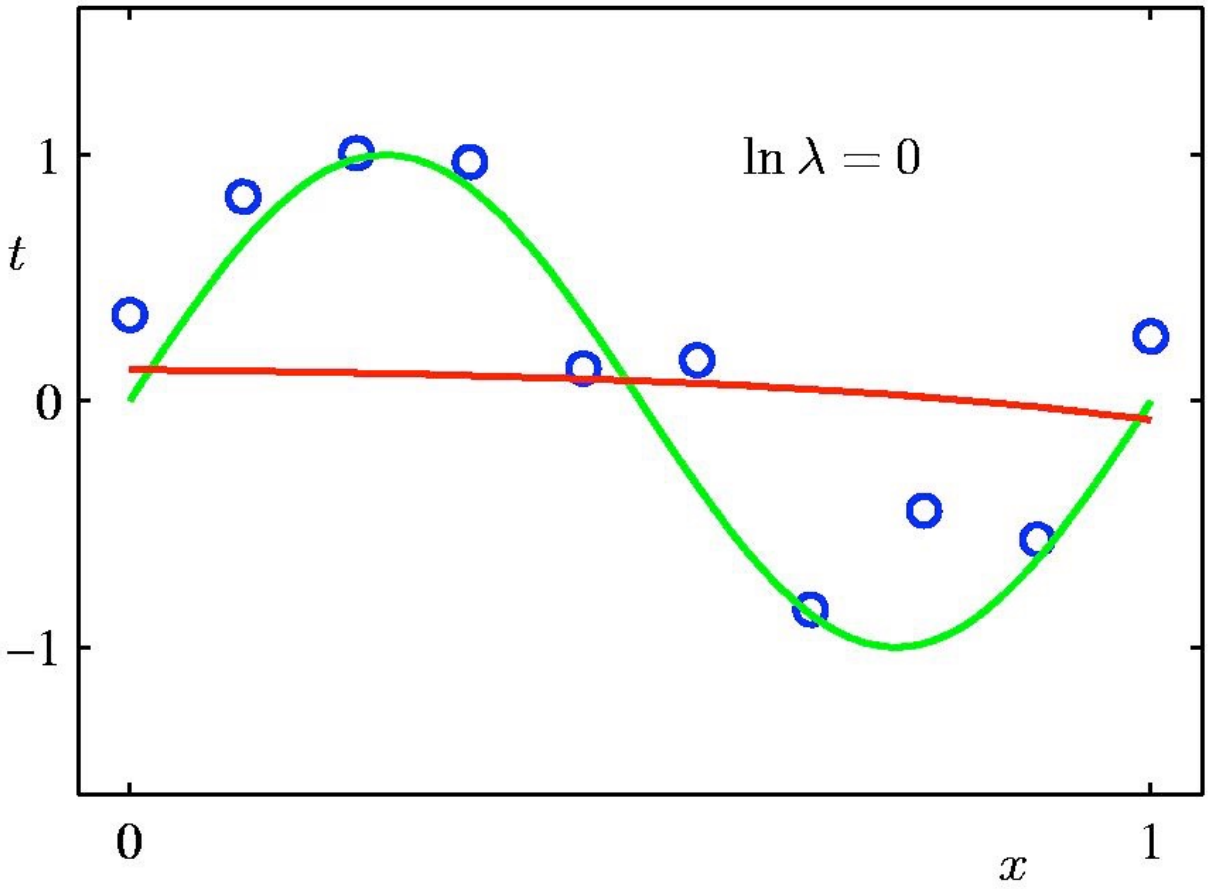
THE UNIVERSITY OF SYDNEY

Regularisation: $\ln \lambda = -18$



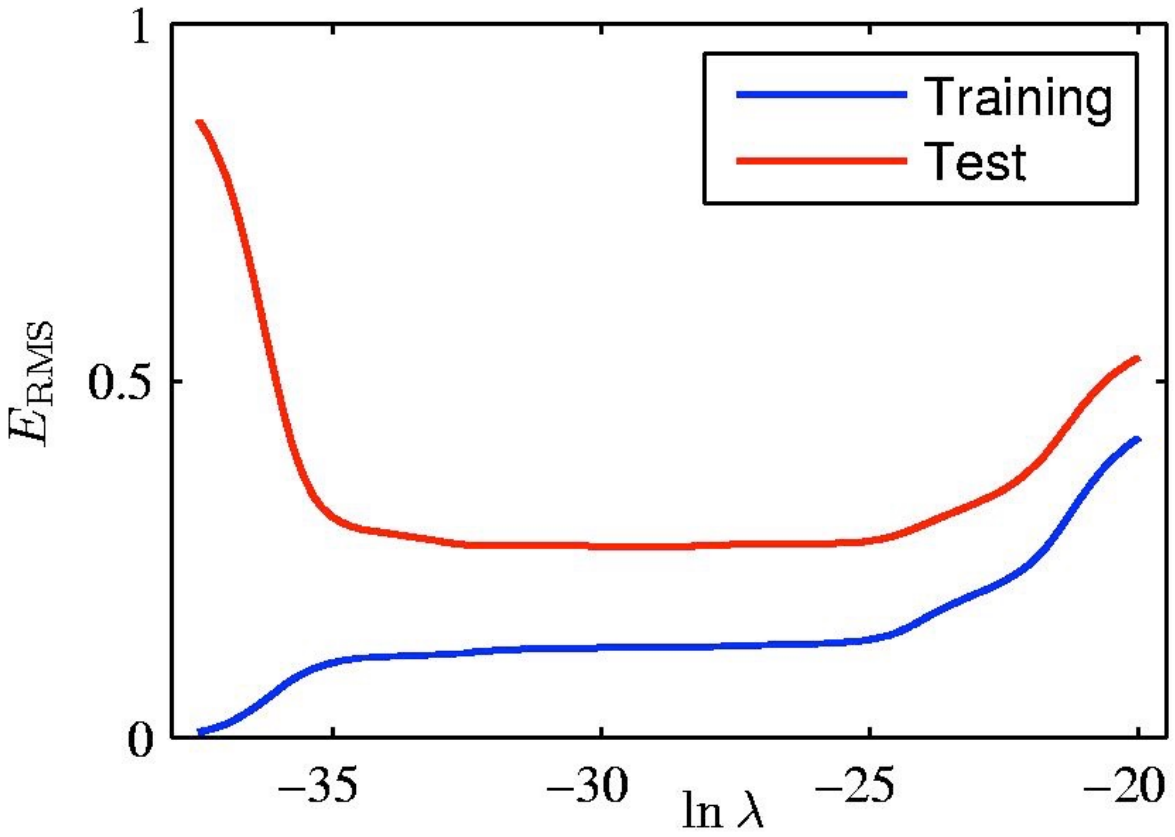
THE UNIVERSITY OF SYDNEY

Regularisation: $\ln \lambda = 0$



THE UNIVERSITY OF SYDNEY

Regularisation: 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

THE UNIVERSITY OF SYDNEY

Maximum Likelihood

