Computer vision: models, learning and inference

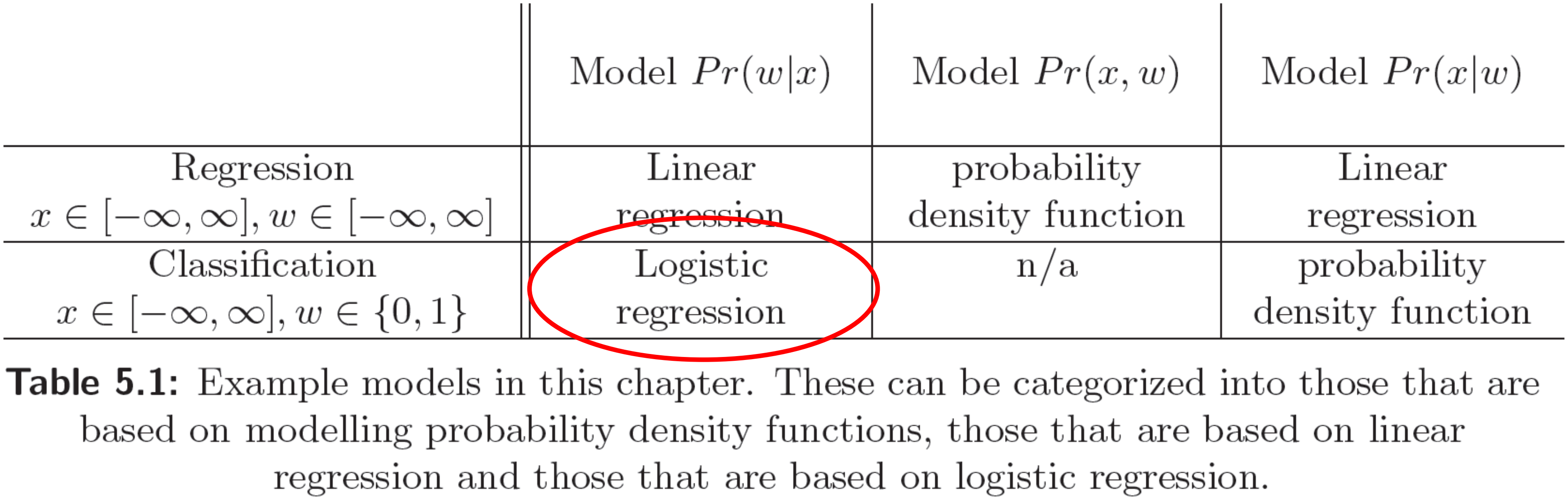
Chapter 9

Classification Models

Structure

- Logistic regression
- Bayesian logistic regression
- Non-linear logistic regression
- Kernelization and Gaussian process classification
- Incremental fitting, boosting and trees
- Multi-class classification
- Random classification trees
- Non-probabilistic classification
- Applications

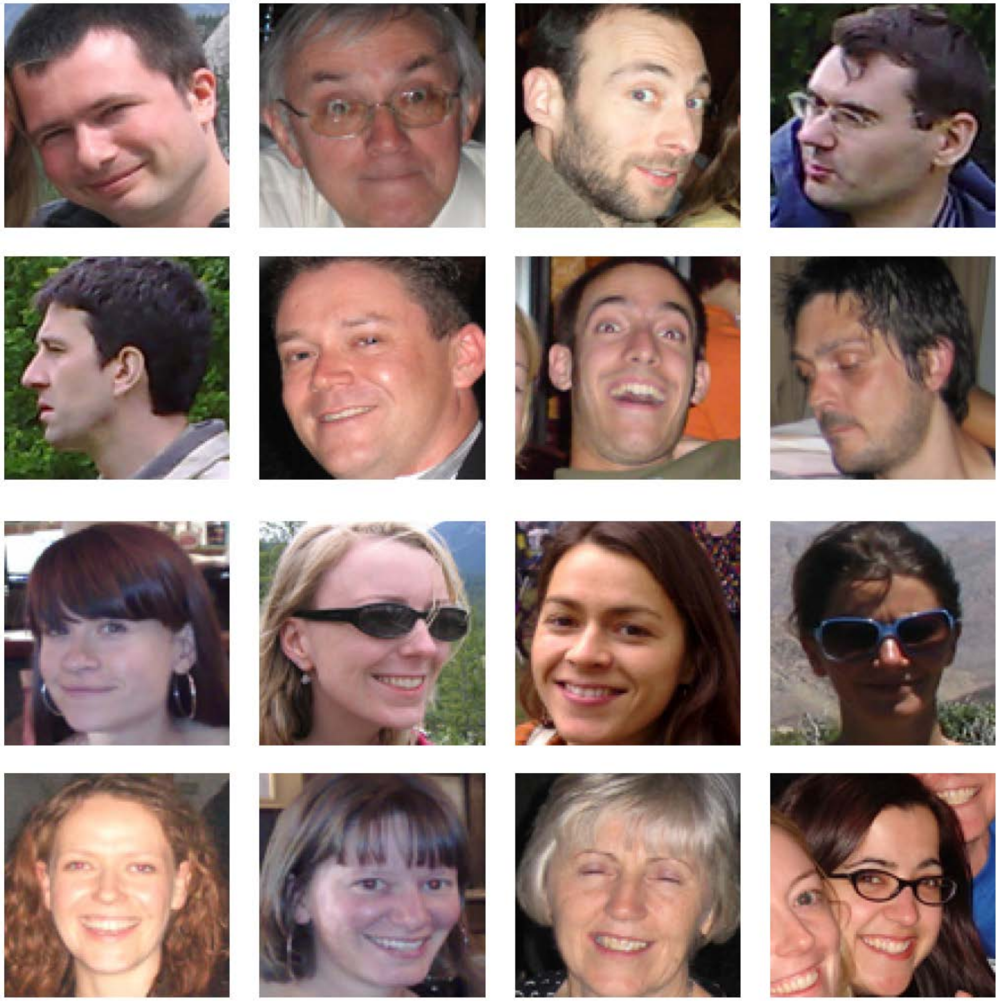
Models for machine vision



	Model $Pr(w\|x)$	Model $Pr(x,w)$	Model $Pr(x\|w)$
Regression $x \in [-\infty, \infty], w \in [-\infty, \infty]$	Linear regression	probability density function	Linear regression
Classification $x \in [-\infty, \infty], w \in \{0, 1\}$	Logistic regression	n/a	probability density function

Table 5.1: Example models in this chapter. These can be categorized into those that are based on modelling probability density functions, those that are based on linear regression and those that are based on logistic regression.

Example application: Gender Classification



Type 1: Model Pr(**w**|**x**) - Discriminative

How to model Pr(**w**|**x**)?

- Choose an appropriate form for Pr(**w**)
- Make parameters a function of **x**
- Function takes parameters θ that define its shape

Learning algorithm: learn parameters θ from training data **x,w**

Inference algorithm: just evaluate **Pr(w|x)**

Logistic Regression

Consider two class problem.

- Choose Bernoulli distribution over world.
- Make parameter λ a function of x

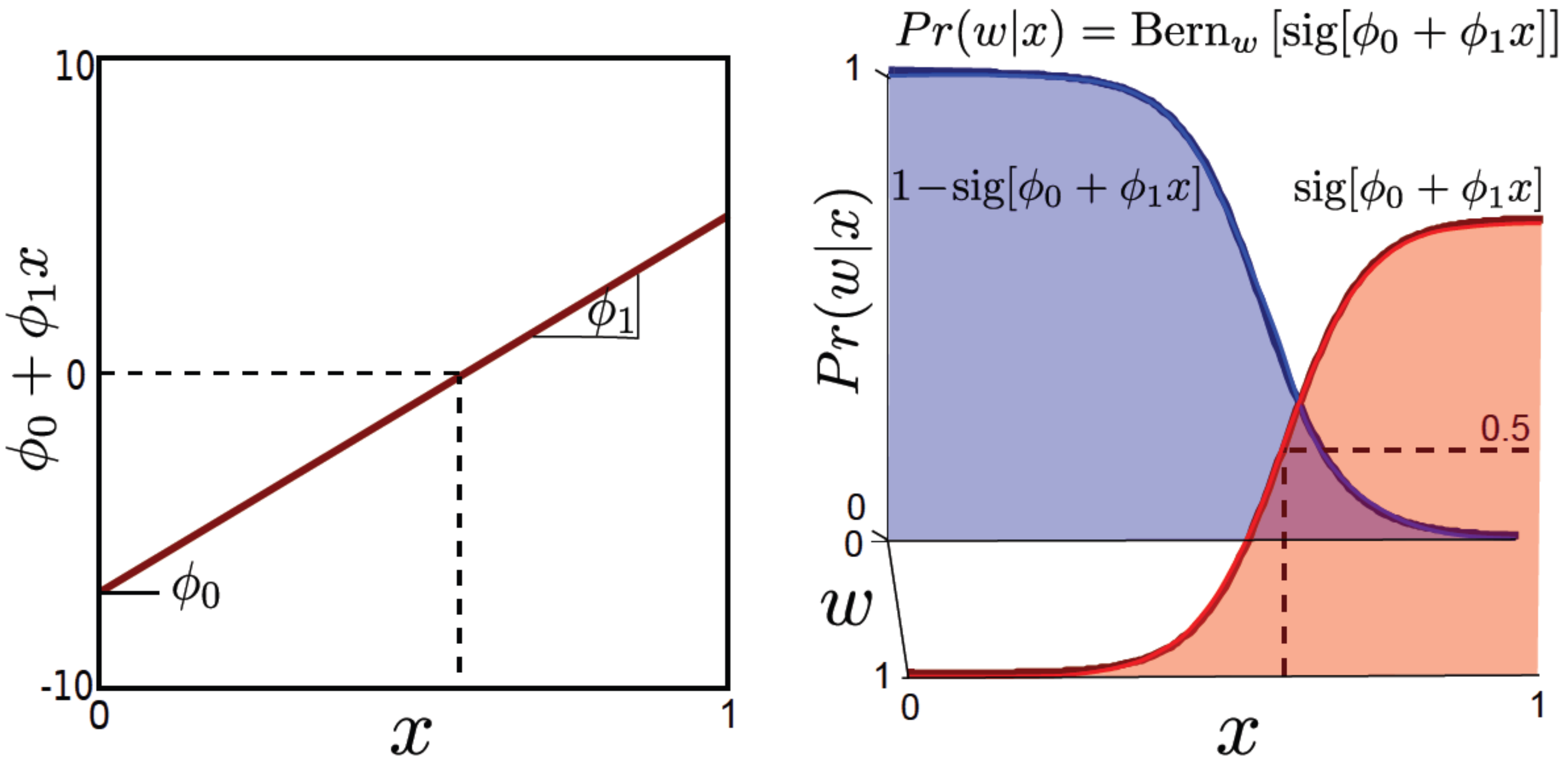
$$Pr(w|\phi_0, \boldsymbol{\phi}, \mathbf{x}) \quad = \quad \text{Bern}_w\left[\text{sig}[a]\right]$$

Model activation with a linear function

$$a = \phi_0 + \boldsymbol{\phi}^T \mathbf{x}$$

creates number between $[-\infty, \infty]$. Maps to $[0, 1]$ with

$$\text{sig}[a] = \frac{1}{1 + \exp[-a]}$$



Two parameters

$$\boldsymbol{\theta} = \{\phi_0, \phi_1\}$$

Learning by standard methods (ML,MAP, Bayesian)

Inference: Just evaluate Pr(w|x)

Neater Notation

$$Pr(w|\phi_0, \boldsymbol{\phi}, \mathbf{x}) \quad = \quad \text{Bern}_w\left[\text{sig}[a]\right]$$

To make notation easier to handle, we

- Attach a 1 to the start of every data vector

$$\mathbf{x}_i \leftarrow [1 \quad \mathbf{x}_i^T]^T$$

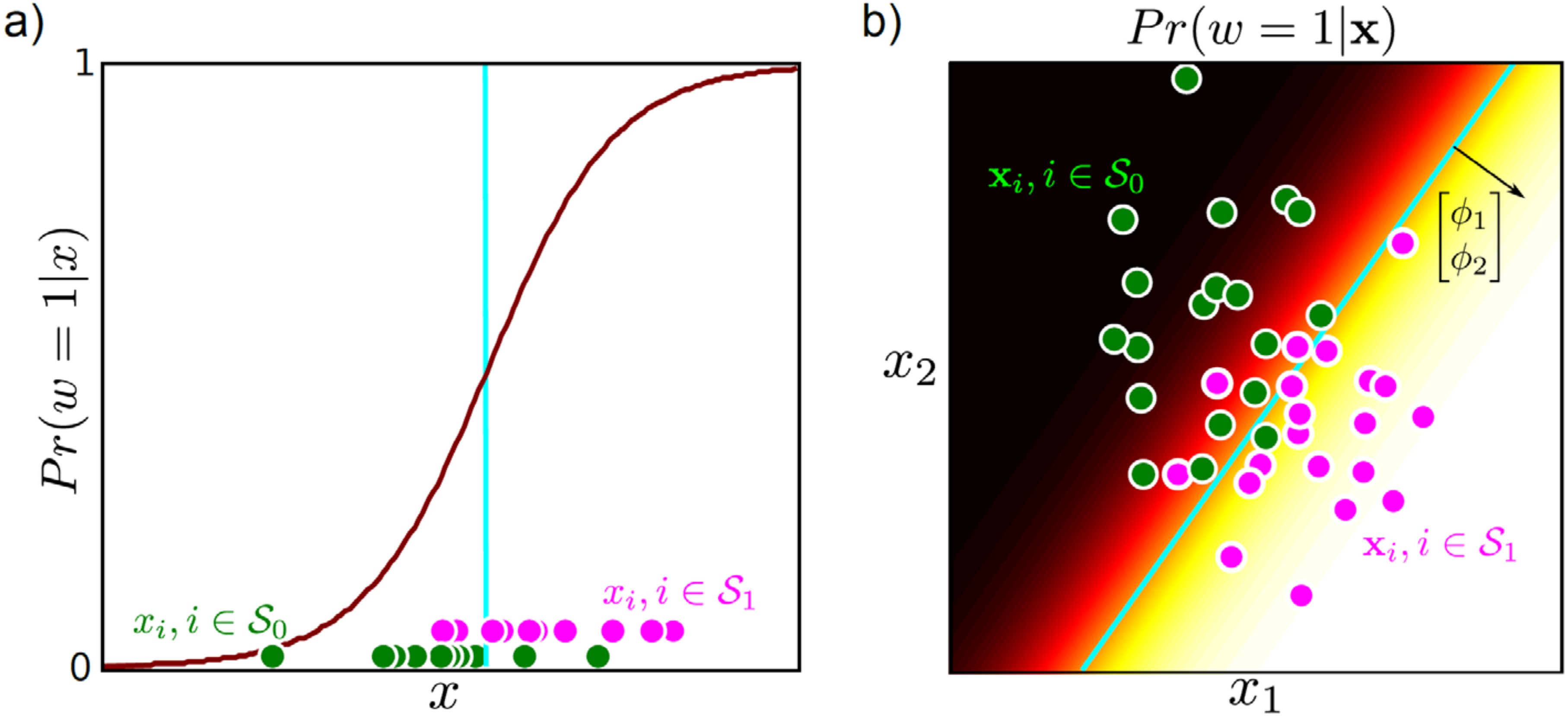
- Attach the offset to the start of the gradient vector ϕ

$$\boldsymbol{\phi} \leftarrow [\phi_0 \quad \boldsymbol{\phi}^T]^T$$

New model:

$$Pr(w|\boldsymbol{\phi}, \mathbf{x}) = \text{Bern}_w\left[\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}]}\right]$$

Logistic regression



$$Pr(w|\boldsymbol{\phi},\mathbf{x}) = \text{Bern}_w\left[\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}]}\right]$$

Maximum Likelihood

$$
\begin{aligned}
Pr(\mathbf{w}|\mathbf{X},\phi) &= \prod_{i=1}^{I}\lambda^{w_i}(1-\lambda)^{1-w_i}\\
&= \prod_{i=1}^{I}\left(\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right)^{w_i}\left(\frac{\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right)^{1-w_i}
\end{aligned}
$$

Take logarithm

$$
L = \sum_{i=1}^{I} w_i \log\left[\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right] + \sum_{i=1}^{I}(1-w_i)\log\left[\frac{\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right]
$$

Take derivative:

$$
\frac{\partial L}{\partial \boldsymbol{\phi}} = -\sum_{i=1}^{I}\left(\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]} - w_i\right)\mathbf{x}_i = -\sum_{i=1}^{I}\left(\text{sig}[a_i]-w_i\right)\mathbf{x}_i
$$

Derivatives

$$\frac{\partial L}{\partial \boldsymbol{\phi}} = -\sum_{i=1}^{I}\left(\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]} - w_i\right)\mathbf{x}_i = -\sum_{i=1}^{I}\left(\text{sig}[a_i] - w_i\right)\mathbf{x}_i$$

Unfortunately, there is no closed form solution– we cannot get an expression for ϕ in terms of x and w

Have to use a general purpose technique:

“iterative non-linear optimization”

Optimization

Goal:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left[f[\boldsymbol{\theta}] \right]$$

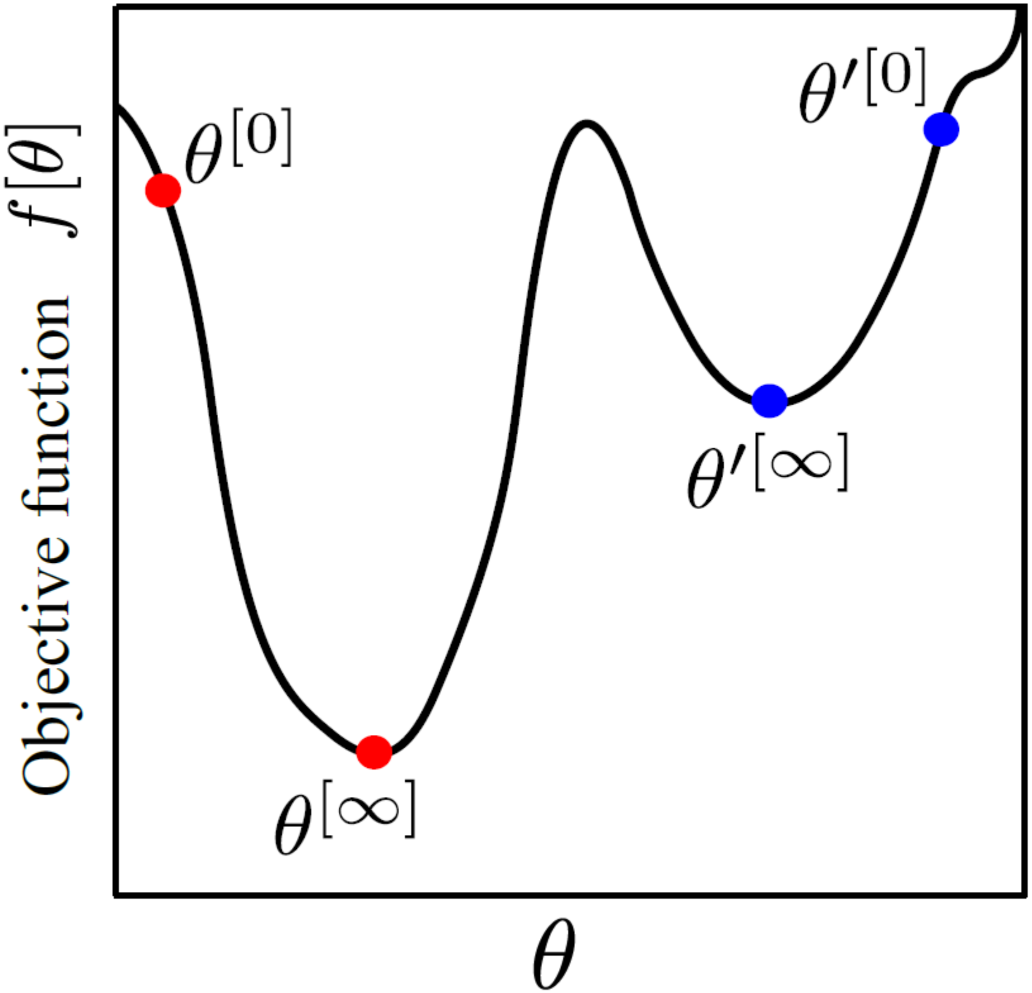
Cost function or
Objective function

How can we find the minimum?