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}\left(t_n | y(x_n, \mathbf{w}), \beta^{-1}\right)$$

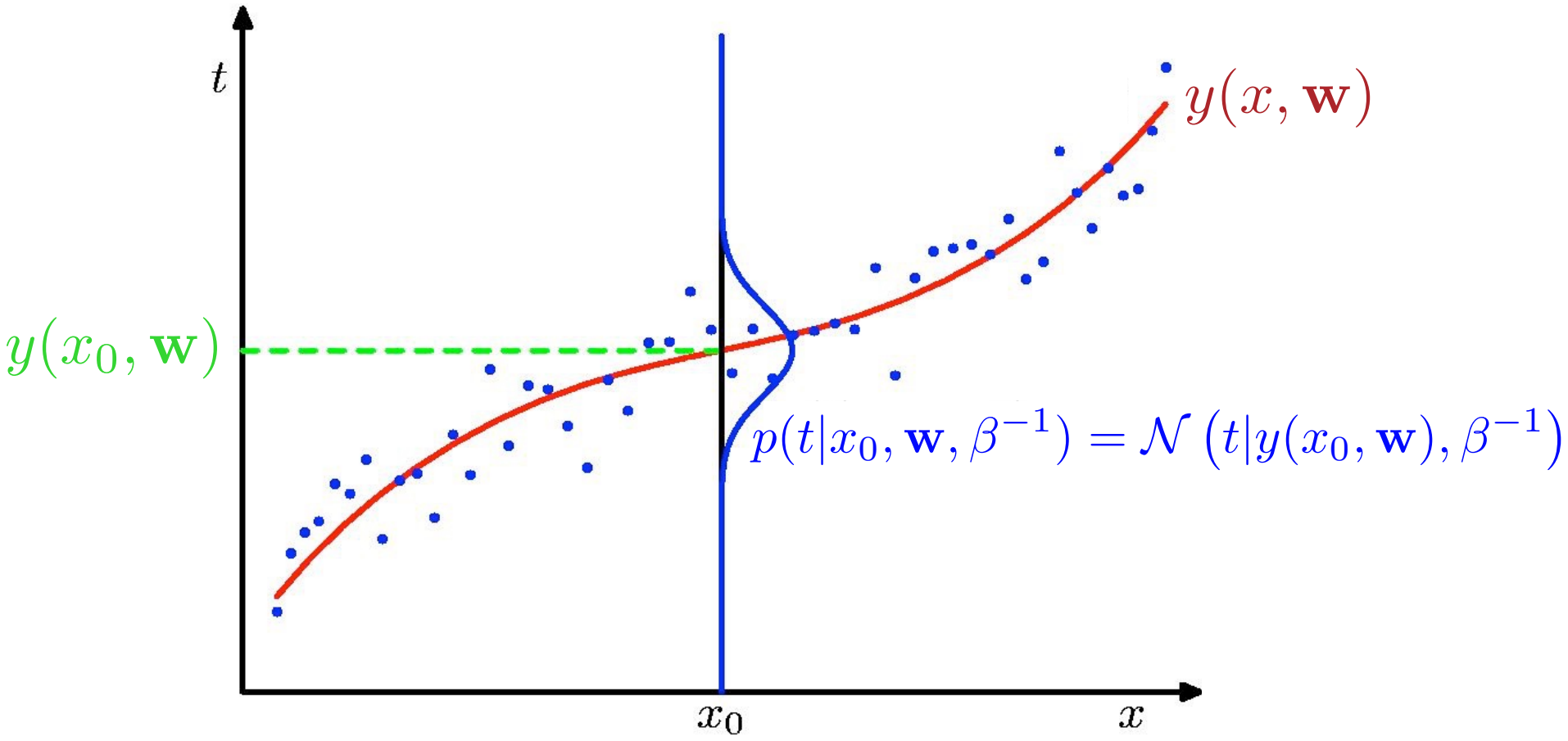
$$\ln p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = \underbrace{-\frac{\beta}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2}_{\beta E(\mathbf{w})} + \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi)$$