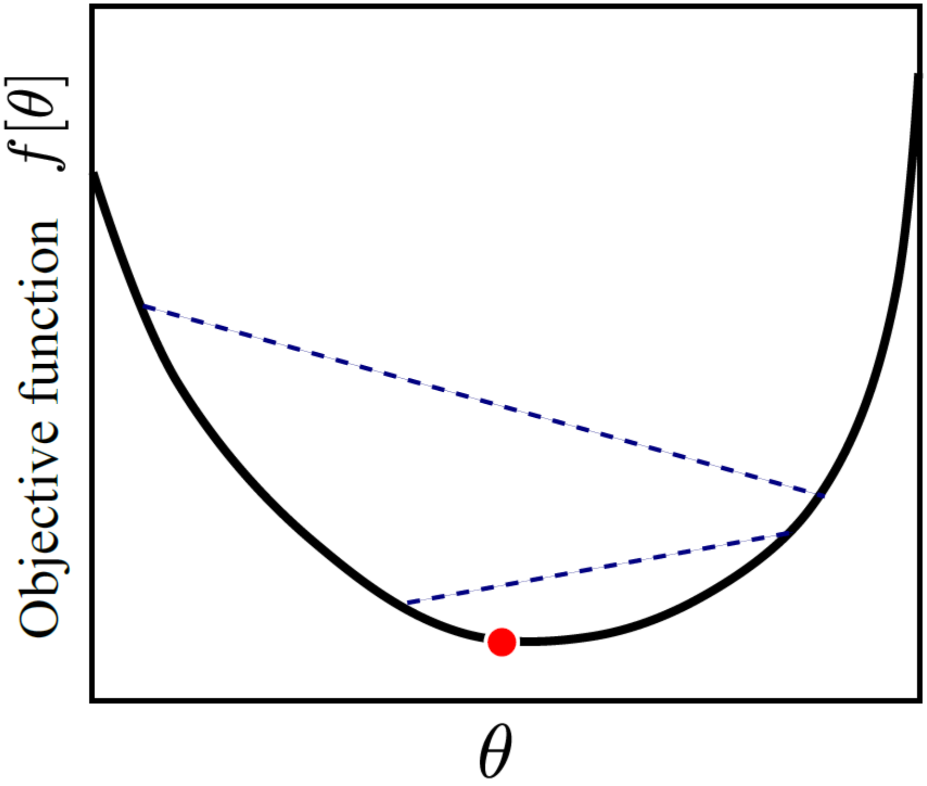
Basic idea:

- Start with estimate $\boldsymbol{\theta}^{[0]}$
- Take a series of small steps to $\boldsymbol{\theta}^{[1]}, \boldsymbol{\theta}^{[2]} \ldots \boldsymbol{\theta}^{[\infty]}$
- Make sure that each step decreases cost
- When can't improve, then must be at minimum

Local Minima

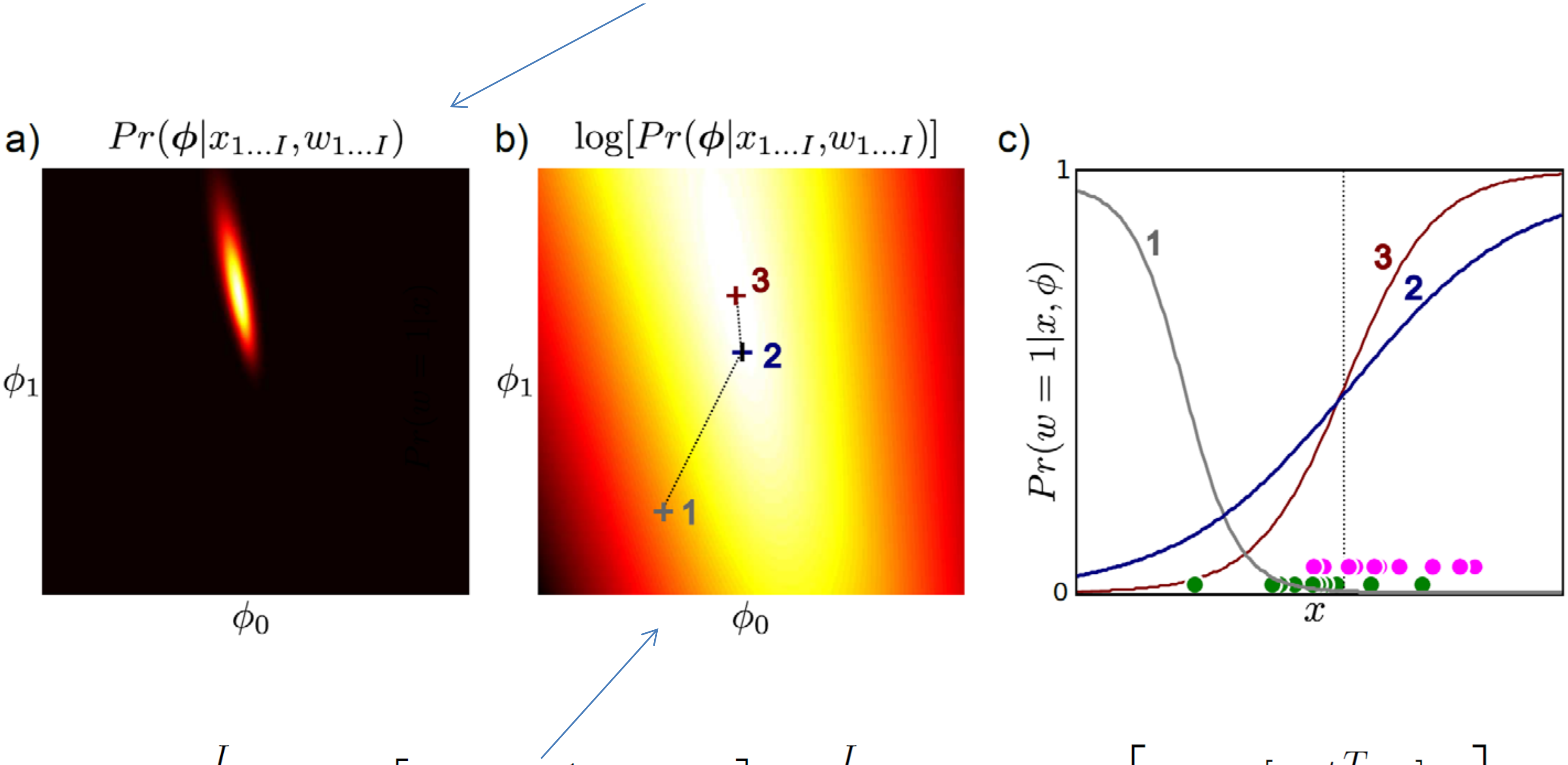


Convexity



If a function is convex, then it has only a single minimum.
Can tell if a function is convex by looking at 2nd derivatives

$$Pr(\mathbf{w}|\mathbf{X},\boldsymbol{\phi}) = \prod_{i=1}^{I}\left(\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right)^{w_i}\left(\frac{\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right)^{1-w_i}$$



$$L = \sum_{i=1}^{I} w_i \log\left[\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right] + \sum_{i=1}^{I}(1-w_i)\log\left[\frac{\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right]$$

Gradient Based Optimization

- Choose a search direction **s** based on the local properties of the function

- Perform an intensive search along the chosen direction. This is called *line search*

$$\hat{\lambda} = \underset{\lambda}{\mathrm{argmin}} \left[f[\boldsymbol{\theta}^{[t]} + \lambda \mathbf{s}] \right]$$

- Then set

$$\boldsymbol{\theta}^{[t+1]} = \boldsymbol{\theta}^{[t]} + \hat{\lambda}\mathbf{s}$$

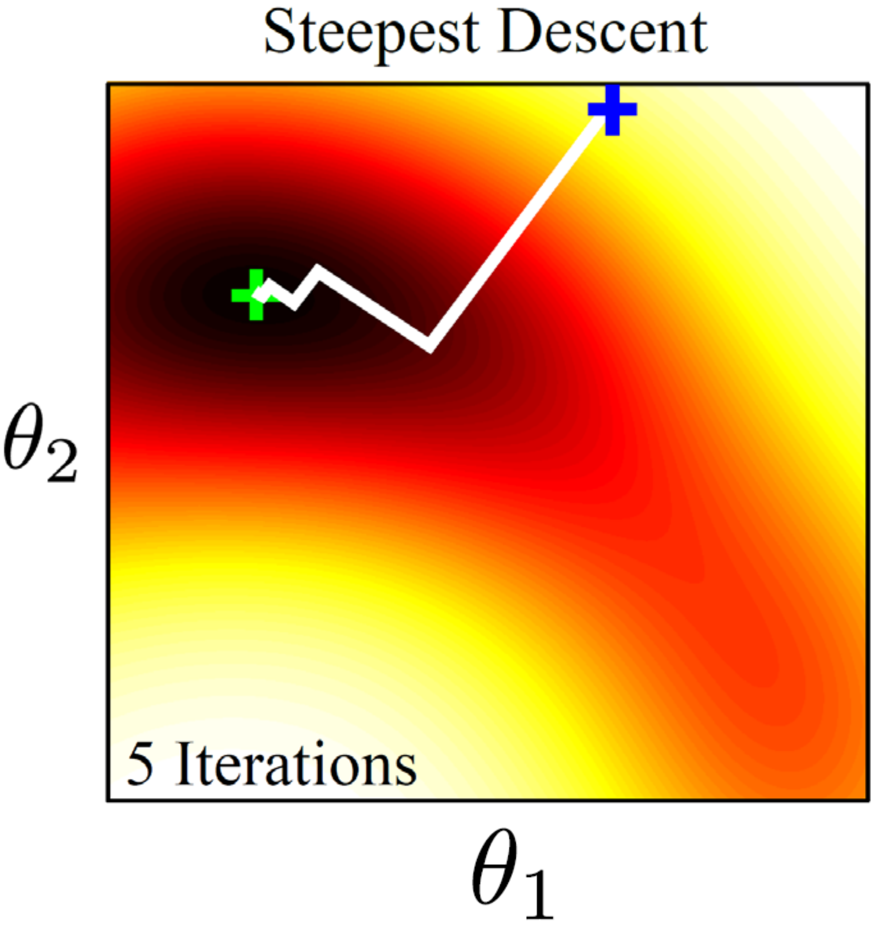
Gradient Descent

Consider standing on a hillside

Look at gradient where you are standing

Find the steepest direction downhill

Walk in that direction for some distance (line search)



Finite differences

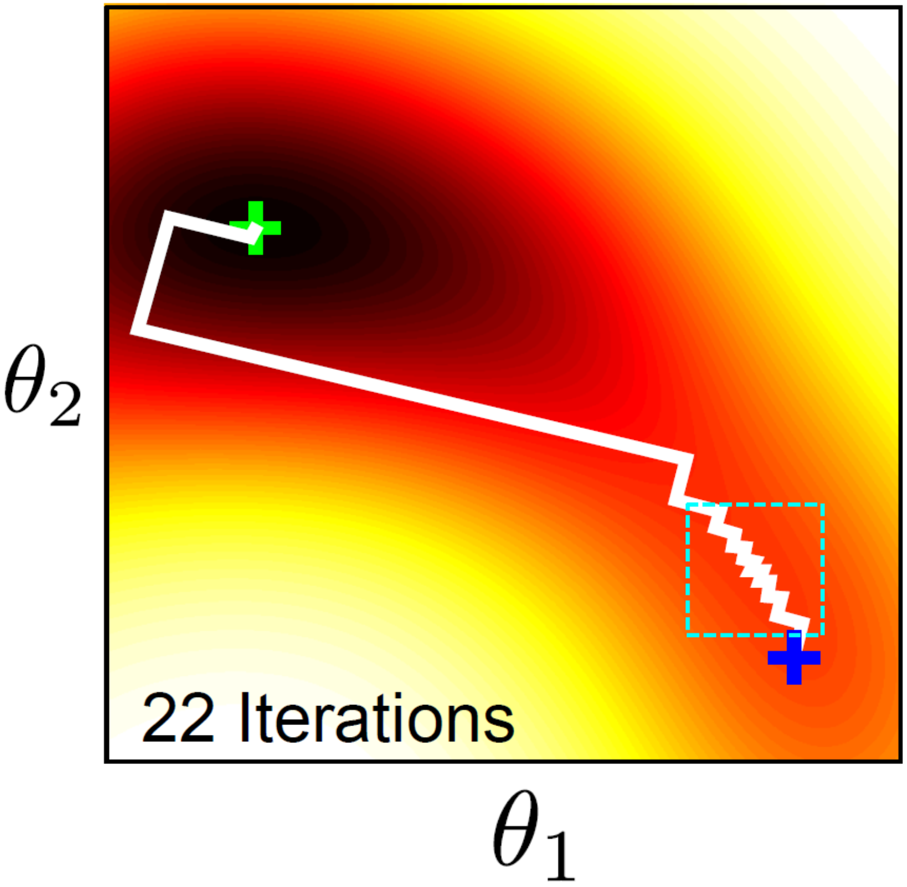
What if we can't compute the gradient?

Compute finite difference approximation:

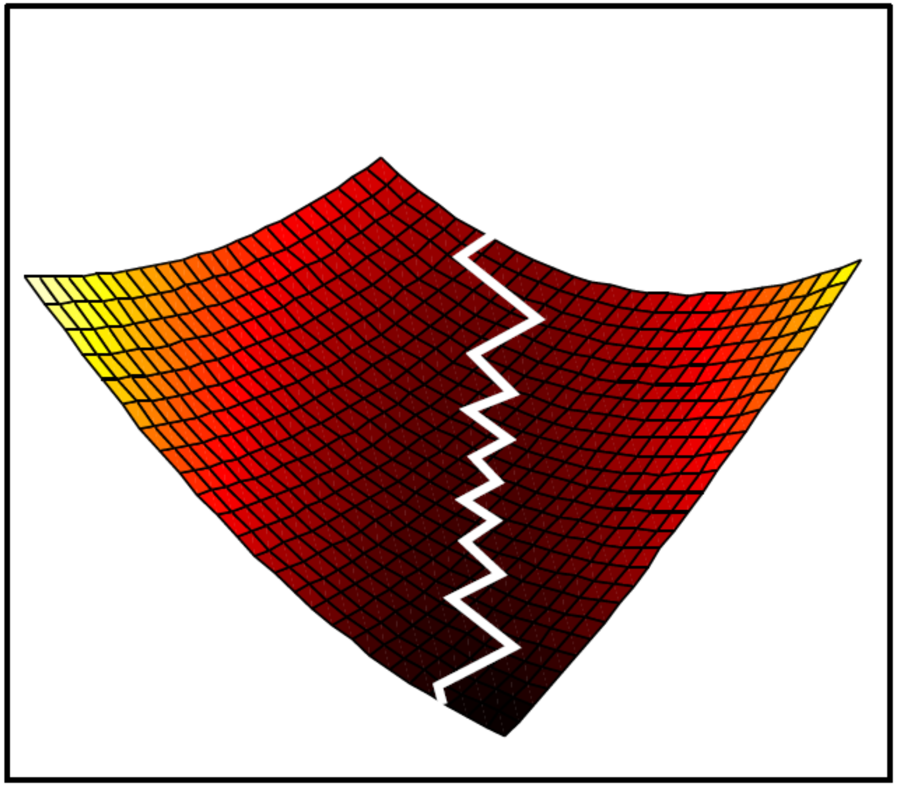
$$\frac{\partial f}{\partial \theta_j} \approx \frac{f\left[\boldsymbol{\theta} + a\mathbf{e}_j\right] - f\left[\boldsymbol{\theta}\right]}{a}$$

where $\mathbf{e}_j$ is the unit vector in the j^{th} direction

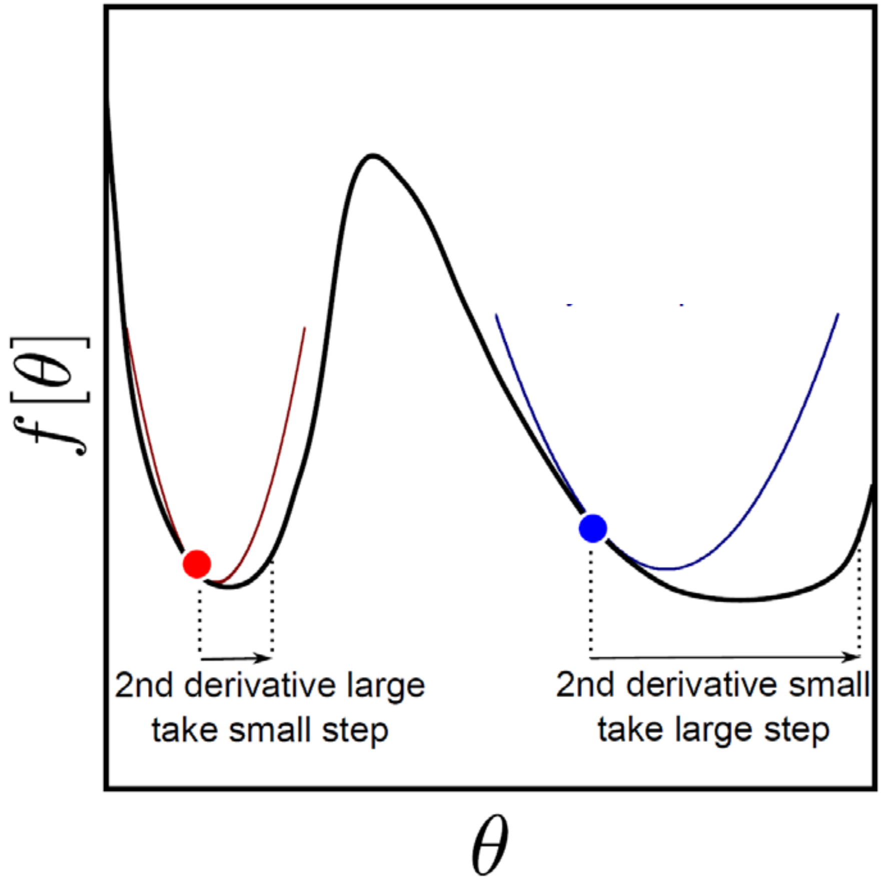
Steepest Descent Problems



Close up



Second Derivatives



In higher dimensions, 2nd derivatives change how much we should move in the different directions: changes best direction to move in.

Newton’s Method

Approximate function with Taylor expansion

$$f[\boldsymbol{\theta}] \approx f[\boldsymbol{\theta}^{[t]}] + (\boldsymbol{\theta} - \boldsymbol{\theta}^{[t]})^T \left.\frac{\partial f}{\partial \boldsymbol{\theta}}\right|_{\theta^{[t]}} + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^{[t]})^T \left.\frac{\partial^2 f}{\partial \boldsymbol{\theta}^2}\right|_{\theta^{[t]}} (\boldsymbol{\theta} - \boldsymbol{\theta}^{[t]})$$

Take derivative

$$\frac{\partial f}{\partial \boldsymbol{\theta}} \approx \left.\frac{\partial f}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}^{[t]}} + \left.\frac{\partial^2 f}{\partial \boldsymbol{\theta}^2}\right|_{\boldsymbol{\theta}^{[t]}} (\boldsymbol{\theta} - \boldsymbol{\theta}^{[t]}) = 0$$

Re-arrange

$$\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^{[t]} - \left(\frac{\partial^2 f}{\partial \boldsymbol{\theta}^2}\right)^{-1} \frac{\partial f}{\partial \boldsymbol{\theta}}$$

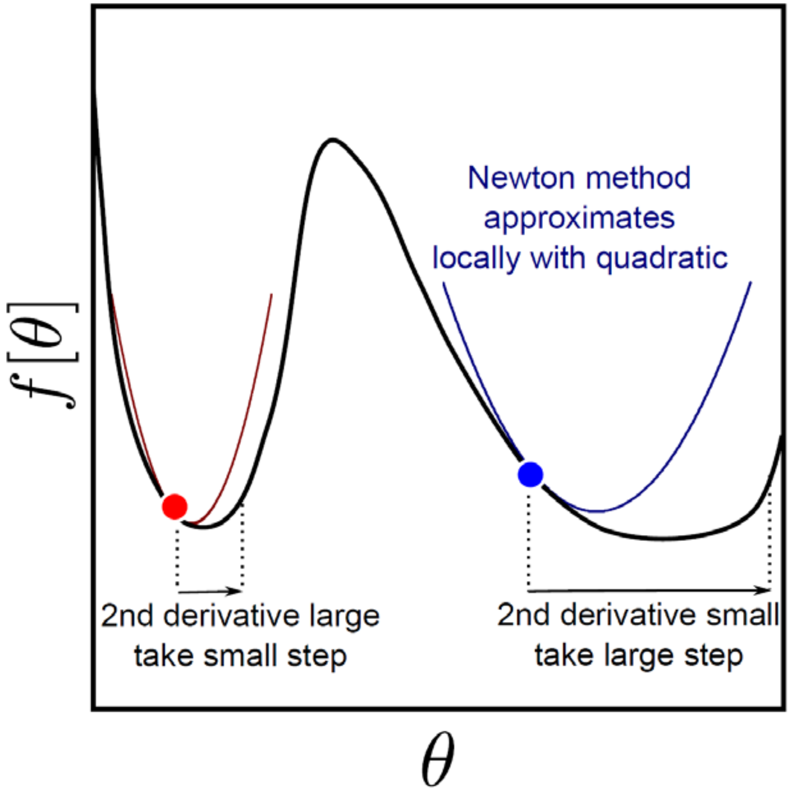
(derivatives taken at time)