Determine $\mathbf{w}_{\mathrm{ML}}$ by minimising sum-of-squares error, $E(\mathbf{w})$.

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

THE UNIVERSITY OF SYDNEY

Curve Fitting Re-visited



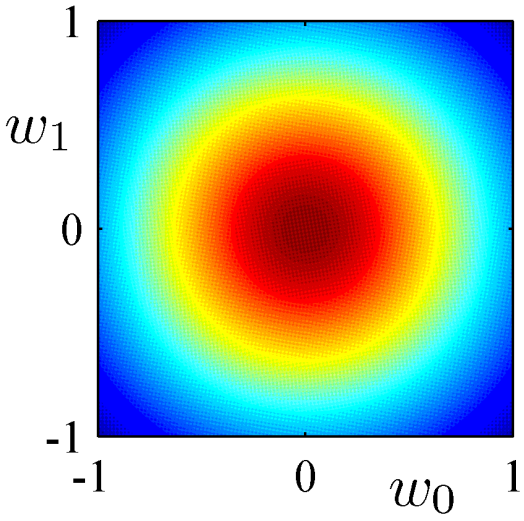
Bayesian Linear Regression

THE UNIVERSITY OF SYDNEY

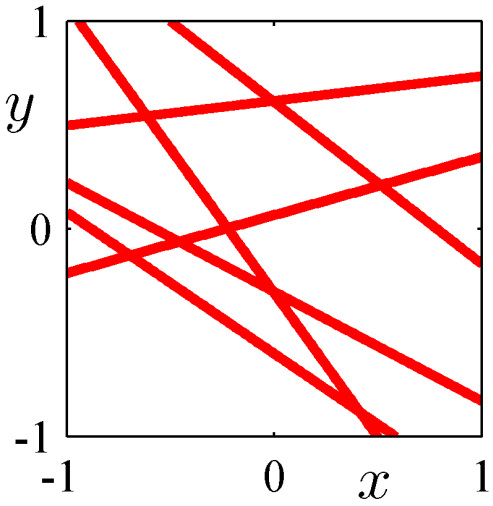
Example:

0 Points Observed

Prior



Data Space



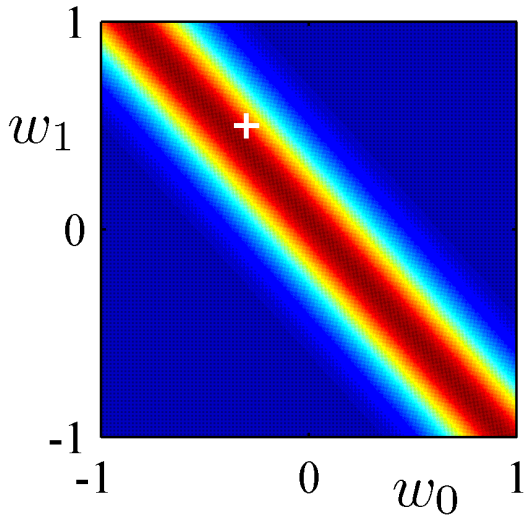
Bayesian Linear Regression

THE UNIVERSITY OF SYDNEY

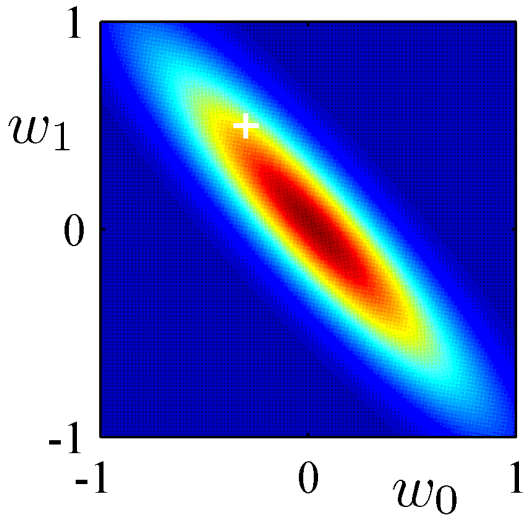
Example:

1 Point Observed

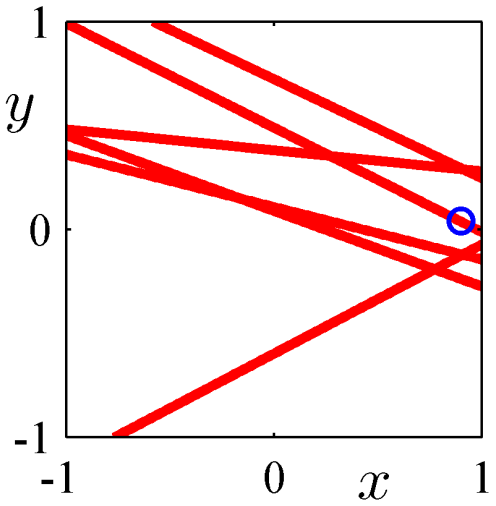
Likelihood



Posterior



Data Space



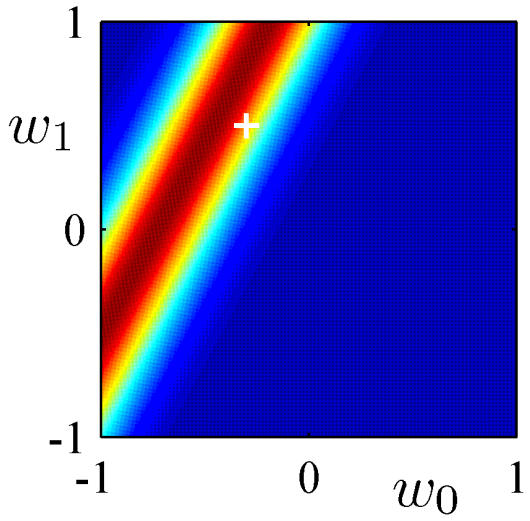
Bayesian Linear Regression

THE UNIVERSITY OF SYDNEY

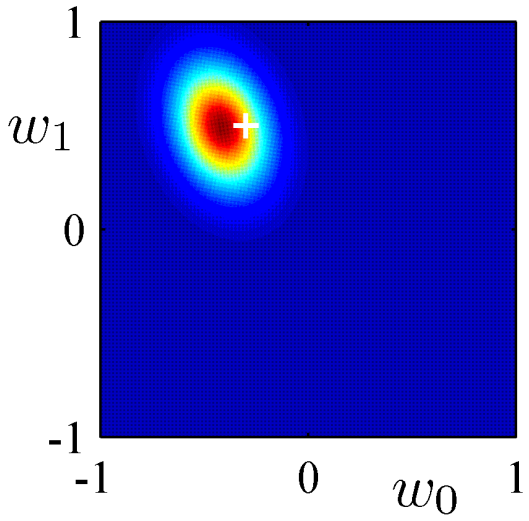
Example:

2 Points Observed

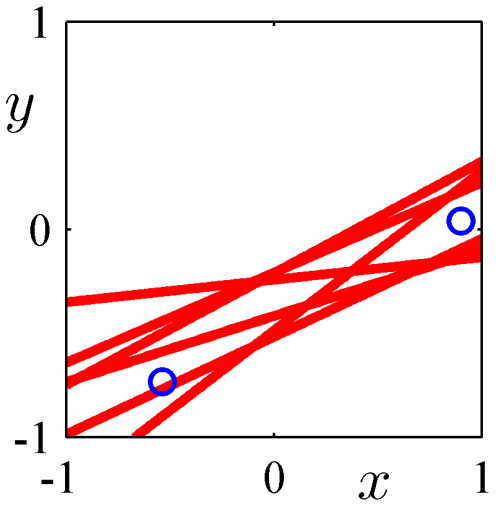
Likelihood



Posterior



Data Space



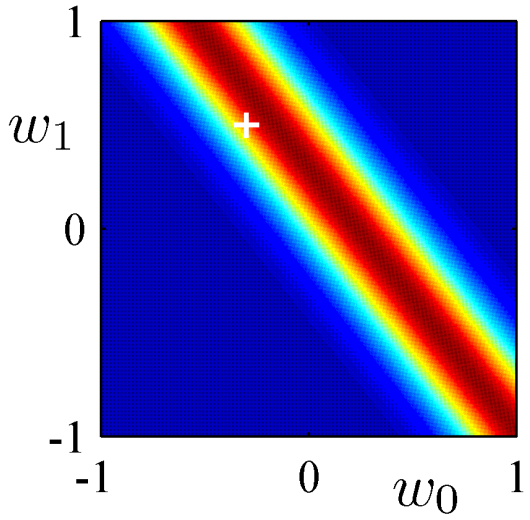
Bayesian Linear Regression

THE UNIVERSITY OF SYDNEY

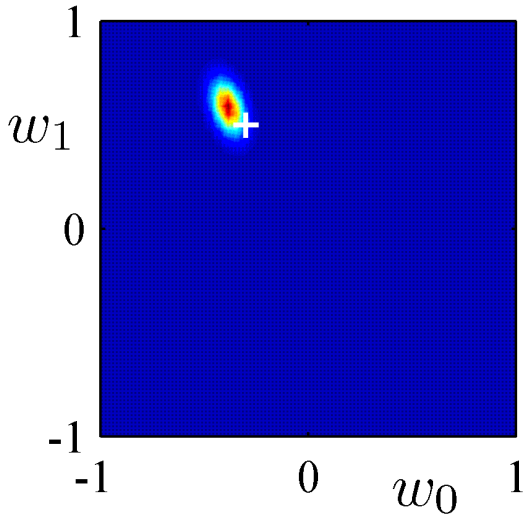
Example:

20 Points Observed

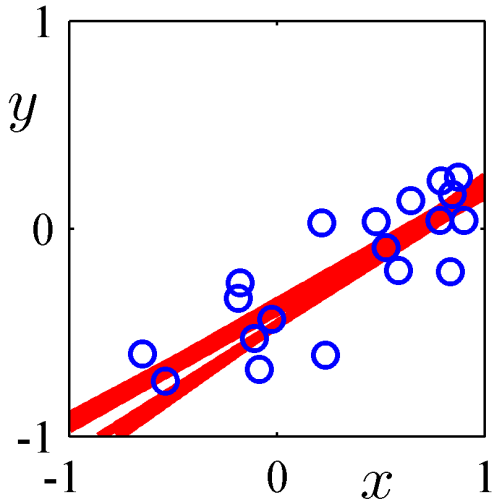
Likelihood



Posterior



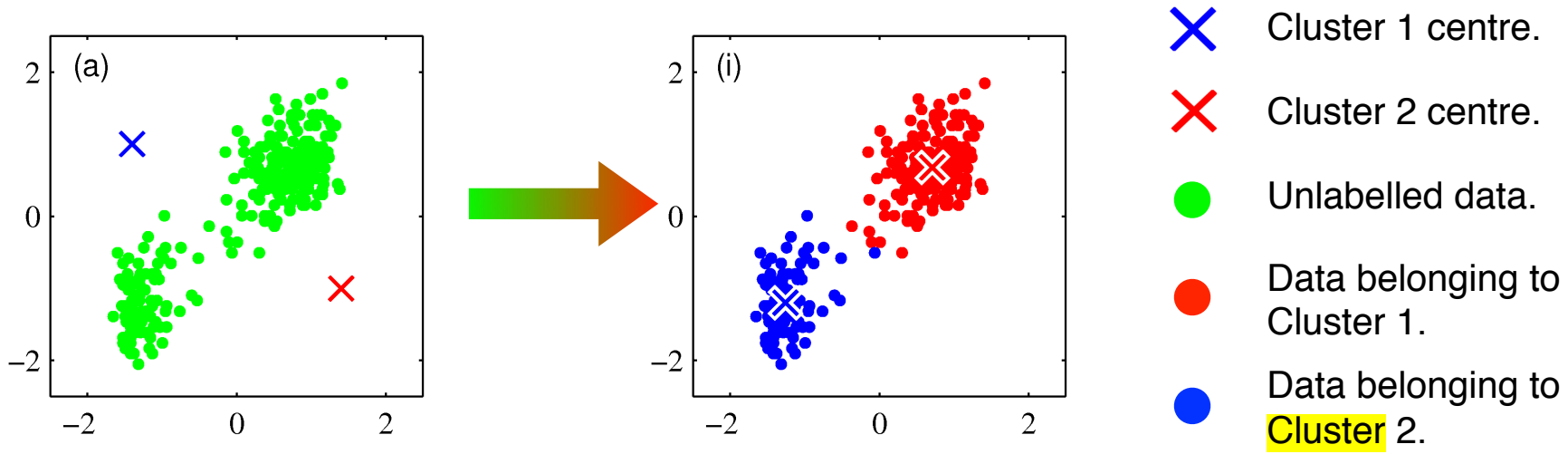
Data Space



Clustering

THE UNIVERSITY OF SYDNEY

Process of grouping similar objects together.