Adding line search

$$\boldsymbol{\theta}^{[t+1]} = \boldsymbol{\theta}^{[t]} - \lambda \left(\frac{\partial^2 f}{\partial \boldsymbol{\theta}^2}\right)^{-1} \frac{\partial f}{\partial \boldsymbol{\theta}}$$

Newton's Method

$$\boldsymbol{\theta}^{[t+1]} = \boldsymbol{\theta}^{[t]} - \lambda \left(\frac{\partial^2 f}{\partial \boldsymbol{\theta}^2} \right)^{-1} \frac{\partial f}{\partial \boldsymbol{\theta}}$$

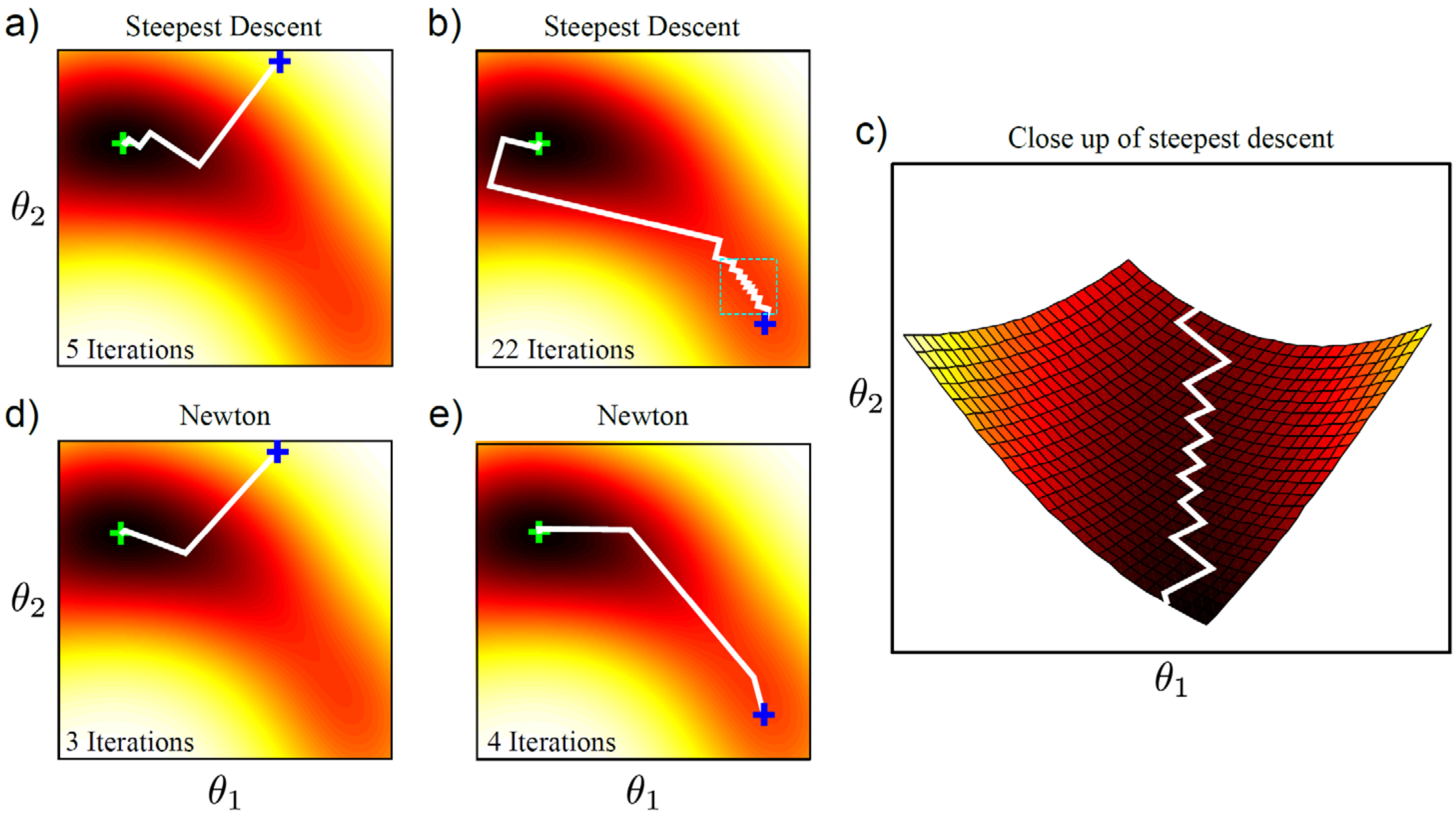


Matrix of second derivatives is called the Hessian.

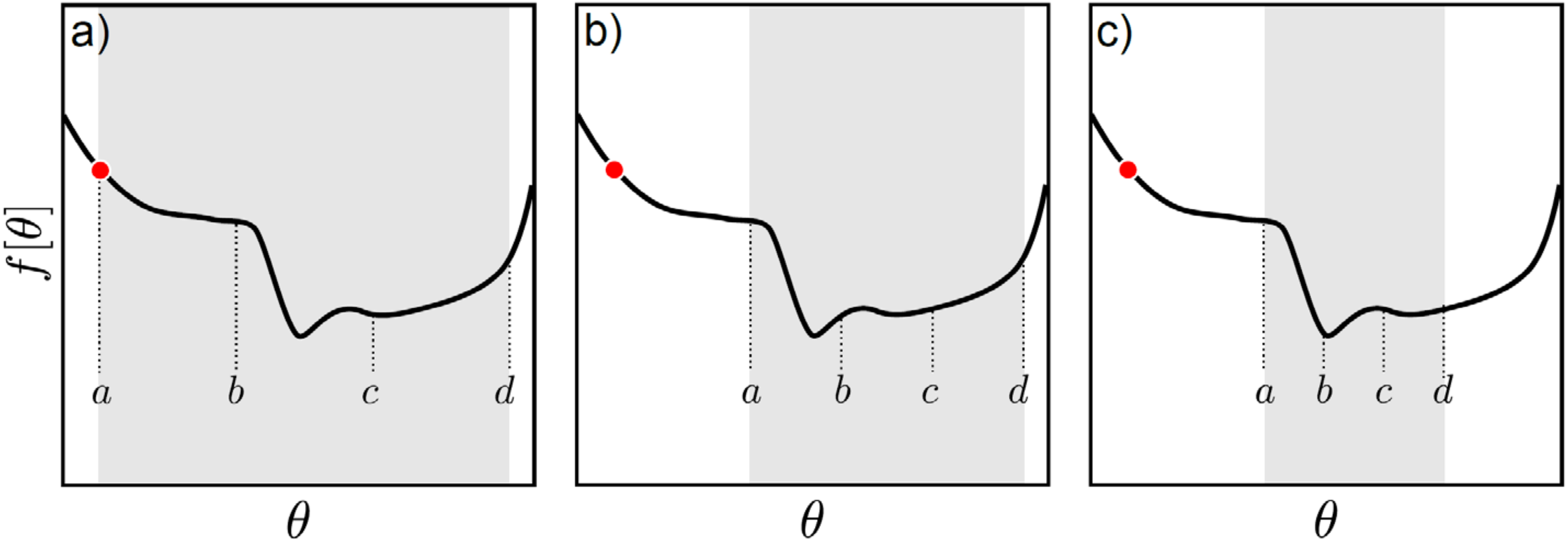
Expensive to compute via finite differences.

If positive definite, then convex

Newton vs. Steepest Descent



Line Search



Gradually narrow down range

Optimization for Logistic Regression

$$\phi^{[t]} = \phi^{[t-1]} + \alpha \left(\frac{\partial^2 L}{\partial \phi^2} \right)^{-1} \frac{\partial L}{\partial \phi}$$

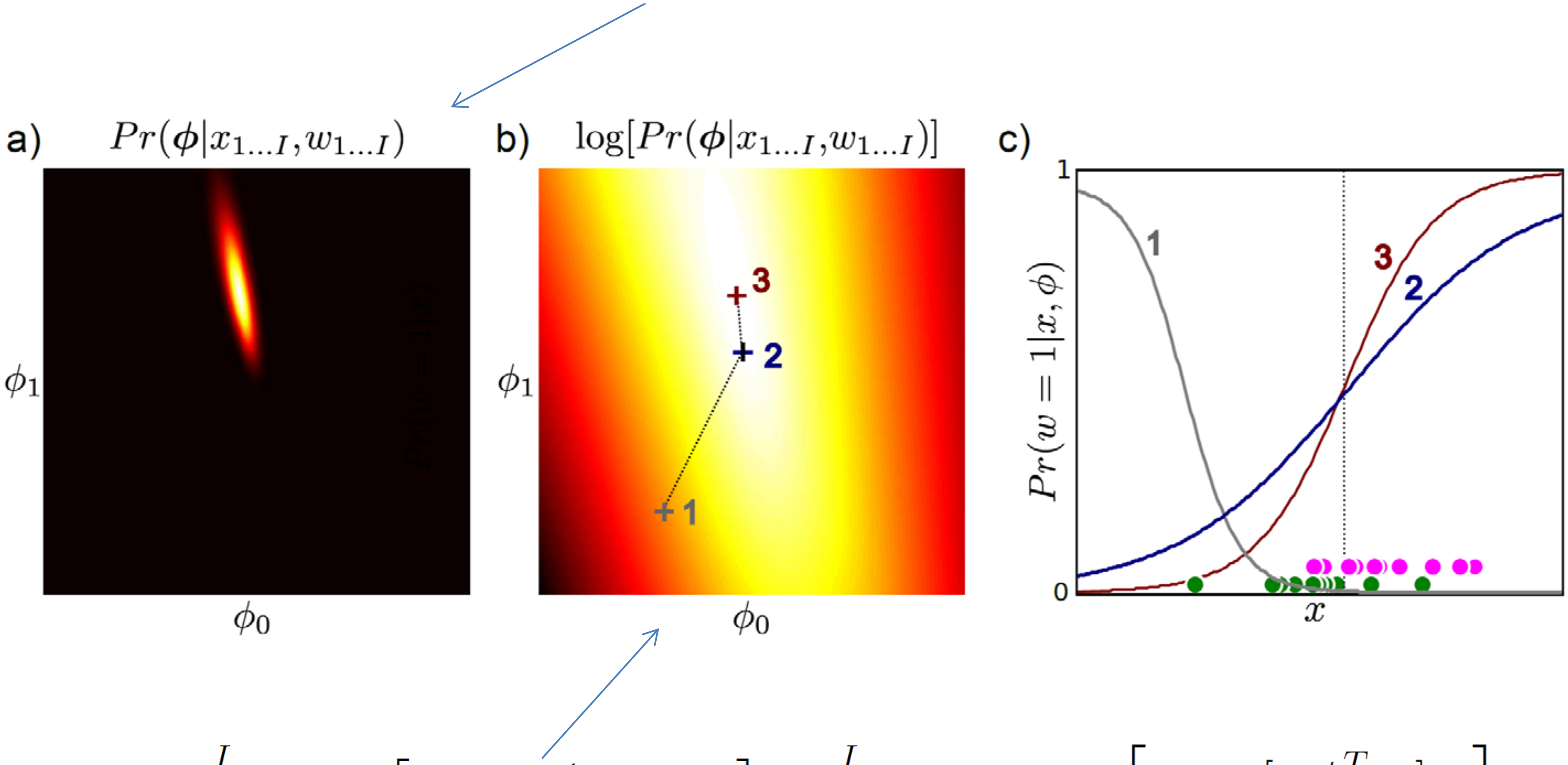
Derivatives of log likelihood:

$$\frac{\partial L}{\partial \phi} = -\sum_{i=1}^{I} (\text{sig}[a_i] - w_i)\mathbf{x}_i$$

$$\frac{\partial^2 L}{\partial \phi^2} = -\sum_{i=1}^{I} \text{sig}[a_i](1 - \text{sig}[a_i])\mathbf{x}_i\mathbf{x}_i^T$$

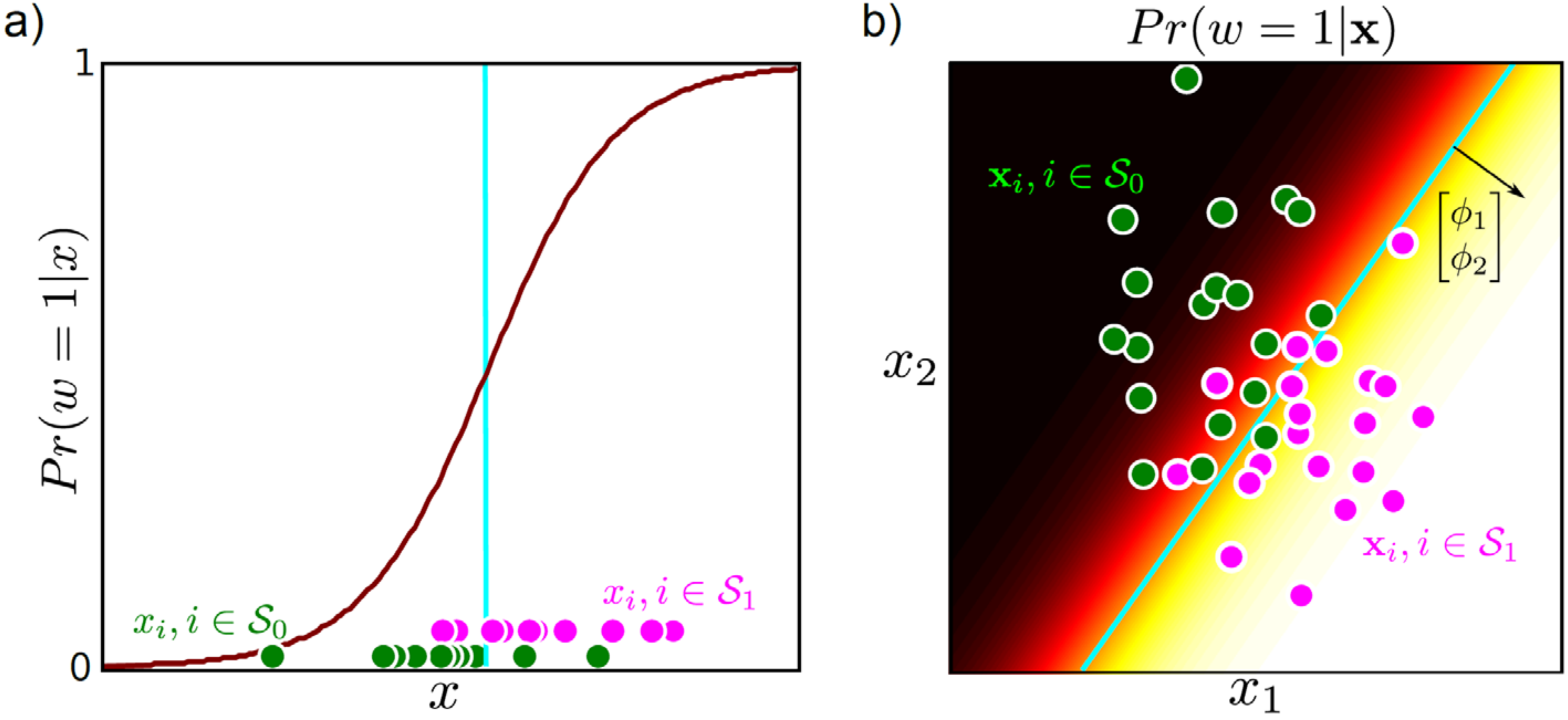
Positive definite!

$$Pr(\mathbf{w}|\mathbf{X},\boldsymbol{\phi}) = \prod_{i=1}^{I}\left(\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right)^{w_i}\left(\frac{\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right)^{1-w_i}$$



$$L = \sum_{i=1}^{I} w_i \log\left[\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right] + \sum_{i=1}^{I}(1-w_i)\log\left[\frac{\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right]$$

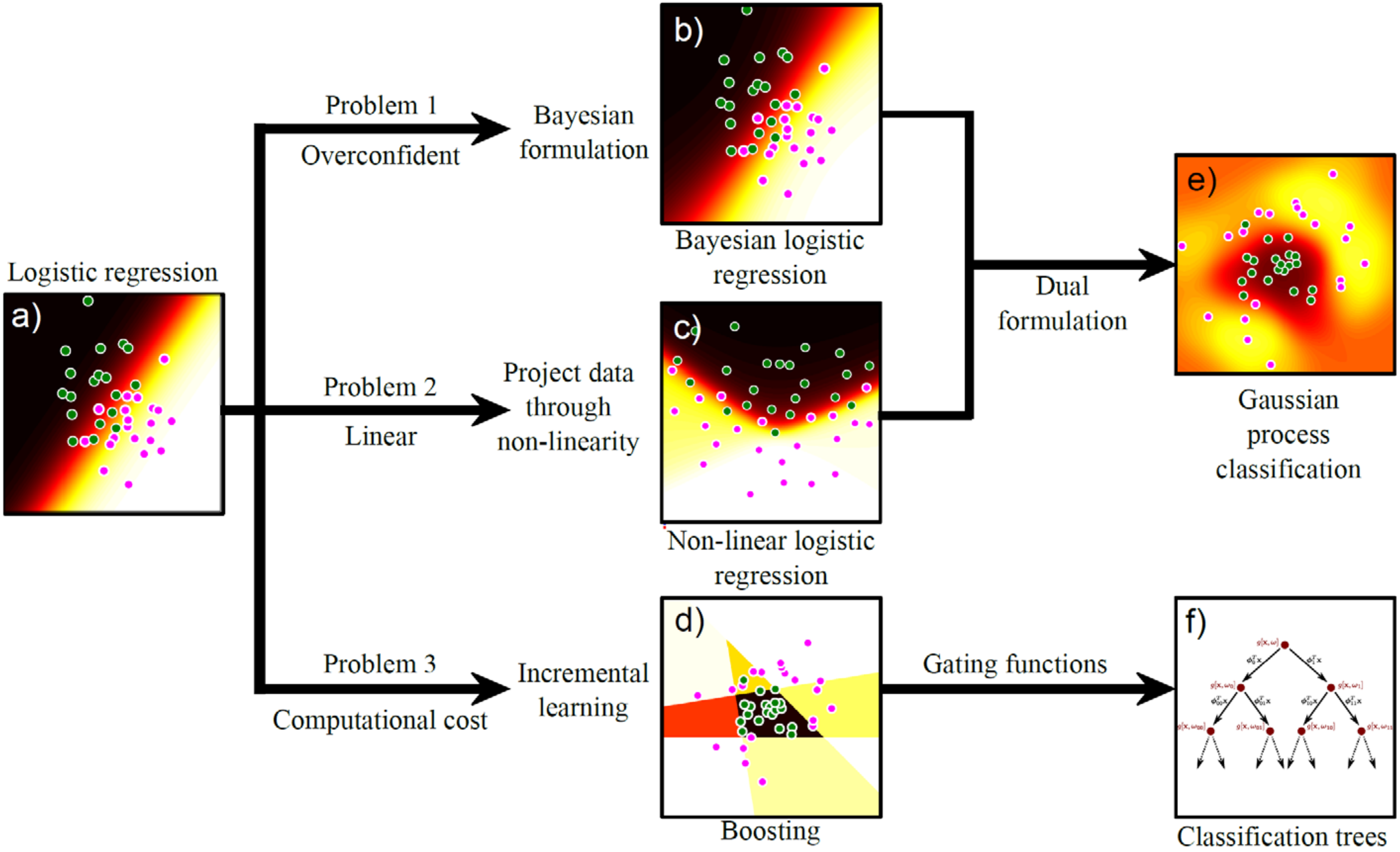
Maximum likelihood fits



$$Pr(w|\boldsymbol{\phi}, \mathbf{x}) = \text{Bern}_w\left[\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}]}\right]$$

Structure

- Logistic regression
- Bayesian logistic regression
- Non-linear logistic regression
- Kernelization and Gaussian process classification
- Incremental fitting, boosting and trees
- Multi-class classification
- Random classification trees
- Non-probabilistic classification
- Applications



Logistic regression

Problem 1

Overconfident

Bayesian formulation

Bayesian logistic regression

Problem 2

Linear

Project data through non-linearity

Non-linear logistic regression

Problem 3

Computational cost

Incremental learning

Boosting

Dual formulation

Gaussian process classification

Gating functions

Classification trees

Bayesian Logistic Regression

Likelihood:

$$Pr(\mathbf{w}|\mathbf{X},\boldsymbol{\phi}) = \prod_{i=1}^{I}\left(\frac{1}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right)^{w_i}\left(\frac{\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}{1+\exp[-\boldsymbol{\phi}^T\mathbf{x}_i]}\right)^{1-w_i}$$

Prior (no conjugate):

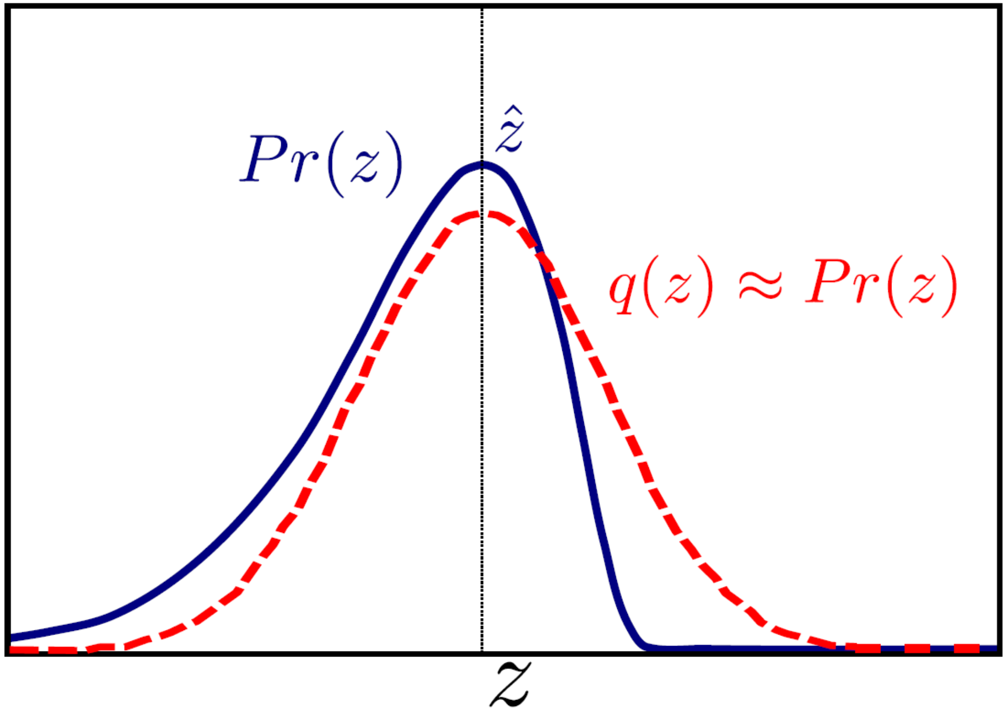
$$Pr(\boldsymbol{\phi}) = \text{Norm}_{\boldsymbol{\phi}}[\mathbf{0},\sigma_p^2\mathbf{I}]$$

Apply Bayes' rule:

$$Pr(\boldsymbol{\phi}|\mathbf{X},\mathbf{w}) = \frac{Pr(\mathbf{w}|\mathbf{X},\boldsymbol{\phi})Pr(\boldsymbol{\phi})}{Pr(\mathbf{w}|\mathbf{X})}$$

(no closed form solution for posterior)

Laplace Approximation



Approximate posterior distribution with normal

- Set mean to MAP estimate
- Set covariance to match that at MAP estimate (actually: get 2nd derivatives to agree)

Laplace Approximation

Find MAP solution by optimizing

$$L = \sum_{i=1}^{I} \log[Pr(w_i|\mathbf{x}_i, \boldsymbol{\phi})] + \log[Pr(\boldsymbol{\phi})]$$

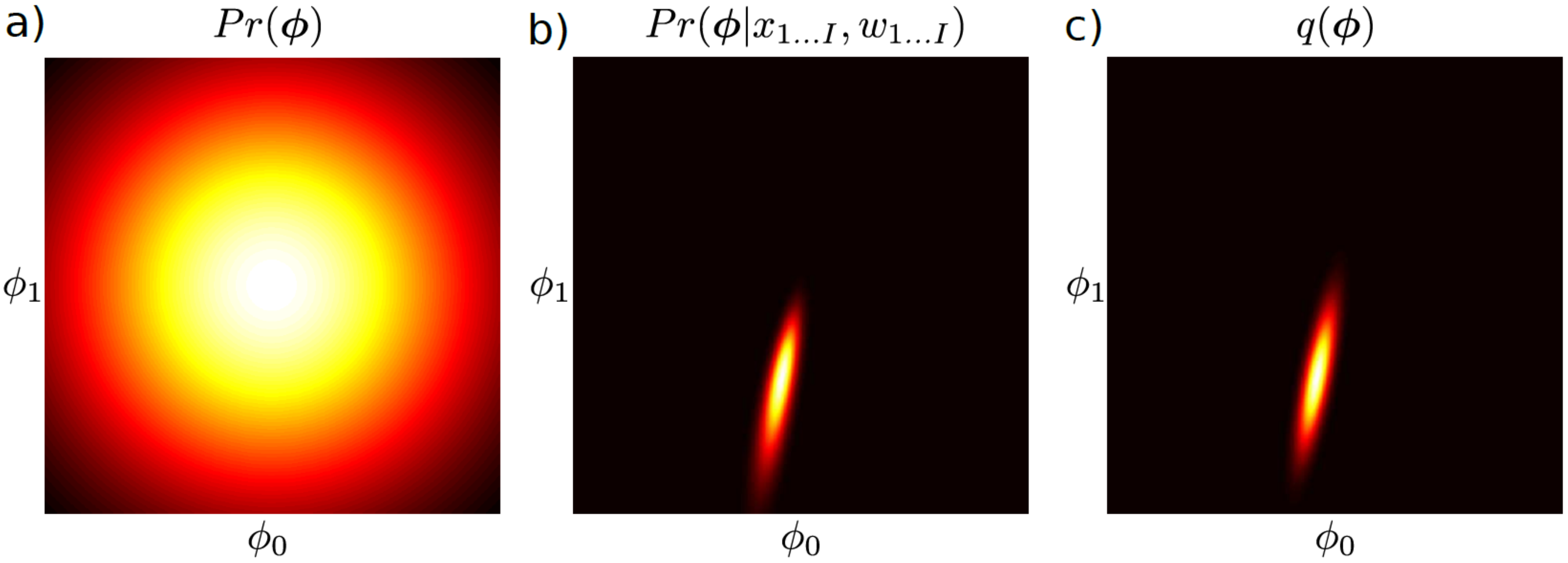
Approximate with normal

$$Pr(\boldsymbol{\phi}|\mathbf{X}, \mathbf{w}) \approx q(\boldsymbol{\phi}) = \text{Norm}_{\boldsymbol{\phi}}[\boldsymbol{\mu}, \boldsymbol{\Sigma}]$$

where

$$\begin{aligned} \boldsymbol{\mu} &= \hat{\boldsymbol{\phi}} \\ \boldsymbol{\Sigma} &= -\left(\frac{\partial^2 L}{\partial \boldsymbol{\phi}^2}\right)^{-1}\Bigg|_{\boldsymbol{\phi}=\hat{\boldsymbol{\phi}}} \end{aligned}$$

Laplace Approximation



Prior

Actual posterior

Approximated

Inference

$$
\begin{aligned}
Pr(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w}) &= \int Pr(w^*|\mathbf{x}^*, \phi) Pr(\phi|\mathbf{X}, \mathbf{w}) d\phi \\
&\approx \int Pr(w^*|\mathbf{x}^*, \boldsymbol{\phi}) q(\boldsymbol{\phi}) d\boldsymbol{\phi}.
\end{aligned}
$$

Can re-express in terms of activation

$$
Pr(w^*|\mathbf{x}^*, \mathbf{X}, \mathbf{w}) \approx \int Pr(w^*|a) Pr(a) da
$$

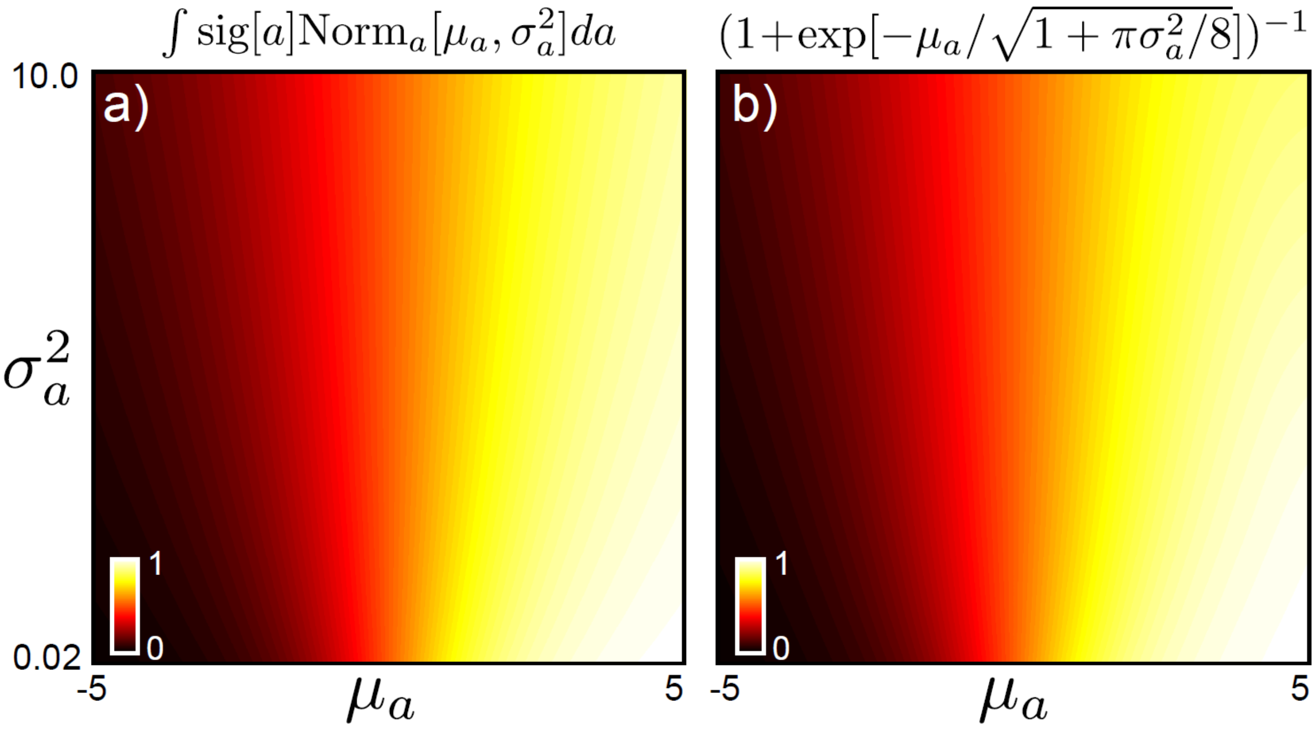
Using transformation properties of normal distributions

$$
\begin{aligned}
Pr(a) = Pr(\boldsymbol{\phi}^T \mathbf{x}^*) &= \text{Norm}_a[\boldsymbol{\mu}^T \mathbf{x}^*, \mathbf{x}^{*T} \boldsymbol{\Sigma} \mathbf{x}] \\
&= \text{Norm}_a[\mu_a, \sigma_a^2],
\end{aligned}
$$

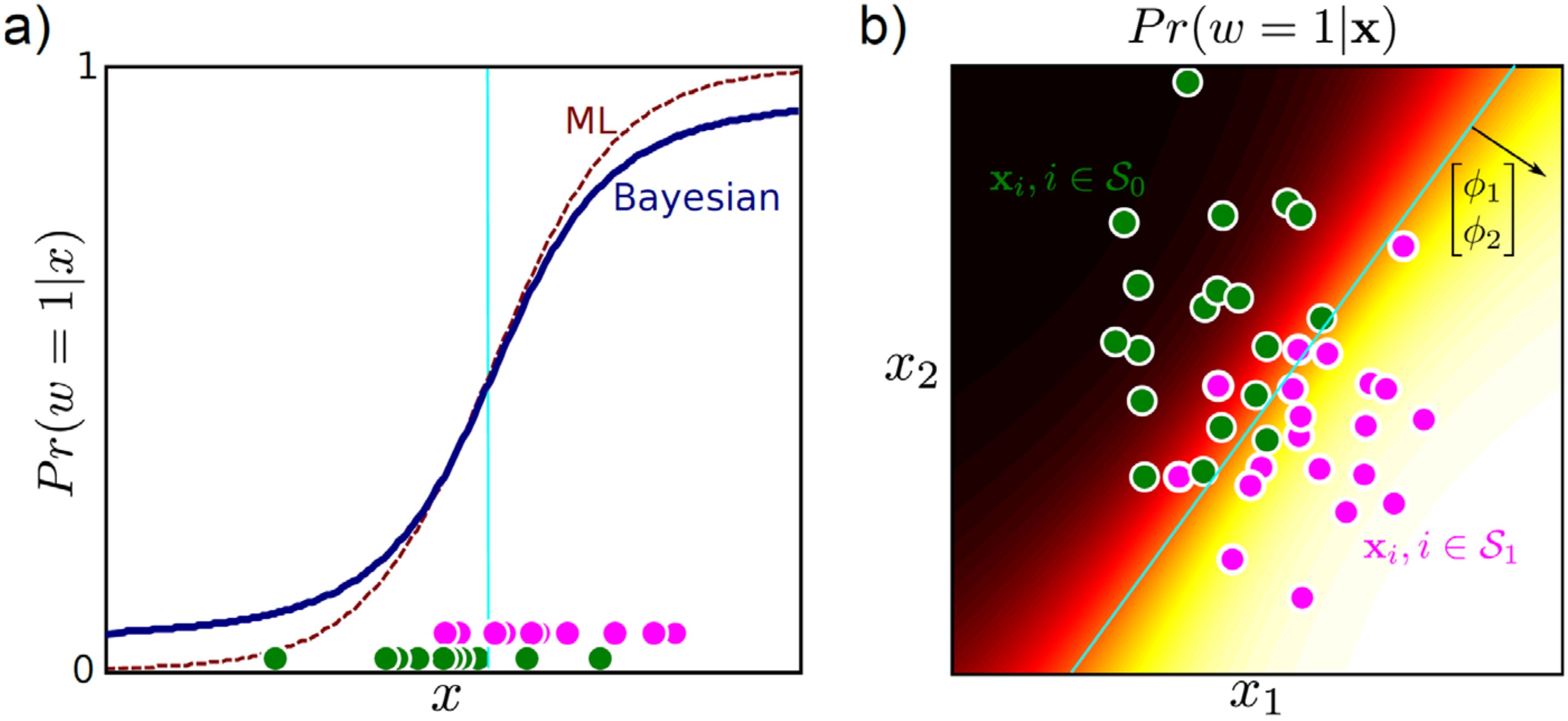
Approximation of Integral

(Or perform numerical integration on *a* – which is 1D)

$$\int Pr(w^*|a)\text{Norm}_a[\mu_a,\sigma_a^2]da \approx \frac{1}{1+\exp[-\mu_a/\sqrt{1+\pi\sigma_a^2/8}]}$$

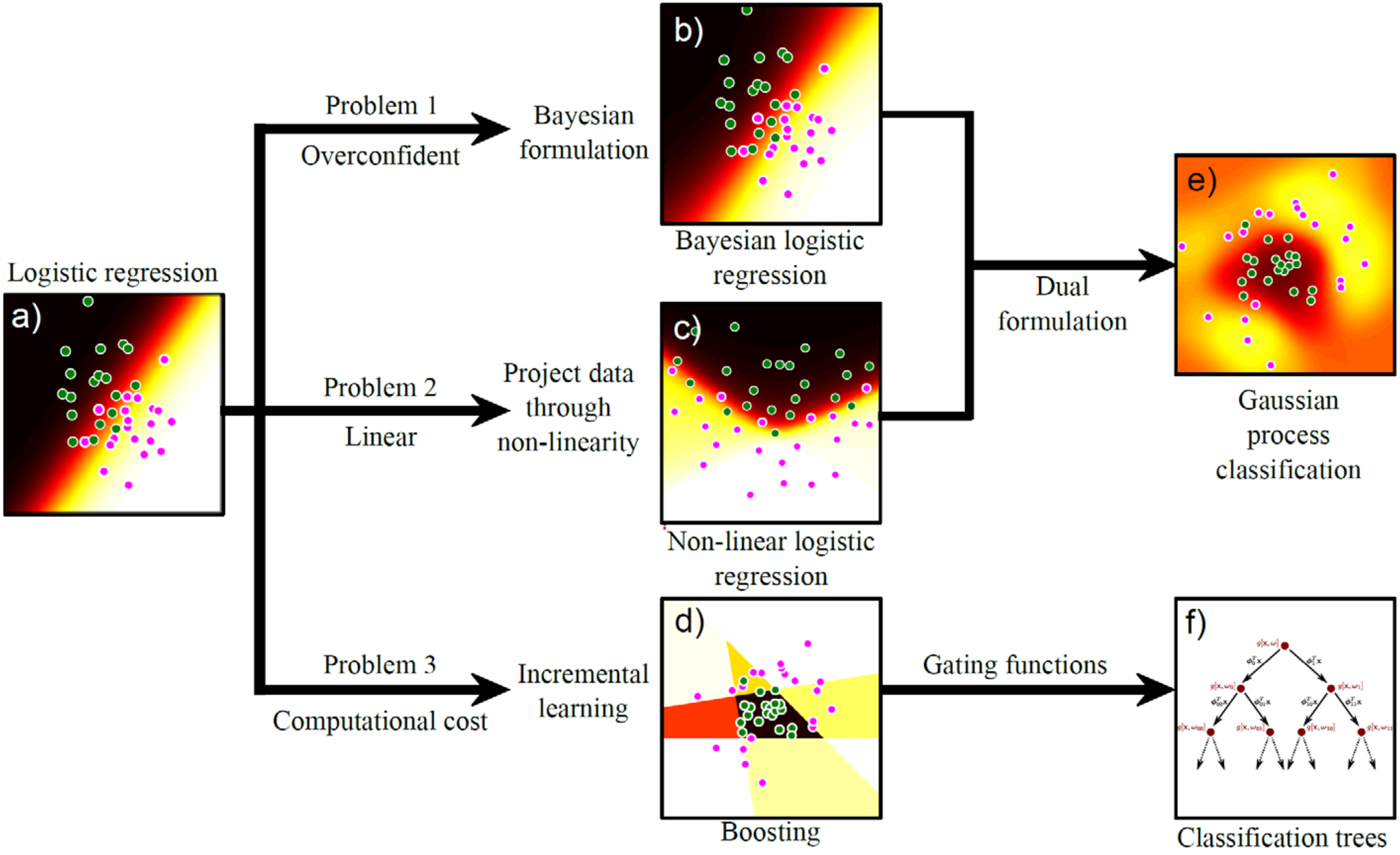


Bayesian Solution



Structure

- Logistic regression
- Bayesian logistic regression
- Non-linear logistic regression
- Kernelization and Gaussian process classification
- Incremental fitting, boosting and trees
- Multi-class classification
- Random classification trees
- Non-probabilistic classification
- Applications



Logistic regression

Problem 1

Overconfident

Bayesian formulation

Bayesian logistic regression

Problem 2

Linear

Project data through non-linearity

Non-linear logistic regression

Problem 3

Computational cost

Incremental learning

Boosting

Dual formulation

Gaussian process classification

Gating functions

Classification trees

Non-linear logistic regression

Same idea as for regression.