Learn a set of clusters and assign data to a specific cluster.

Deterministic: Hard assignment to each cluster (K-means).

Probabilistic: Model assignment as a discrete latent variable. (Mixtures of Gaussians, Dirichlet Process)

K-Means

Objective function:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

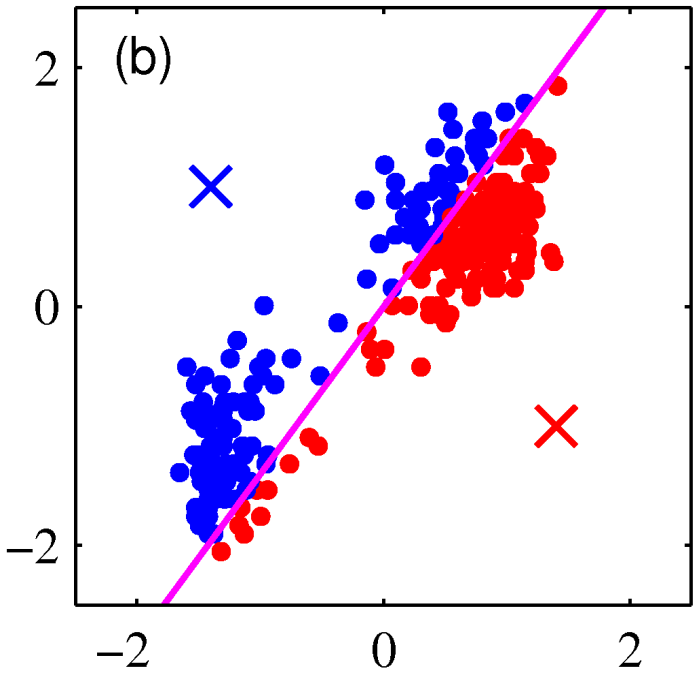
Represents the sum of the squares of the distances of each datapoint to its assigned prototype vector.

Goal: Find $\{r_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$ that minimise J.

$$\{r_{nk}, \boldsymbol{\mu}_k\}^\star = \underset{\{r_{nk}, \boldsymbol{\mu}_k\}}{\operatorname{argmin}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

K-Means Example

Number of clusters: $K = 2$

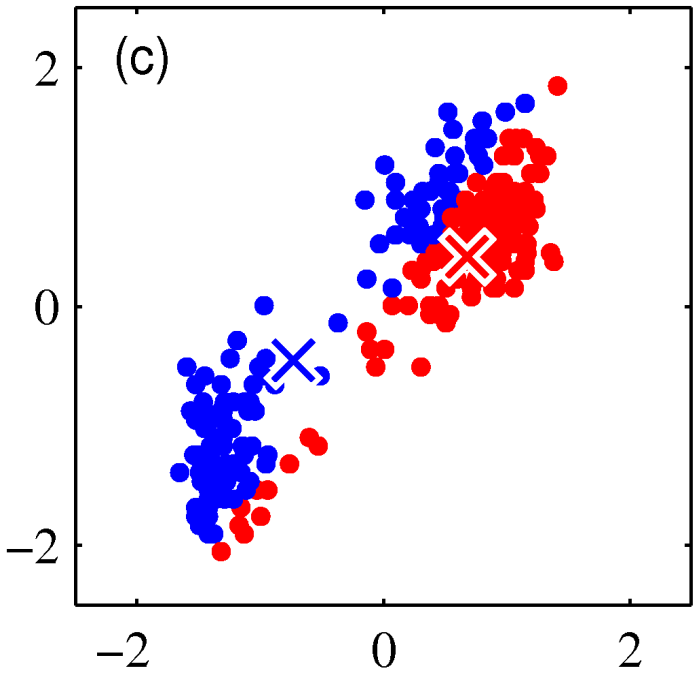


1 Data Preprocessing
2 Initialise $\{\boldsymbol{\mu}_k\}$
3 Repeat until convergence or Max Iterations
4 Minimise J w.r.t. $\{r_{nk}\}$ keeping $\{\boldsymbol{\mu}_k\}$ fixed.
5 Minimise J w.r.t. $\{\boldsymbol{\mu}_k\}$ keeping $\{r_{nk}\}$ fixed.