- Apply non-linear transformation

$$\mathbf{z} = \mathbf{f}[\mathbf{x}]$$

- Build model as usual

$$\begin{aligned} Pr(w=1|\mathbf{x},\boldsymbol{\phi}) &= \text{Bern}_w\left[\text{sig}[\boldsymbol{\phi}^T\mathbf{z}]\right] \\ &= \text{Bern}_w\left[\text{sig}[\boldsymbol{\phi}^T\mathbf{f}[\mathbf{x}]]\right] \end{aligned}$$

Non-linear logistic regression

Example transformations:

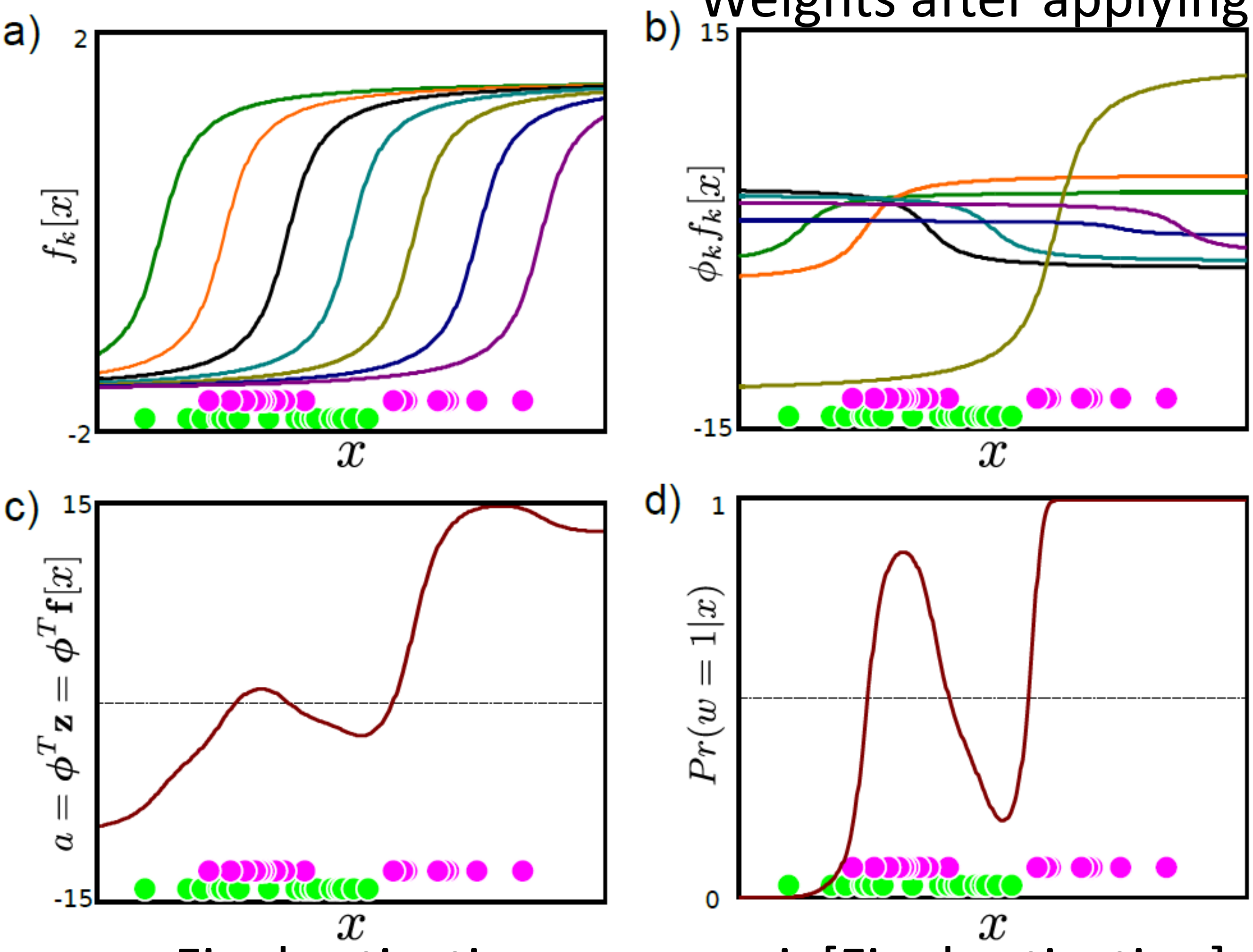
- Heaviside Step functions of projections: $z_k = \text{Heaviside}[\boldsymbol{\alpha}_k^T \mathbf{x}]$
- Arc tan functions of projections: $z_k = \arctan[\boldsymbol{\alpha}_k^T \mathbf{x}]$
- Radial basis functions: $z_k = \exp\left[-\frac{1}{\lambda_0}(\mathbf{x}-\boldsymbol{\alpha}_k)^T(\mathbf{x}-\boldsymbol{\alpha}_k)\right]$

Fit using optimization (also transformation parameters **α**):

$$\begin{aligned}\frac{\partial L}{\partial \boldsymbol{\theta}} &= -\sum_{i=1}^{I}(w_i - \text{sig}[a_i])\frac{\partial a_i}{\partial \boldsymbol{\theta}}\\ \frac{\partial^2 L}{\partial \boldsymbol{\theta}^2} &= -\sum_{i=1}^{I}\text{sig}[a_i](\text{sig}[a_i]-1)\frac{\partial a_i}{\partial \boldsymbol{\theta}}\frac{\partial a_i}{\partial \boldsymbol{\theta}}^T - (w_i - \text{sig}[a_i])\frac{\partial^2 a_i}{\partial \boldsymbol{\theta}^2}\end{aligned}$$

Non-linear logistic regression in 1D

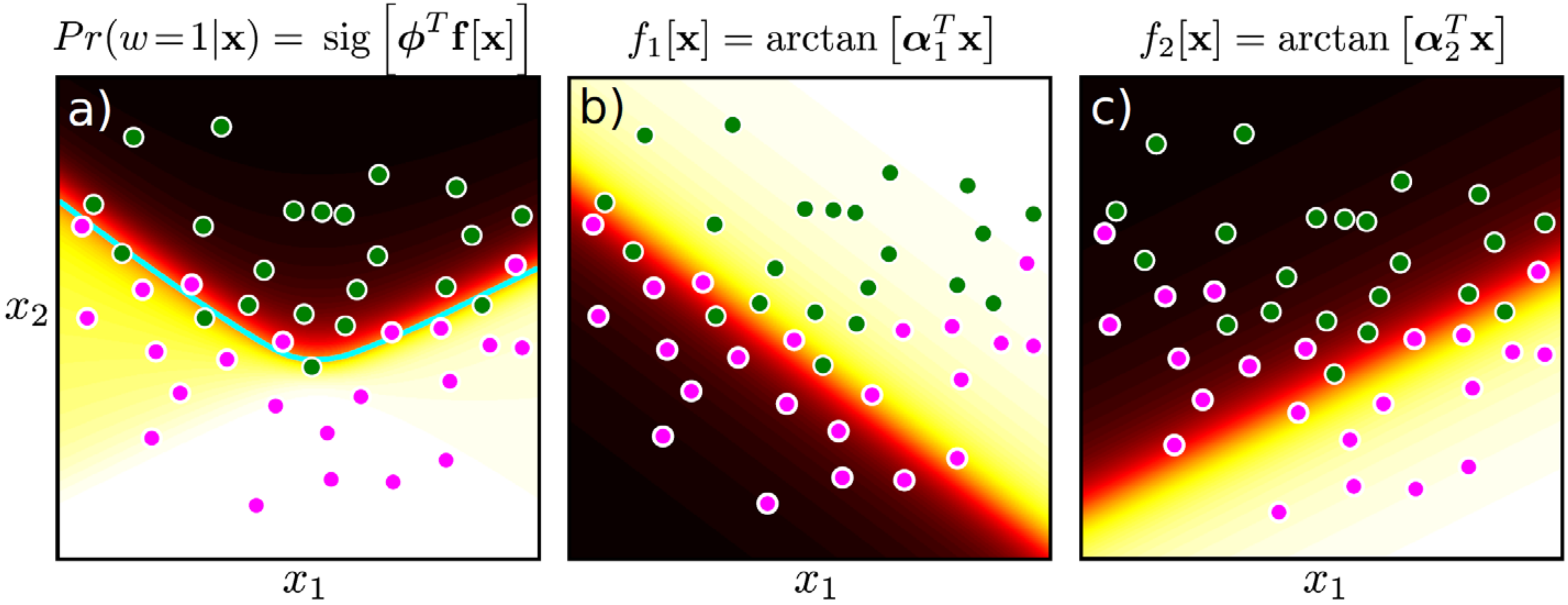
Weights after applying ML $\hat{\phi}$



Final activation

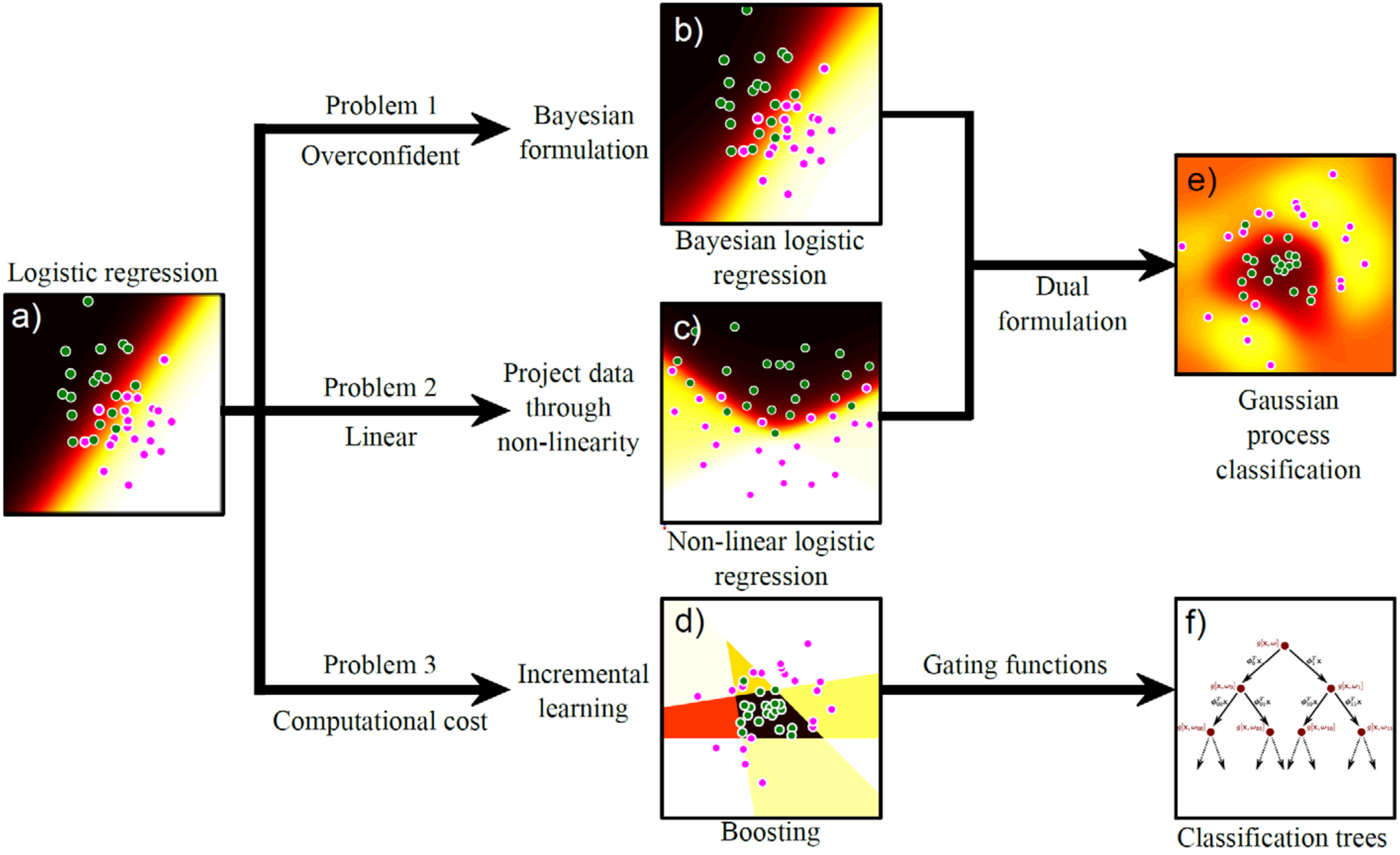
sig[Final activation]

Non-linear logistic regression in 2D



Structure

- Logistic regression
- Bayesian logistic regression
- Non-linear logistic regression
- Kernelization and Gaussian process classification
- Incremental fitting, boosting and trees
- Multi-class classification
- Random classification trees
- Non-probabilistic classification
- Applications



Logistic regression

Problem 1

Overconfident

Bayesian formulation

Bayesian logistic regression

Problem 2

Linear

Project data through non-linearity

Non-linear logistic regression

Problem 3

Computational cost

Incremental learning

Boosting

Dual formulation

Gaussian process classification

Gating functions

Classification trees

Dual Logistic Regression

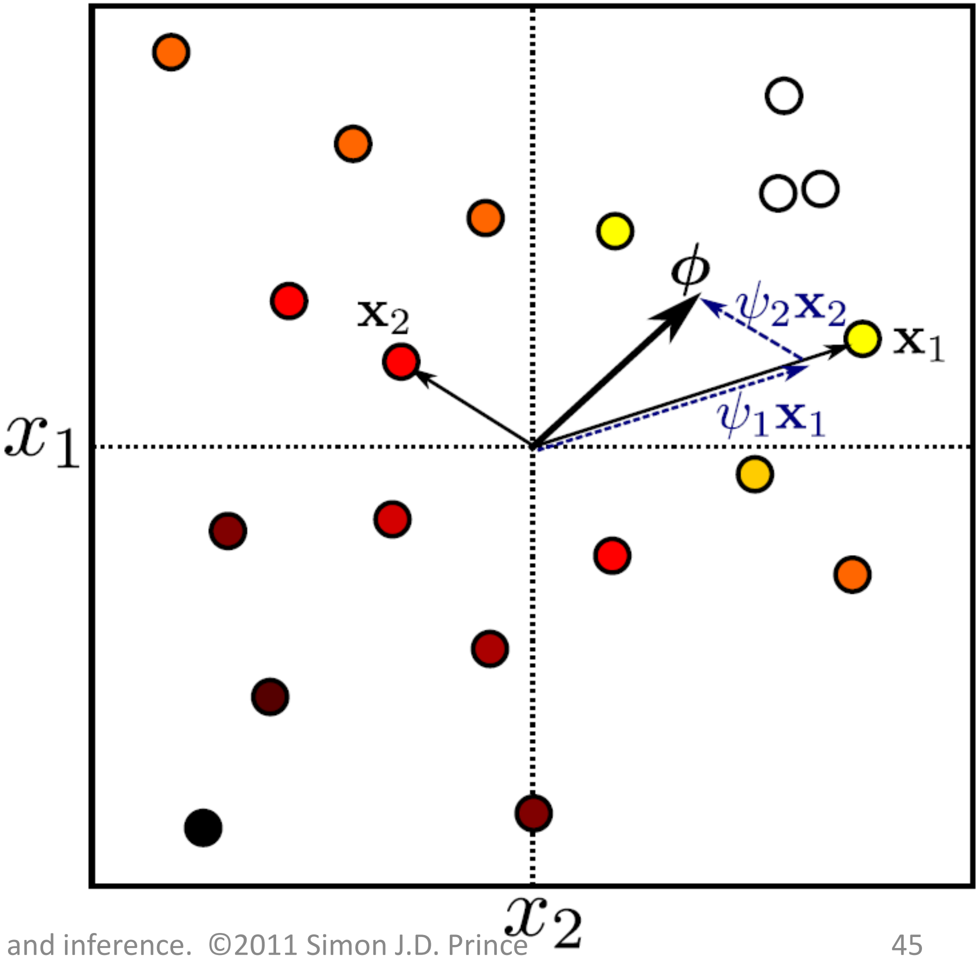
KEY IDEA:

Gradient Φ is just a vector in the data space

Can represent as a weighted sum of the data points

$$\phi = \mathbf{X}\psi$$

Now solve for Ψ. One parameter per training example.



Maximum Likelihood

Likelihood

$$Pr(\mathbf{w}|\mathbf{X},\boldsymbol{\psi}) = \prod_{i=1}^{I} \text{Bern}_{w_i}\left[\text{sig}[a_i]\right] = \prod_{i=1}^{I} \text{Bern}_{w_i}\left[\text{sig}[\boldsymbol{\psi}^T\mathbf{X}^T\mathbf{x}_i]\right]$$

Derivatives

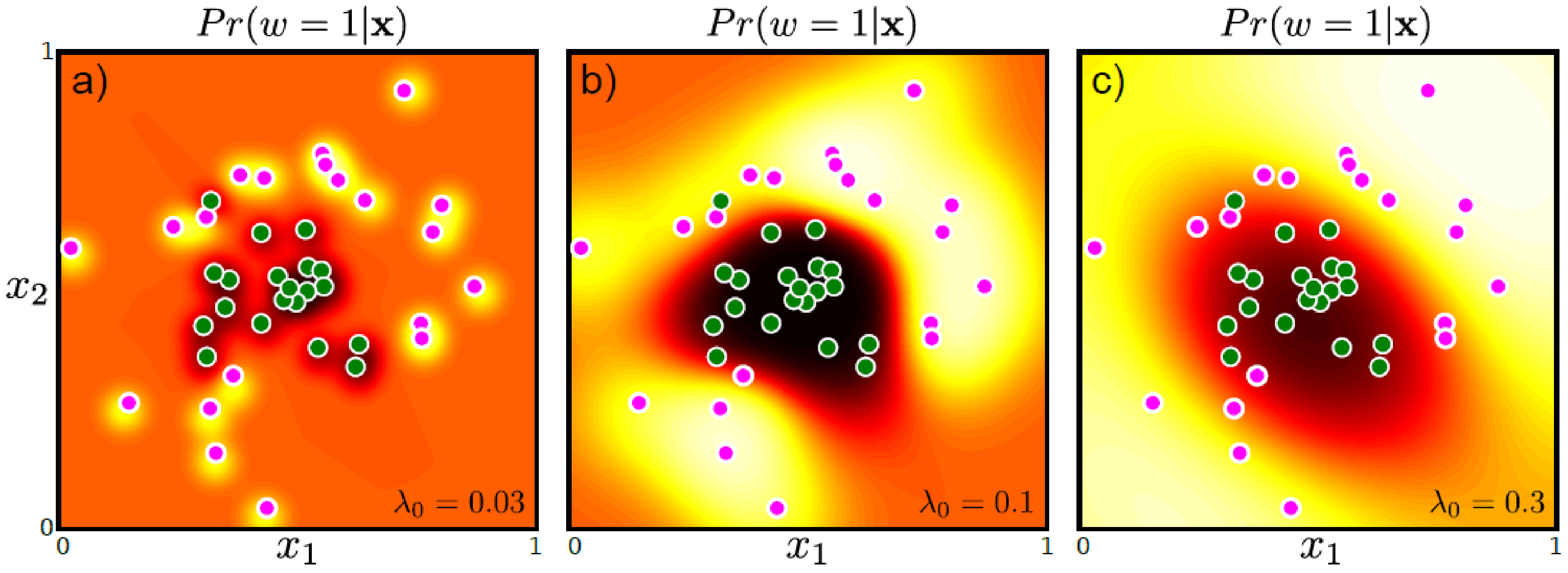
$$\frac{\partial L}{\partial \boldsymbol{\psi}} = -\sum_{i=1}^{I}\left(\text{sig}[a_i] - w_i\right)\mathbf{X}^T\mathbf{x}_i$$

$$\frac{\partial^2 L}{\partial \boldsymbol{\psi}^2} = -\sum_{i=1}^{I}\text{sig}[a_i]\left(1-\text{sig}[a_i]\right)\mathbf{X}^T\mathbf{x}_i\mathbf{x}_i^T\mathbf{X}$$

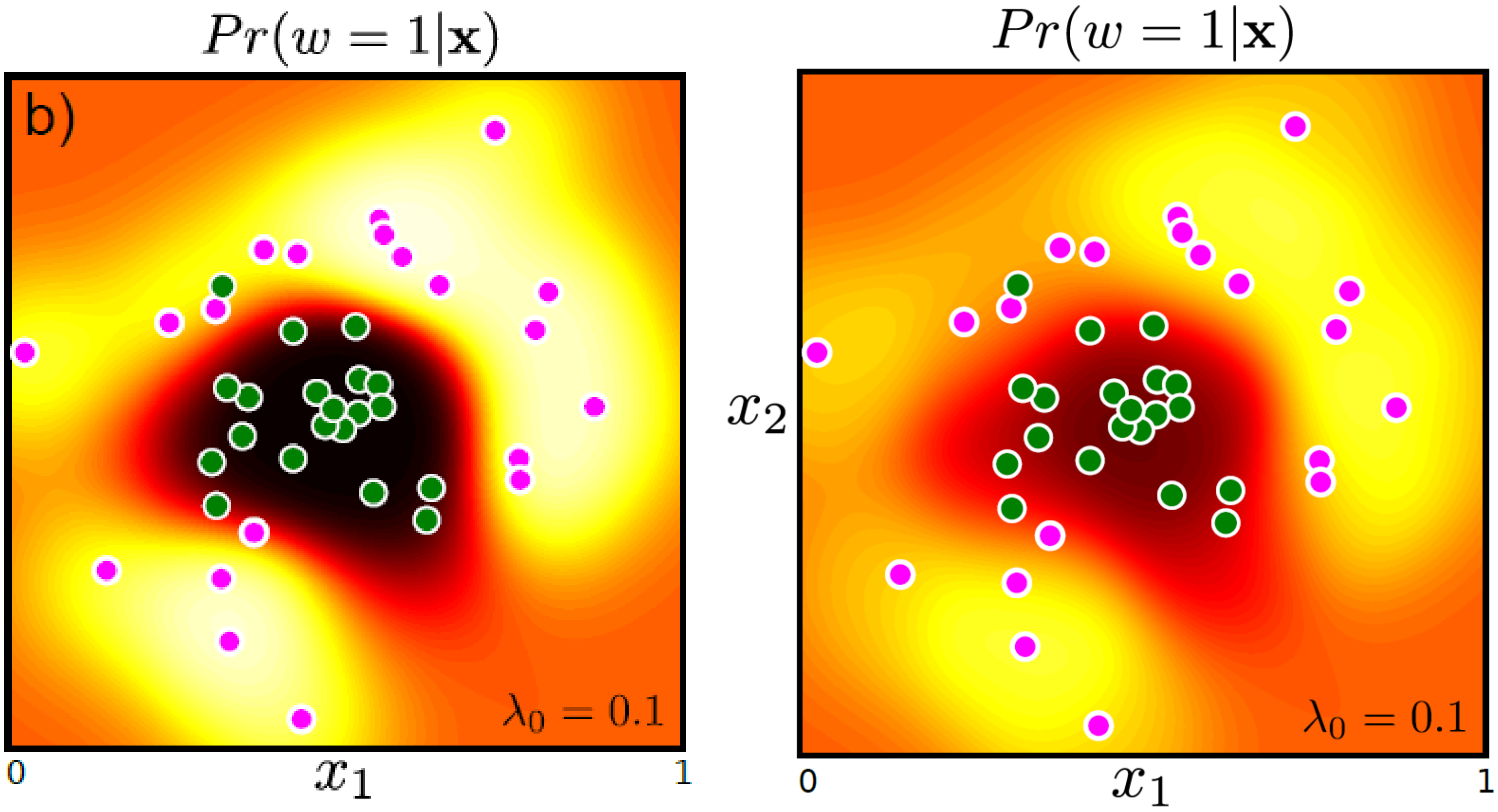
Depend only depend on inner products!

Kernel Logistic Regression

$$k[\mathbf{x}_i, \mathbf{x}_j] = \exp\left[-0.5\left(\frac{(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)}{\lambda_0^2}\right)\right]$$



ML vs. Bayesian



Bayesian case is known as Gaussian process classification

Relevance vector classification

Apply sparse prior to dual variables:

$$Pr(\boldsymbol{\psi}) = \prod_{i=1}^{I} \text{Stud}_{\boldsymbol{\psi}_i}\left[0, 1, \nu\right]$$

As before, write as marginalization of dual variables:

$$\begin{aligned} Pr(\boldsymbol{\psi}) &= \prod_{i=1}^{I} \int \text{Norm}_{\psi_i}\left[0, \frac{1}{h_i}\right] \text{Gam}_{h_i}\left[\frac{\nu}{2}, \frac{\nu}{2}\right] \, dh_i \\ &= \int \text{Norm}_{\boldsymbol{\psi}}[0, \mathbf{H}^{-1}] \prod_{d=1}^{D} \text{Gam}_{h_d}[\nu/2, \nu/2] \, d\mathbf{H}. \end{aligned}$$

Relevance vector classification

Apply sparse prior to dual variables:

$$Pr(\boldsymbol{\psi}) = \int \text{Norm}_{\boldsymbol{\psi}}[0, \mathbf{H}^{-1}] \prod_{d=1}^{D} \text{Gam}_{h_d}[\nu/2, \nu/2] \, d\mathbf{H}$$

Gives likelihood:

$$\begin{aligned} Pr(\mathbf{w}|\mathbf{X}) \\ &= \int Pr(\mathbf{w}|\mathbf{X}, \boldsymbol{\psi}) Pr(\boldsymbol{\psi}) \, d\boldsymbol{\psi} \\ &= \int\int \prod_{i=1}^{I} \text{Bern}_{w_i}\Big[\text{sig}[\boldsymbol{\psi}^T \mathbf{K}[\mathbf{X}, \mathbf{x}_i]\Big] \text{Norm}_{\boldsymbol{\psi}}[0, \mathbf{H}^{-1}] \prod_{d=1}^{D} \text{Gam}_{h_d}[\nu/2, \nu/2] \, d\mathbf{H} d\boldsymbol{\psi}. \end{aligned}$$

Relevance vector classification

$$Pr(\mathbf{w}|\mathbf{X})$$

$$= \iint \prod_{i=1}^{I} \text{Bern}_{w_i}\Big[\text{sig}[\boldsymbol{\psi}^T\mathbf{K}[\mathbf{X},\mathbf{x}_i]\Big]\text{Norm}_{\boldsymbol{\psi}}[0,\mathbf{H}^{-1}]\prod_{d=1}^{D}\text{Gam}_{h_d}[\nu/2,\nu/2]\; d\mathbf{H}d\boldsymbol{\psi}.$$

Use Laplace approximation result:

$$\begin{aligned}\int q(\boldsymbol{\psi})\,d\boldsymbol{\psi} &\approx q(\boldsymbol{\mu})\int \exp\left[-\frac{1}{2}(\boldsymbol{\psi}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\psi}-\boldsymbol{\mu})\right]\,d\boldsymbol{\psi}\\ &= q(\boldsymbol{\mu})(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}.\end{aligned}$$

giving:

$$Pr(\mathbf{w}|\mathbf{X}) \approx$$

$$\int\prod_{i=1}^{I}(2\pi)^{I/2}|\boldsymbol{\Sigma}|^{0.5}\text{Bern}_{w_i}[\text{sig}[\boldsymbol{\mu}^T\mathbf{K}[\mathbf{X},\mathbf{x}_i]]]\text{Norm}_{\boldsymbol{\mu}}[0,\mathbf{H}^{-1}]\text{Gam}_{h_i}\left[\frac{\nu}{2},\frac{\nu}{2}\right]\,d\mathbf{H}$$

Relevance vector classification

Previous result:

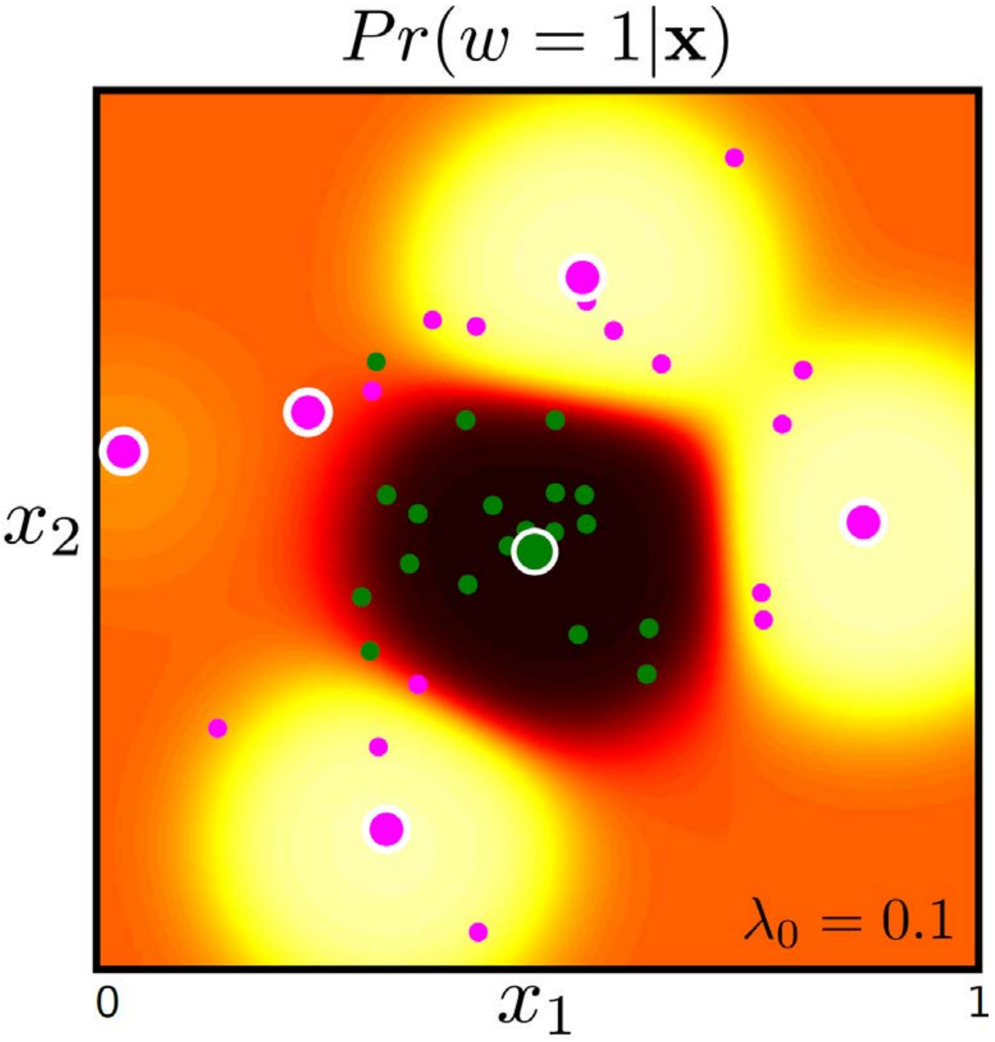
$$Pr(\mathbf{w}|\mathbf{X}) \approx \int \prod_{i=1}^{I} (2\pi)^{I/2} |\boldsymbol{\Sigma}|^{0.5} \text{Bern}_{w_i}[\text{sig}[\boldsymbol{\mu}^T \mathbf{K}[\mathbf{X}, \mathbf{x}_i]]\text{Norm}_{\boldsymbol{\mu}}[0, \mathbf{H}^{-1}]\text{Gam}_{h_i}\left[\frac{\nu}{2}, \frac{\nu}{2}\right] d\mathbf{H}$$

Second approximation:

$$Pr(\mathbf{w}|\mathbf{X}) \approx \max_{\mathbf{H}} \left[\prod_{i=1}^{I} (2\pi)^{I/2} |\boldsymbol{\Sigma}|^{0.5} \text{Bern}_{w_i}[\text{sig}[\boldsymbol{\mu}^T \mathbf{K}[\mathbf{X}, \mathbf{x}_i]]\text{Norm}_{\boldsymbol{\mu}}[0, \mathbf{H}^{-1}]\text{Gam}_{h_i}\left[\frac{\nu}{2}, \frac{\nu}{2}\right]\right]$$

To solve, alternately update hidden variables in **H** and mean and variance of Laplace approximation.

Relevance vector classification



Results:

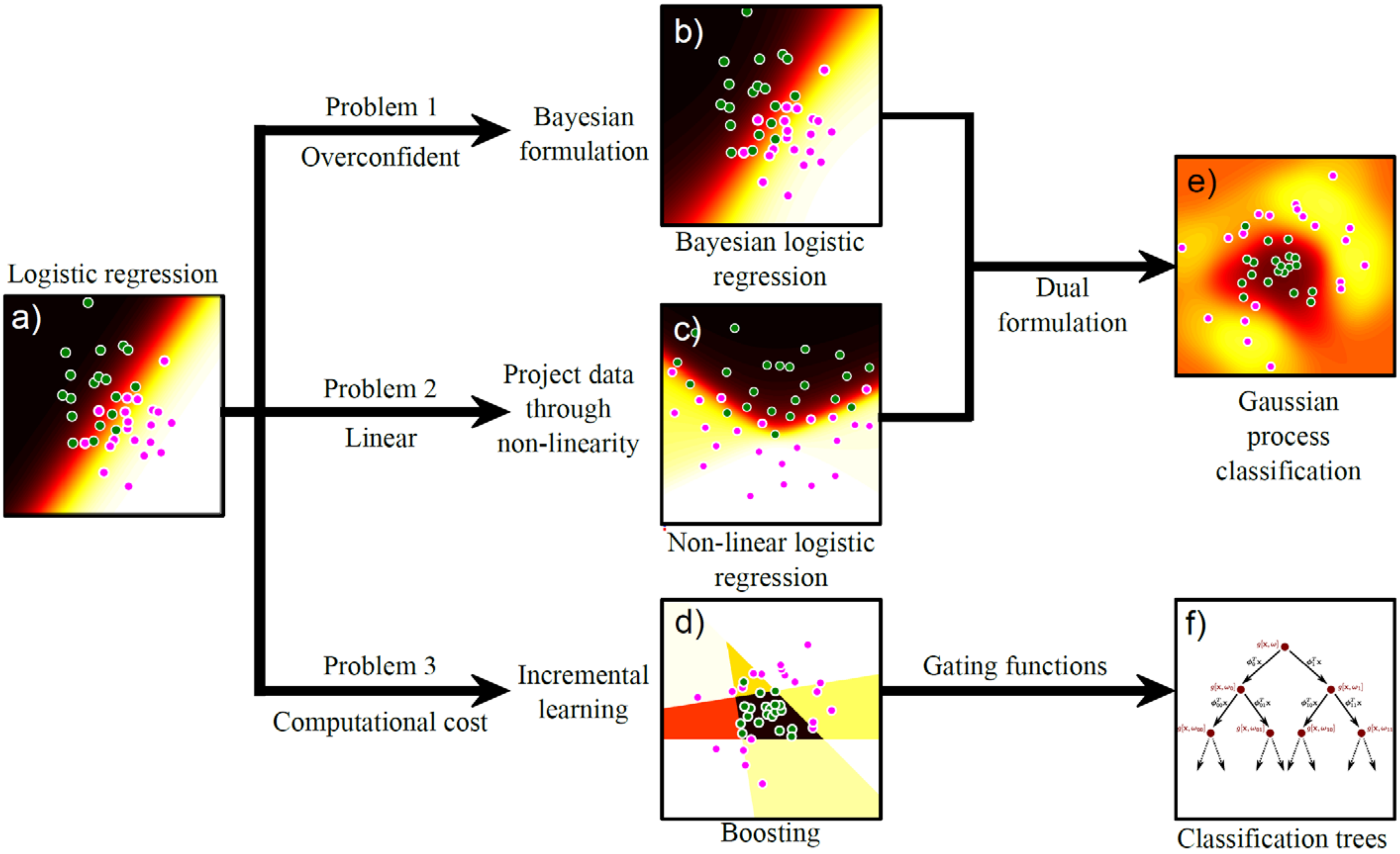
Most hidden variables increase to larger values

This means prior over dual variable is very tight around zero

The final solution only depends on a very small number of examples – efficient

Structure

- Logistic regression
- Bayesian logistic regression
- Non-linear logistic regression
- Kernelization and Gaussian process classification
- Incremental fitting & boosting
- Multi-class classification
- Random classification trees
- Non-probabilistic classification
- Applications



Logistic regression

Problem 1

Overconfident

Bayesian formulation

Bayesian logistic regression

Problem 2

Linear

Project data through non-linearity

Non-linear logistic regression

Problem 3

Computational cost

Incremental learning

Boosting

Dual formulation

Gaussian process classification

Gating functions

Classification trees

Incremental Fitting

Previously wrote:

$$a_i = \boldsymbol{\phi}^T \mathbf{z}_i = \boldsymbol{\phi}^T \mathbf{f}[\mathbf{x}_i]$$

Now write:

$$a_i = \phi_0 + \sum_{k=1}^{K} \phi_k f[\mathbf{x}_i, \boldsymbol{\xi}_k]$$

- Arc tan functions, $\boldsymbol{\xi} = \{\boldsymbol{\alpha}\}$

$$f[\mathbf{x}, \boldsymbol{\xi}] = \arctan[\boldsymbol{\alpha}^T \mathbf{x}]$$

- Radial basis functions, $\boldsymbol{\xi} = \{\boldsymbol{\alpha}, \lambda_0\}$

$$f[\mathbf{x}, \boldsymbol{\xi}] = \exp\left[-\frac{(\mathbf{x}-\boldsymbol{\alpha})^T(\mathbf{x}-\boldsymbol{\alpha})}{\lambda_0^2}\right]$$

Incremental Fitting

KEY IDEA: Greedily add terms one at a time.

STAGE 1: Fit $\phi_0,\ \phi_1,\ \xi_1$

$$a_i = \phi_0 + \phi_1 f[\mathbf{x}_i, \boldsymbol{\xi}_1]$$

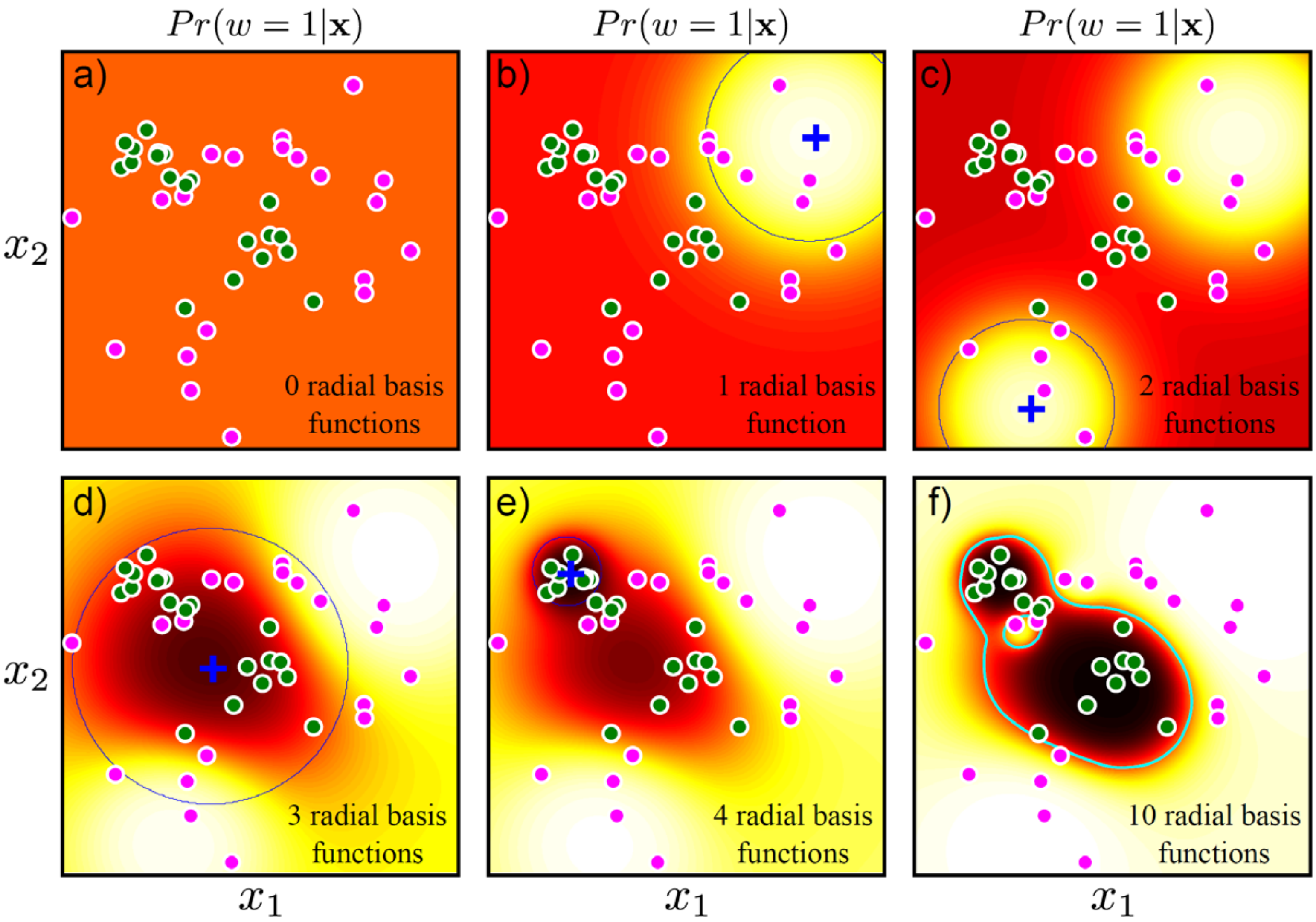
STAGE 2: Fit $\phi_0,\ \phi_2,\ \xi_2$

$$a_i = \boxed{\phi_0} + \phi_1 f[\mathbf{x}_i, \boldsymbol{\xi}_2] + \boxed{\phi_2 f[\mathbf{x}_i, \boldsymbol{\xi}_2]}$$

STAGE K: Fit $\phi_0,\ \phi_k,\ \xi_k$

$$a_i = \phi_0 + \sum_{k=1}^{K} \phi_k f[\mathbf{x}_i, \boldsymbol{\xi}_k]$$

Incremental Fitting



Derivative

It is worth considering the form of the derivative in the context of the incremental fitting procedure

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = -\sum_{i=1}^{I}(w_i - \text{sig}[a_i])\frac{\partial a_i}{\partial \boldsymbol{\theta}}$$

Actual label

Predicted Label

Points contribute to derivative more if they are still misclassified: the later classifiers become increasingly specialized to the difficult examples.