Each data point is assigned to the closest cluster centre.

K-Means Example

Number of clusters: $K = 2$

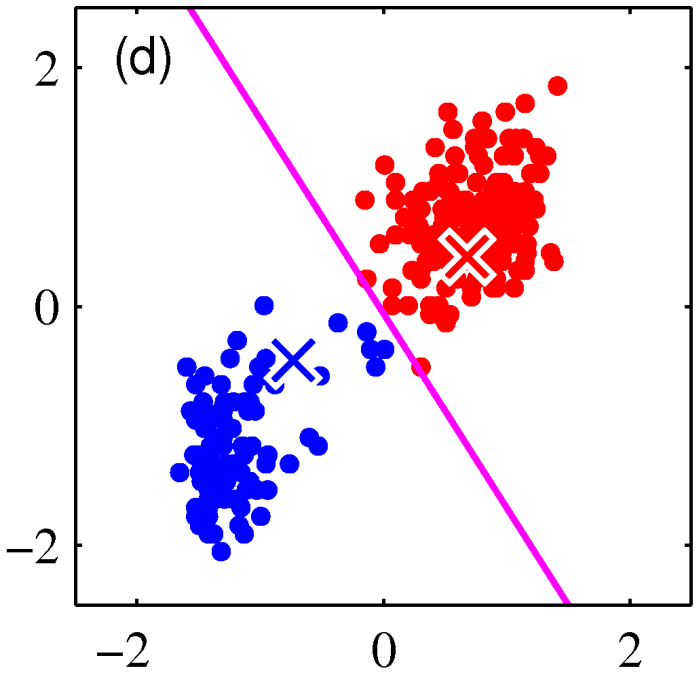


1 Data Preprocessing
2 Initialise $\{\boldsymbol{\mu}_k\}$
3 Repeat until convergence or Max Iterations
4 Minimise J w.r.t. $\{r_{nk}\}$ keeping $\{\boldsymbol{\mu}_k\}$ fixed.
5 Minimise J w.r.t. $\{\boldsymbol{\mu}_k\}$ keeping $\{r_{nk}\}$ fixed.

Re-compute each cluster centre to be the mean of the points previously assigned.

K-Means Example

Number of clusters: $K = 2$



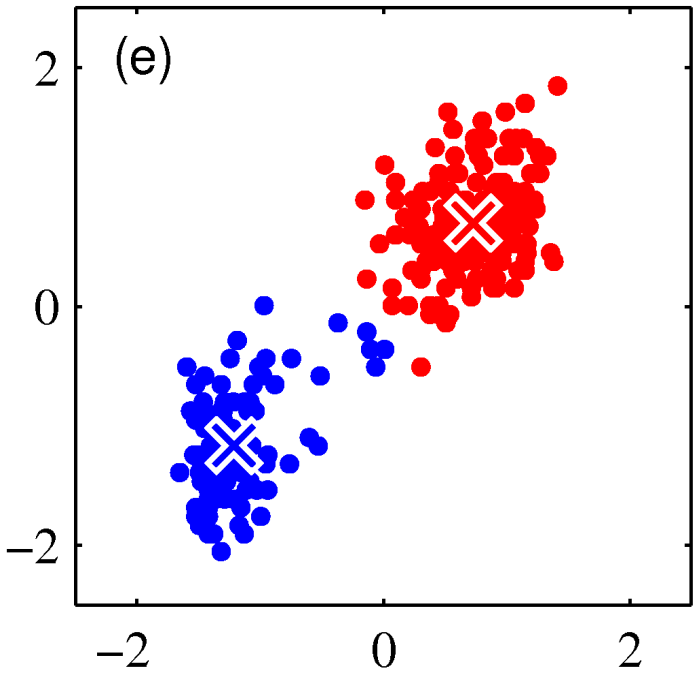
1 Data Preprocessing
2 Initialise $\{\boldsymbol{\mu}_k\}$
3 Repeat until convergence or Max Iterations
4 Minimise J w.r.t. $\{r_{nk}\}$ keeping $\{\boldsymbol{\mu}_k\}$ fixed.
5 Minimise J w.r.t. $\{\boldsymbol{\mu}_k\}$ keeping $\{r_{nk}\}$ fixed.

Each data point is assigned to the closest cluster centre.