Boosting

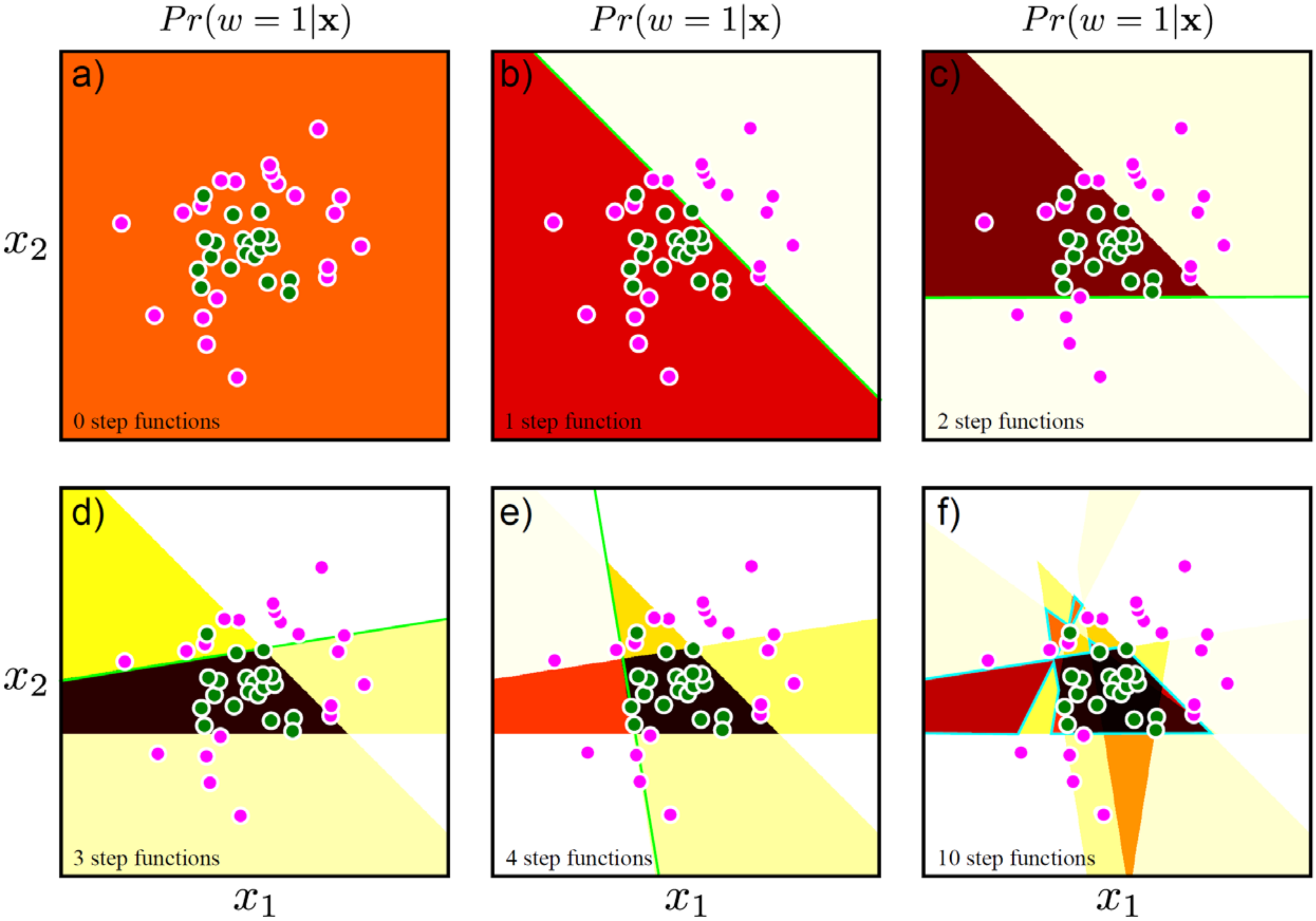
Incremental fitting with step functions

$$a_i = \phi_0 + \sum_{k=1}^{K} \phi_k \text{Heaviside}[\boldsymbol{\alpha}_k^T \mathbf{x}]$$

Each step function is called a ``weak classifier``

Can’t take derivative w.r.t α so have to just use exhaustive search

Boosting


$Pr(w=1|\mathbf{x})$
$Pr(w=1|\mathbf{x})$
$Pr(w=1|\mathbf{x})$
a)
0 step functions
b)
1 step function
c)
2 step functions
x_2
d)
3 step functions
e)
4 step functions
f)
10 step functions
x_2
x_1
x_1
x_1

Branching Logistic Regression

A different way to make non-linear classifiers

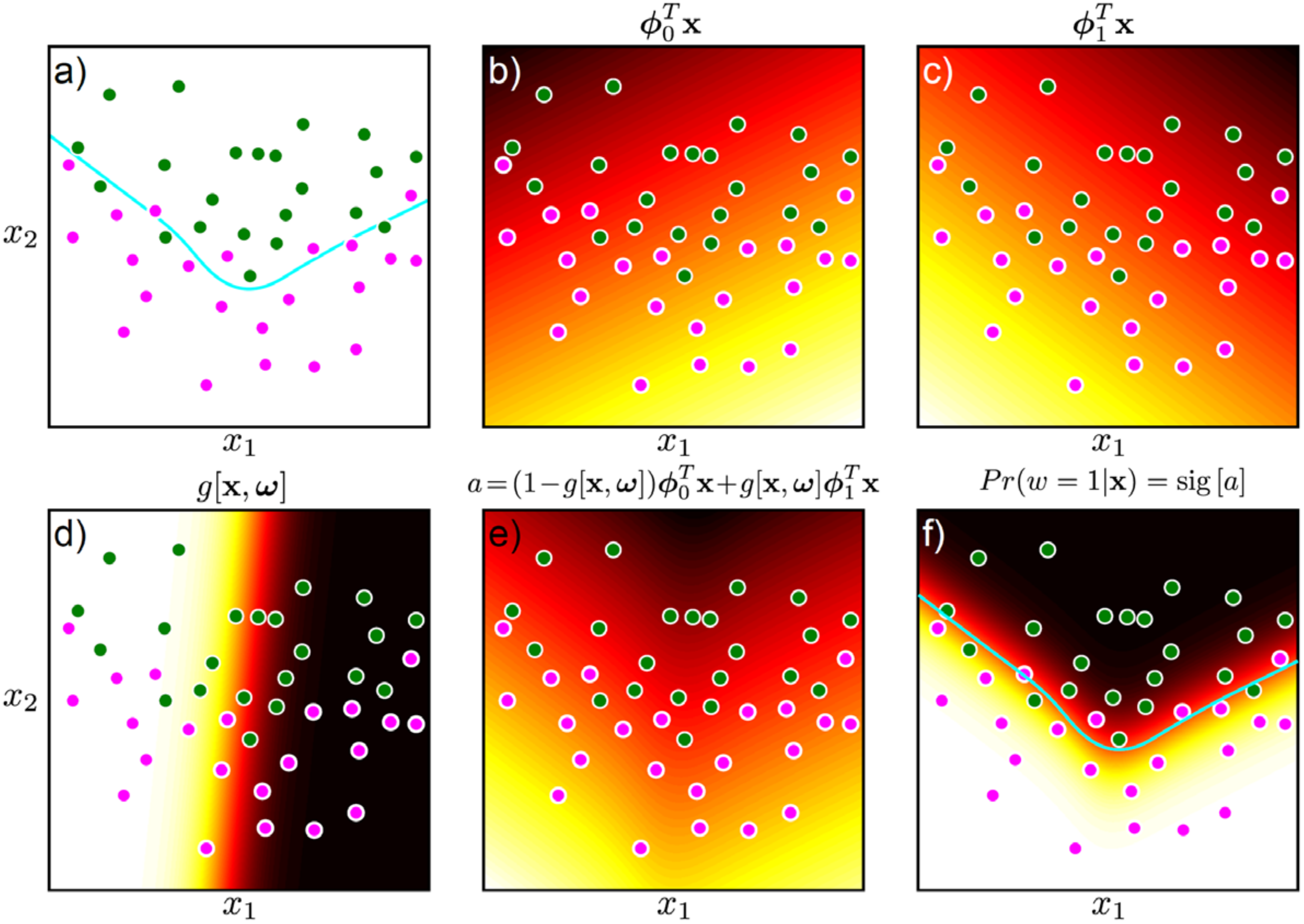
New activation

$$a_i = (1 - g[\mathbf{x}_i, \boldsymbol{\omega}])\boldsymbol{\phi}_0^T\mathbf{x}_i + g[\mathbf{x}_i, \boldsymbol{\omega}]\boldsymbol{\phi}_1^T\mathbf{x}_i$$

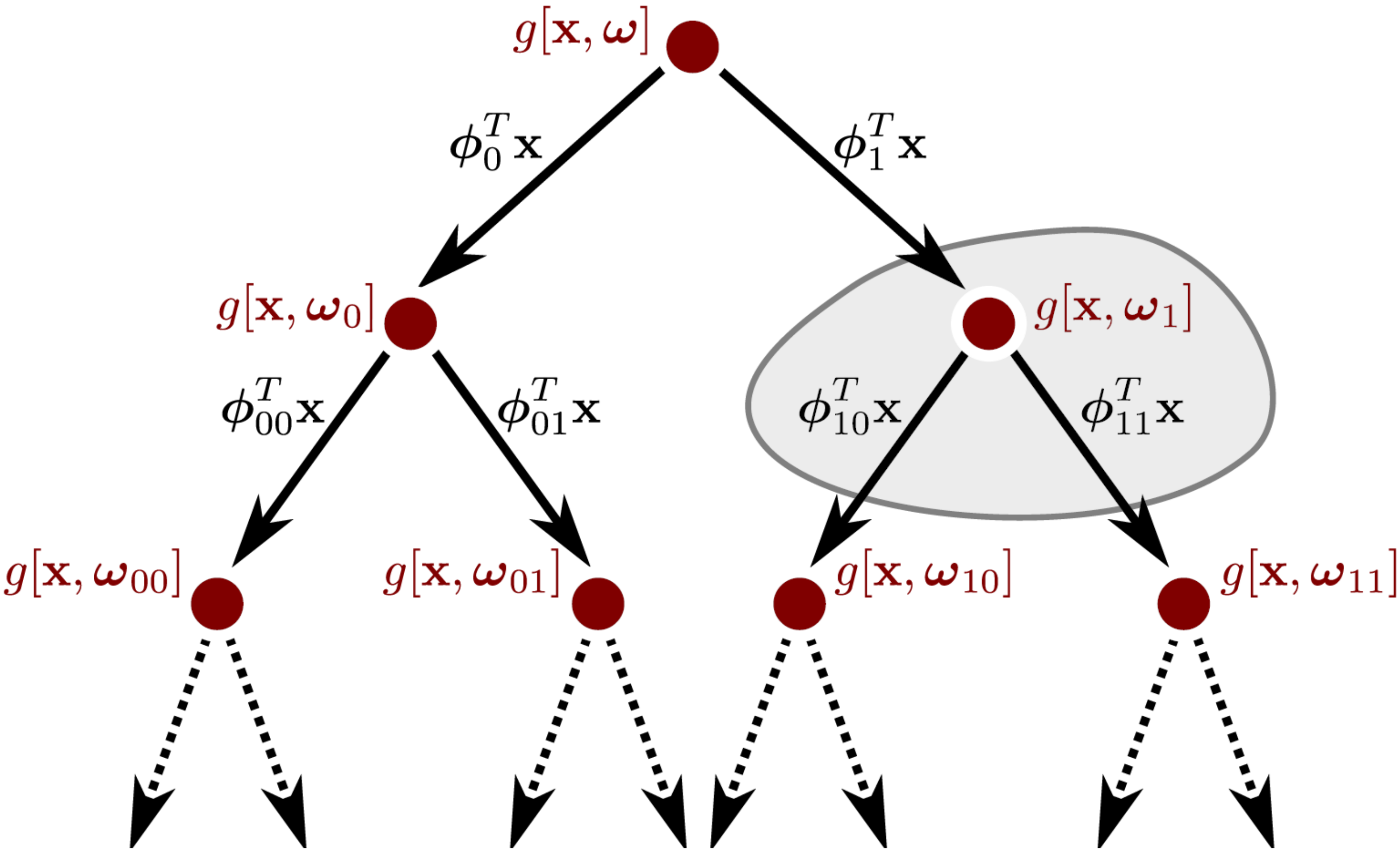
The term $g[\bullet, \bullet]$ is a gating function.

- Returns a number between 0 and 1
- If 0, then we get one logistic regression model
- If 1, then get a different logistic regression model

Branching Logistic Regression



Logistic Classification Trees


$g[\mathbf{x}, \boldsymbol{\omega}]$
$\phi_0^T\mathbf{x}$
$\phi_1^T\mathbf{x}$
$g[\mathbf{x}, \boldsymbol{\omega}_0]$
$g[\mathbf{x}, \boldsymbol{\omega}_1]$
$\phi_{00}^T\mathbf{x}$
$\phi_{01}^T\mathbf{x}$
$\phi_{10}^T\mathbf{x}$
$\phi_{11}^T\mathbf{x}$
$g[\mathbf{x}, \boldsymbol{\omega}_{00}]$
$g[\mathbf{x}, \boldsymbol{\omega}_{01}]$
$g[\mathbf{x}, \boldsymbol{\omega}_{10}]$
$g[\mathbf{x}, \boldsymbol{\omega}_{11}]$

Structure

- Logistic regression
- Bayesian logistic regression
- Non-linear logistic regression
- Kernelization and Gaussian process classification
- Incremental fitting, boosting and trees
- Multi-class classification
- Random classification trees
- Non-probabilistic classification
- Applications

Multiclass Logistic Regression

For multiclass recognition, choose distribution over w and make the parameters of this a function of **x**.

$$Pr(w|\mathbf{x}) = \text{Cat}_w[\boldsymbol{\lambda}[\mathbf{x}]]$$

Softmax function maps real activations $\{a_n\}$ to numbers between zero and one that sum to one

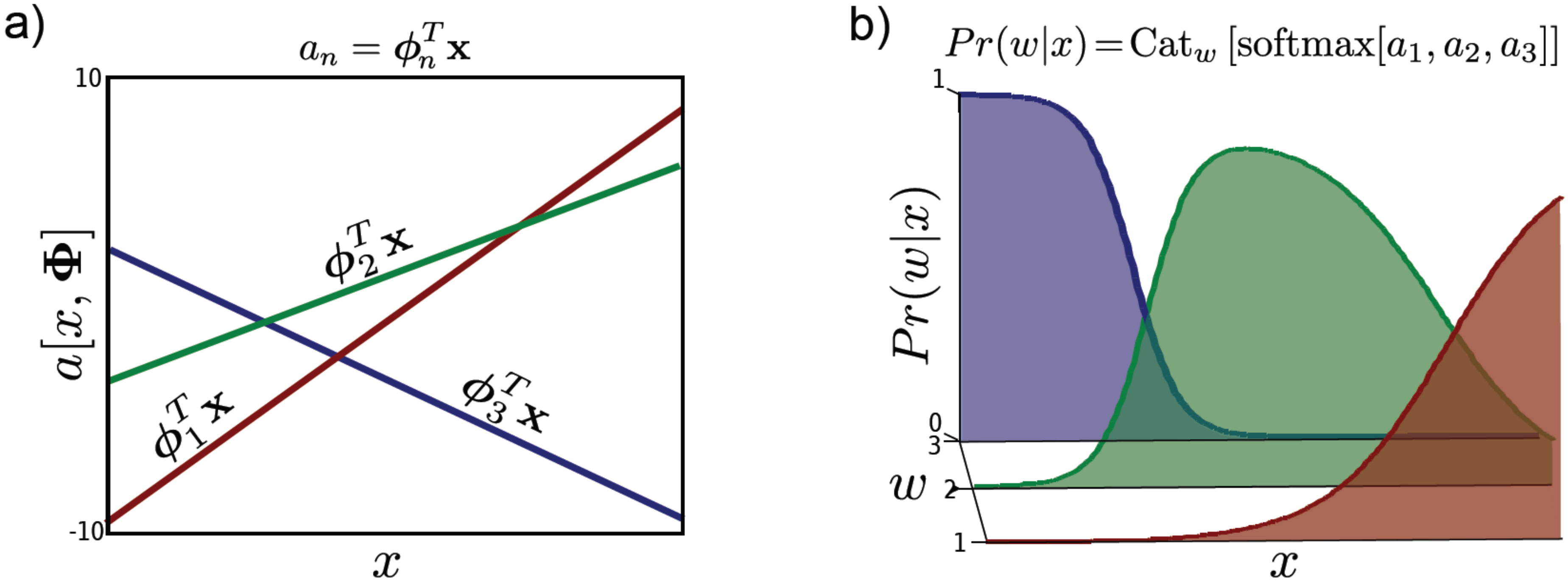
$$\lambda_n = \text{softmax}_n[a_1, a_2 \ldots a_N] = \frac{\exp[a_n]}{\sum_{m=1}^{N} \exp[a_m]}$$

Parameters are vectors $\{\phi_n\}$

$$a_n = \boldsymbol{\phi}_n^T \mathbf{x}$$

Multiclass Logistic Regression

Softmax function maps activations which can take any value to parameters of categorical distribution between 0 and 1



Multiclass Logistic Regression

To learn model, maximize log likelihood

$$L = \sum_{i=1}^{I} \log\left[Pr(w_i|\mathbf{x}_i)\right]$$

No closed from solution, learn with non-linear optimization

$$\begin{aligned}\frac{\partial L}{\partial \boldsymbol{\phi}_n} &= -\sum_{i=1}^{I}(y_{in} - \delta[w_i - n])\mathbf{x}_i\\ \frac{\partial^2 L}{\partial \boldsymbol{\phi}_m \boldsymbol{\phi}_n} &= -\sum_{i=1}^{I} y_{im}(\delta[m-n] - y_{in})\mathbf{x}_i\mathbf{x}_i^T\end{aligned}$$

where

$$y_{in} = Pr(w_i = n|\mathbf{x}_i) = \text{softmax}_n[a_{i1}, a_{i2} \ldots a_{iN}]$$

Structure

- Logistic regression
- Bayesian logistic regression
- Non-linear logistic regression
- Kernelization and Gaussian process classification
- Incremental fitting, boosting and trees
- Multi-class classification
- Random classification trees
- Non-probabilistic classification
- Applications

Random classification tree

Key idea:

- Binary tree
- Randomly chosen function at each split
- Choose threshold t to maximize log probability

$$\begin{aligned} L &= \sum_{i=1}^{I}(1-\text{heaviside}[q[\mathbf{x}_i]-\tau])\log\left[\text{Cat}_{w_i}\left[\boldsymbol{\lambda}^{[l]}\right]\right] \\ &\qquad\qquad\qquad\qquad +\text{heaviside}[q[\mathbf{x}_i]-\tau]\log\left[\text{Cat}_{w_i}\left[\boldsymbol{\lambda}^{[r]}\right]\right] \end{aligned}$$

For given threshold, can compute parameters in closed form

$$\begin{aligned} \lambda_k^{[l]} &= \frac{\sum_{i=1}^{I}\delta[w_i-k](1-\text{heaviside}[q[\mathbf{x}_i]-\tau])}{\sum_{i=1}^{I}(1-\text{heaviside}[q[\mathbf{x}_i]-\tau])} \\ \lambda_k^{[r]} &= \frac{\sum_{i=1}^{I}\delta[w_i-k](\text{heaviside}[q[\mathbf{x}_i]-\tau])}{\sum_{i=1}^{I}(\text{heaviside}[q[\mathbf{x}_i]-\tau])}. \end{aligned}$$

Random classification tree

Related models:

Fern:

- A tree where all of the functions at a level are the same
- Thresholds per level may be same or different
- Very efficient to implement

Forest

- Collection of trees
- Average results to get more robust answer
- Similar to `Bayesian’ approach – average of models with different parameters

Structure

- Logistic regression
- Bayesian logistic regression
- Non-linear logistic regression
- Kernelization and Gaussian process classification
- Incremental fitting, boosting and trees
- Multi-class classification
- Random classification trees
- Non-probabilistic classification
- Applications

Non-probabilistic classifiers

Most people use non-probabilistic classification methods such as neural networks, adaboost, support vector machines. This is largely for historical reasons

Probabilistic approaches:

- No serious disadvantages
- Naturally produce estimates of uncertainty
- Easily extensible to multi-class case
- Easily related to each other

Non-probabilistic classifiers

Multi-layer perceptron (neural network)

- Non-linear logistic regression with sigmoid functions
- Learning known as back propagation
- Transformed variable z is hidden layer

Adaboost

- Very closely related to logitboost
- Performance very similar

Support vector machines

- Similar to relevance vector classification but objective fn is convex
- No certainty
- Not easily extended to multi-class
- Produces solutions that are less sparse
- More restrictions on kernel function

Structure

- Logistic regression
- Bayesian logistic regression
- Non-linear logistic regression
- Kernelization and Gaussian process classification
- Incremental fitting, boosting and trees
- Multi-class classification
- Random classification trees
- Non-probabilistic classification
- Applications