K-Means Example

THE UNIVERSITY OF SYDNEY

Number of clusters: $K = 2$



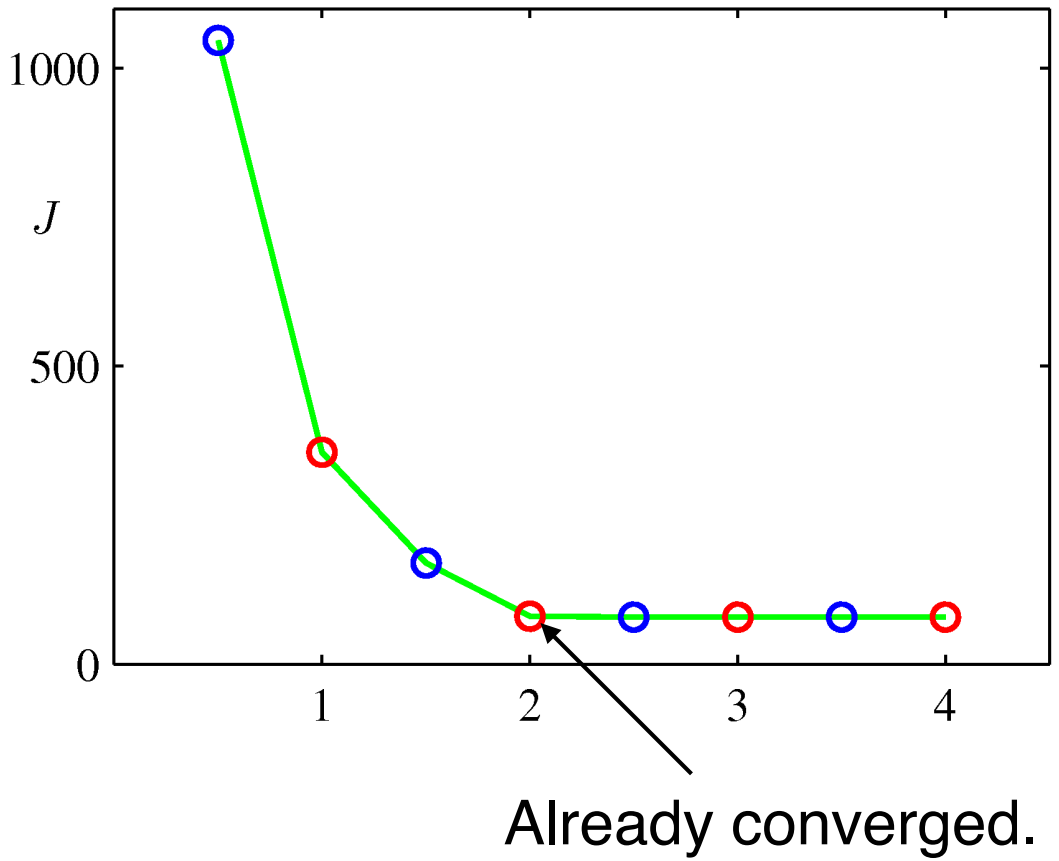
1 Data Preprocessing
2 Initialise $\{\boldsymbol{\mu}_k\}$
3 Repeat until convergence or Max Iterations
4 Minimise J w.r.t. $\{r_{nk}\}$ keeping $\{\boldsymbol{\mu}_k\}$ fixed.
5 Minimise J w.r.t. $\{\boldsymbol{\mu}_k\}$ keeping $\{r_{nk}\}$ fixed.

Re-compute each cluster centre to be the mean of the points previously assigned.

K-Means Example

THE UNIVERSITY OF SYDNEY

Plot of the cost function for each iteration.



EM for Gaussian Mixtures

THE UNIVERSITY OF SYDNEY

Likelihood function:

$$\ln\, p(\mathbf{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k) \right\}$$

Conditions to be satisfied at maximum likelihood:

$$0 = \frac{\partial \ln\, p(\mathbf{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})}{\partial \boldsymbol{\mu}_k}$$

$$0 = -\sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)}}_{\gamma(z_{nk})} \boldsymbol{\Sigma}_k(\mathbf{x}_n - \boldsymbol{\mu}_k)$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N} \gamma(z_{nk})\mathbf{x}_n \qquad N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

N_k is the effective number of points assigned to cluster k.

THE UNIVERSITY OF SYDNEY

Exam: what to expect?

- Numerical questions, e.g., what is probability of X … given … Y
- Theoretical on the main concepts, e.g., why should I regularise logistic regression
- Maths will be simple; no need to memorise equations
- Pros vs Cons for different methods
- In Dim Red, what does each method preserve in low D?
- Check the exercises in Bishop, Ullman, and Murphy's books
- Check the questions here: https://www.analyticsvidhya.com/blog/2016/09/40-interview-questions-asked-at-startups-in-machine-learning-data-science/
- For deep learning, see these: https://www.analyticsvidhya.com/blog/2017/01/must-know-questions-deep-learning/