Gender Classification

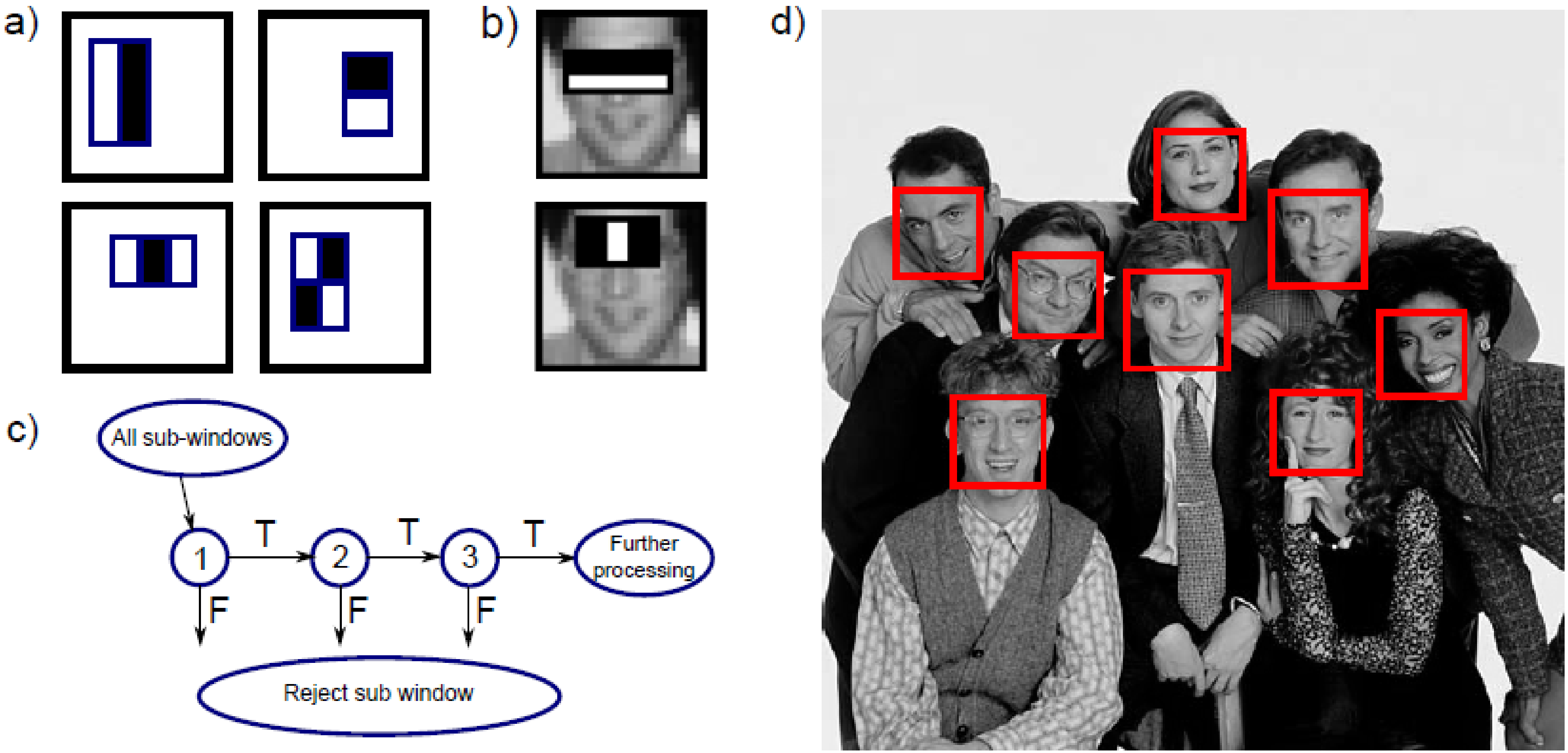
Incremental logistic regression

$$Pr(w_i|\mathbf{x}_i) = \text{Bern}_{w_i}\left[\frac{1}{1+\exp[-\phi_0+\sum_{k=1}^{K}\phi_k \text{f}[\mathbf{x}_i,\boldsymbol{\xi}_k]]}\right]$$

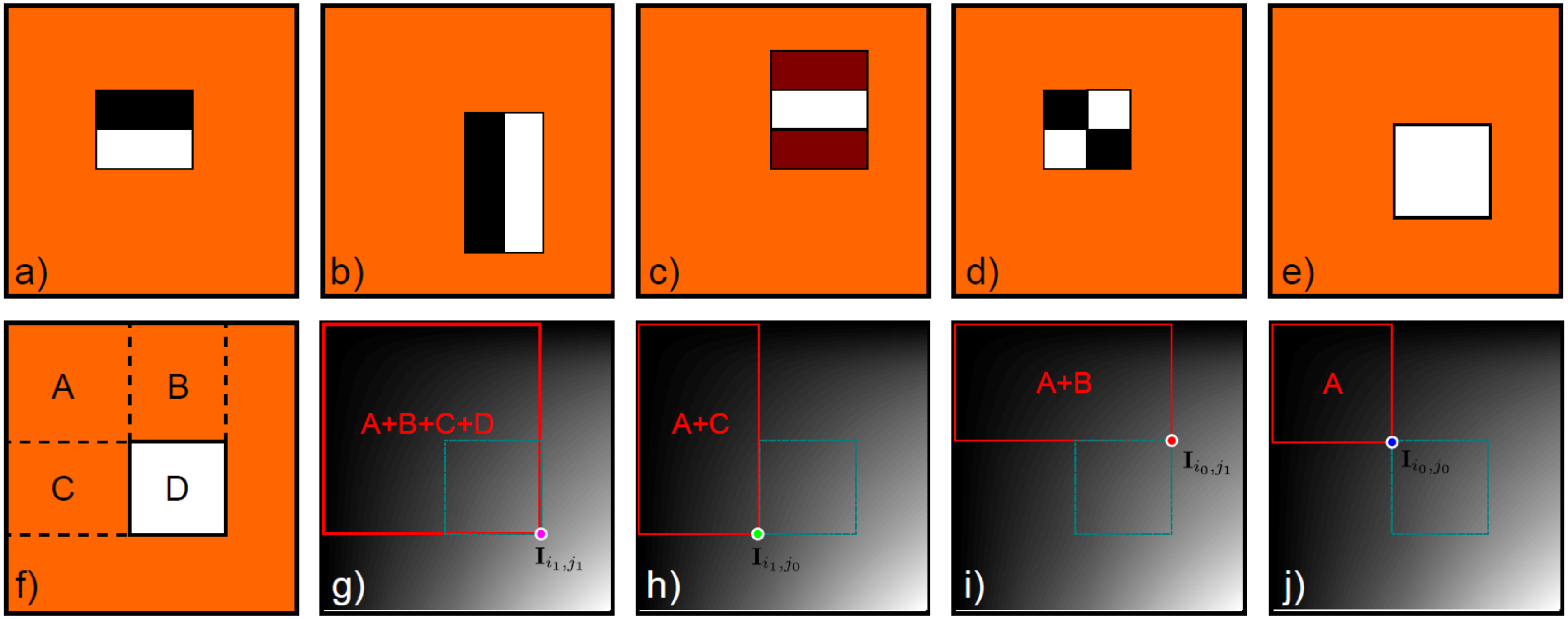
300 arc tan basis functions: $\text{f}[\mathbf{x}_i,\boldsymbol{\xi}_k] = \arctan[\boldsymbol{\xi}_k^T\mathbf{x}_i]$

Results: 87.5% (humans=95%)

Fast Face Detection (Viola and Jones 2001)

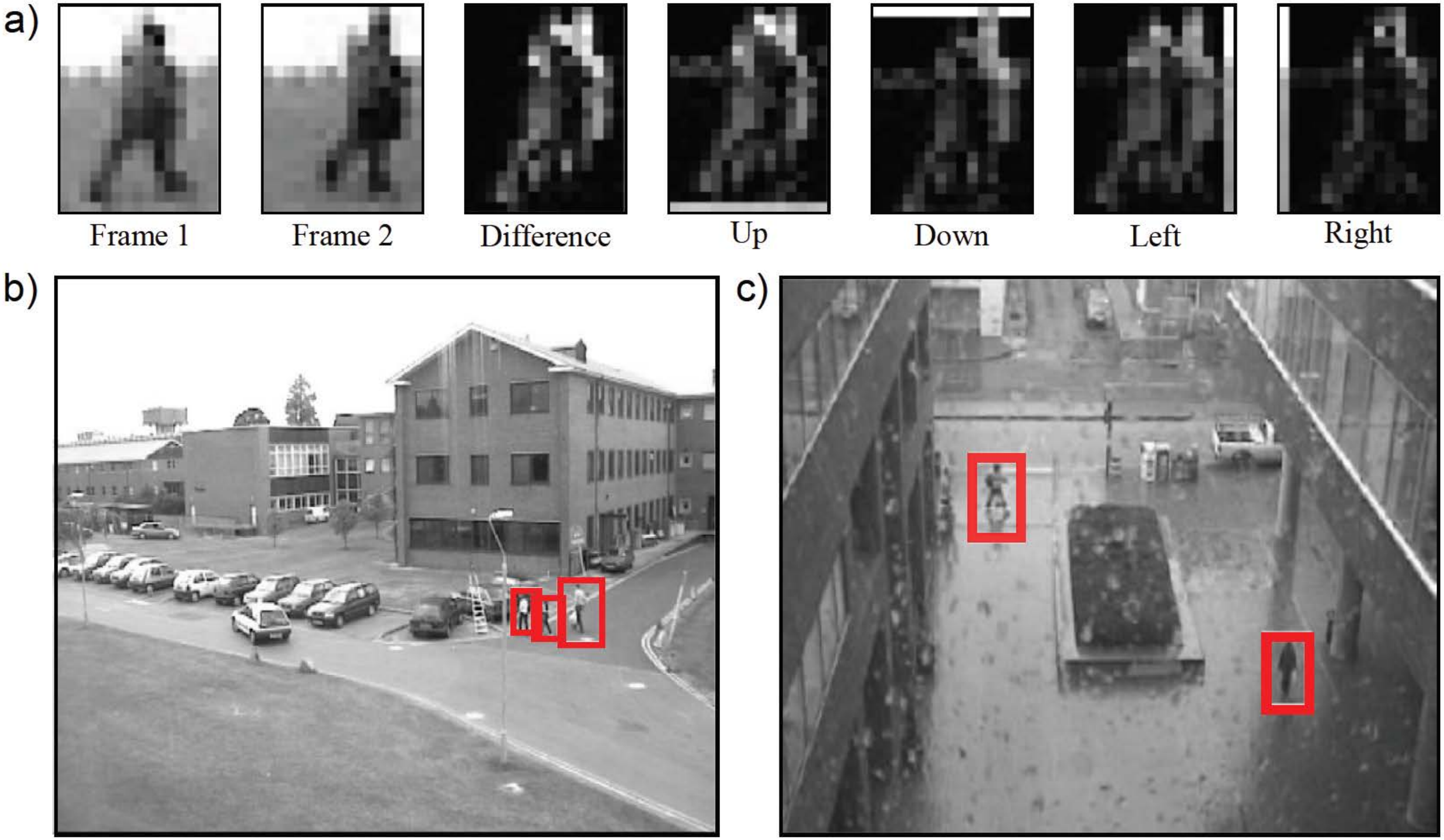


Computing Haar Features



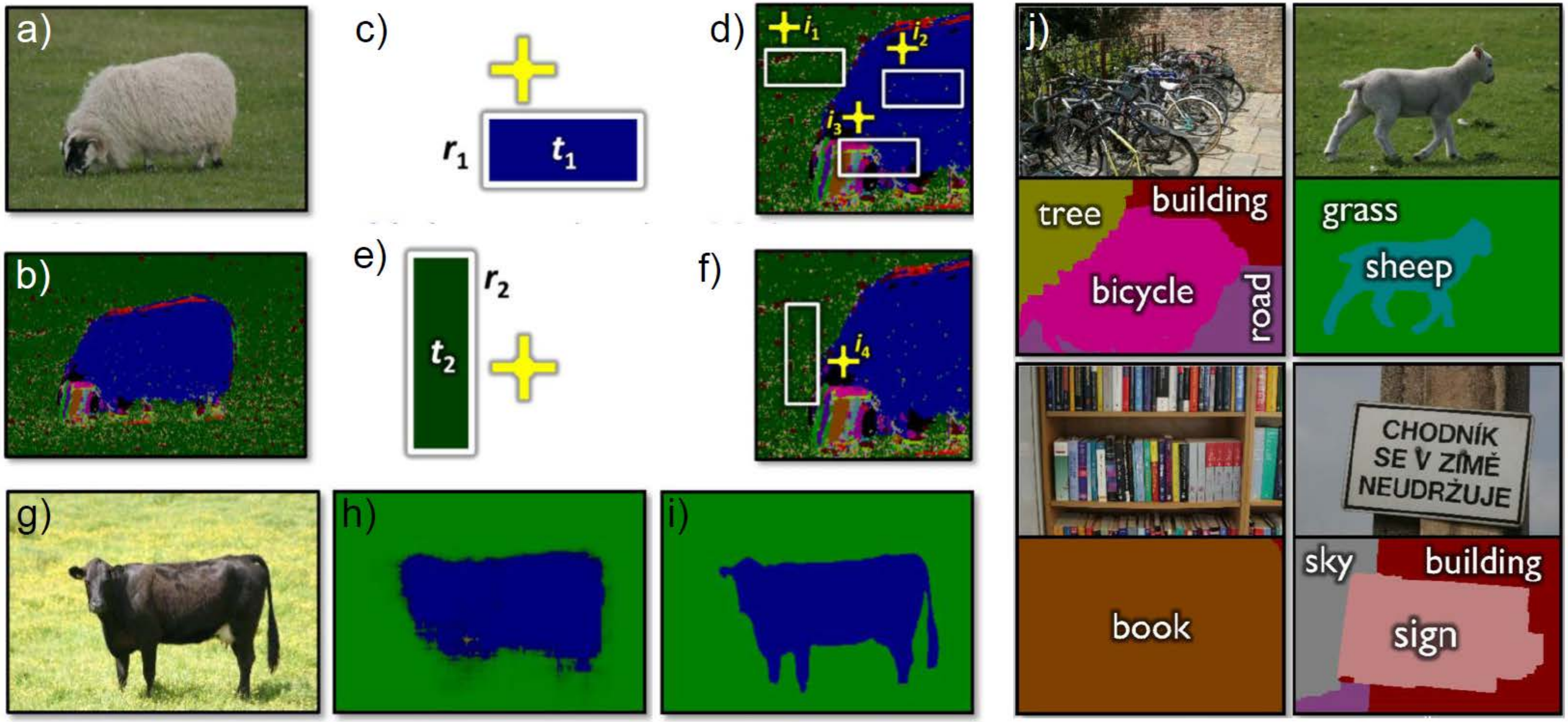
(See “Integral Images” or summed-area tables)

Pedestrian Detection



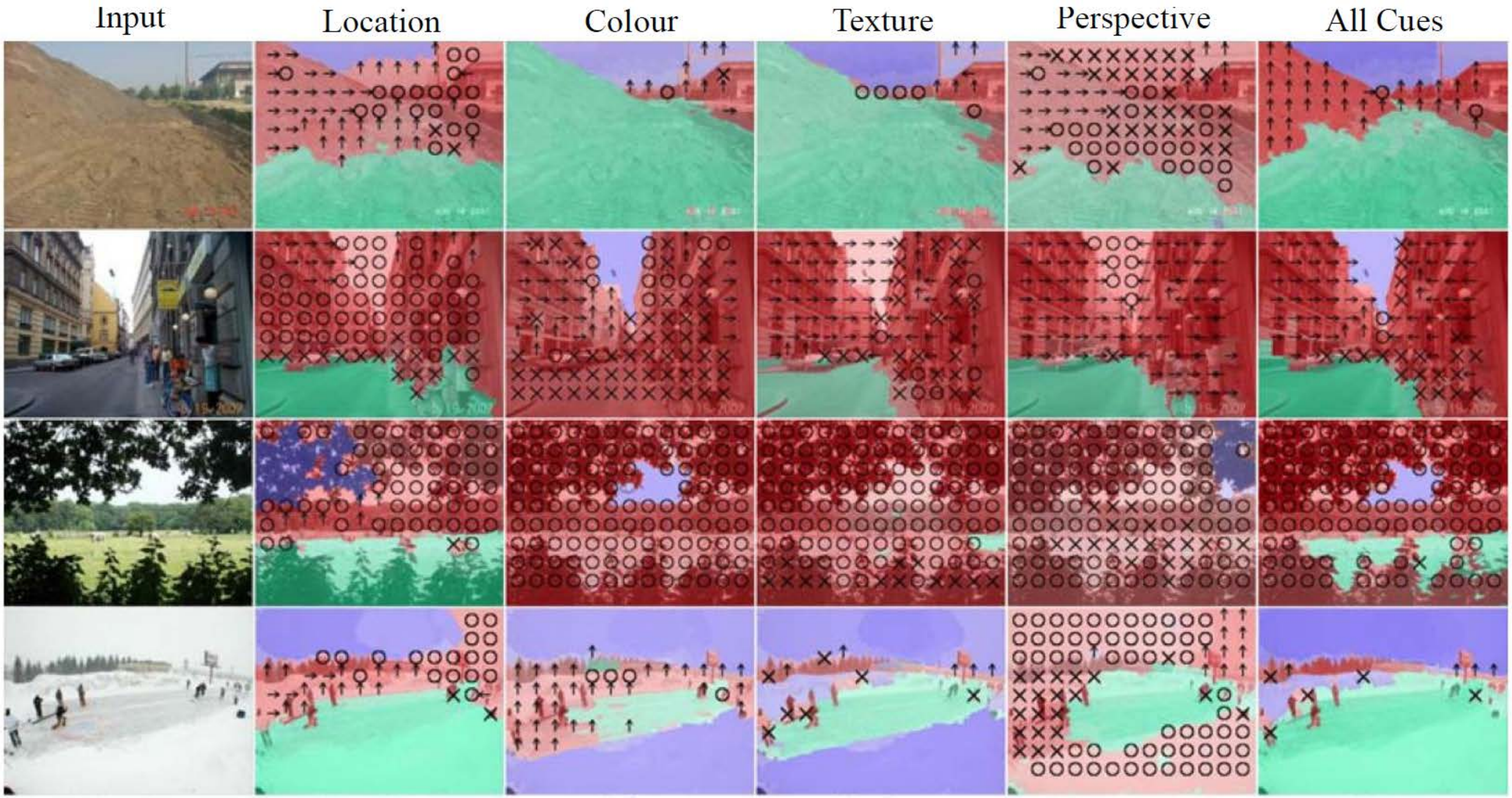
Adapted from Viola *et al.* (2005)

Semantic segmentation



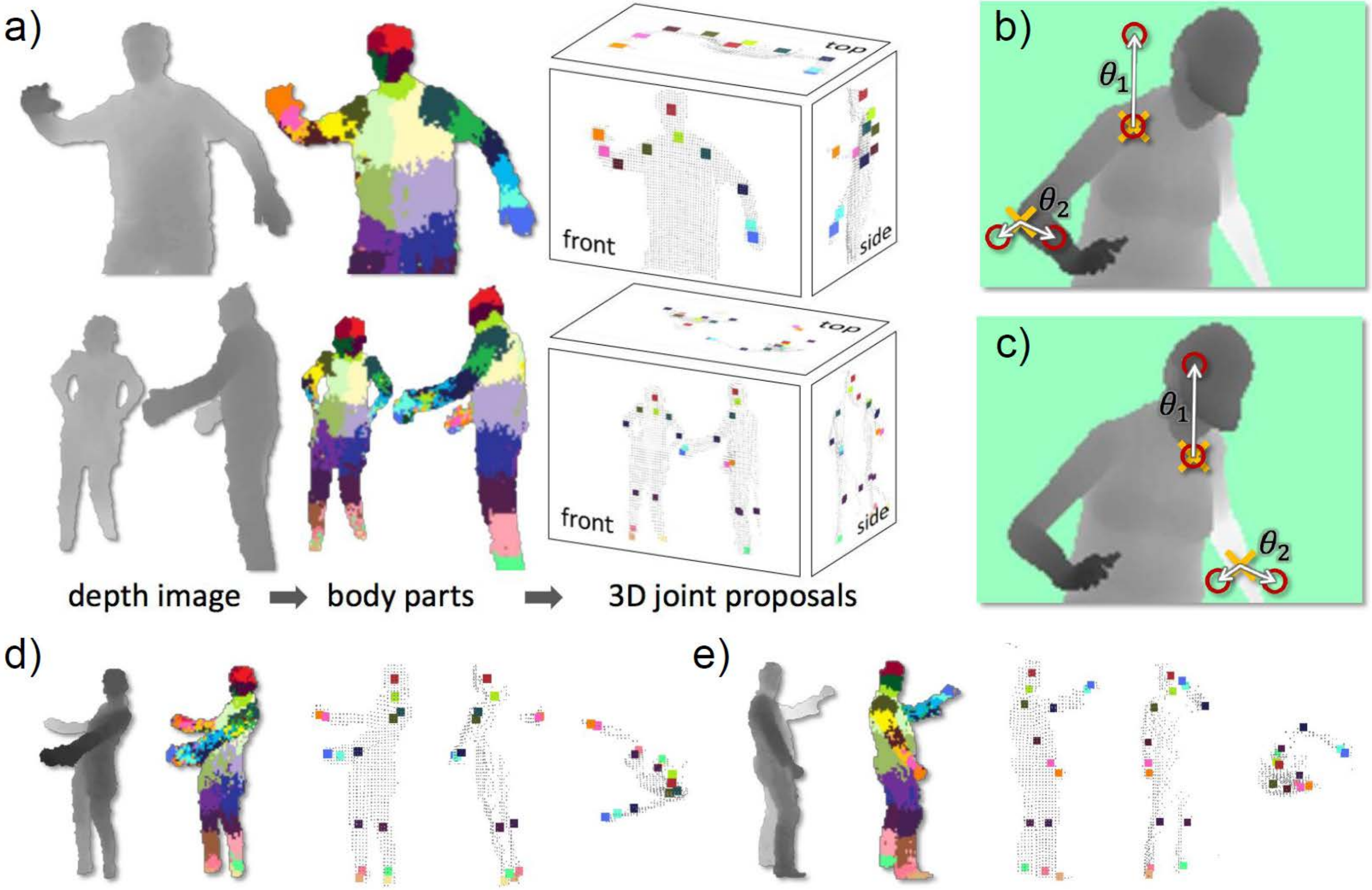
Shotton *et al.* (2009)

Recovering surface layout



Hoiem *et al.* (2007) ©2007

Recovering body pose



Adapted from Shotton *et al.* (2011)