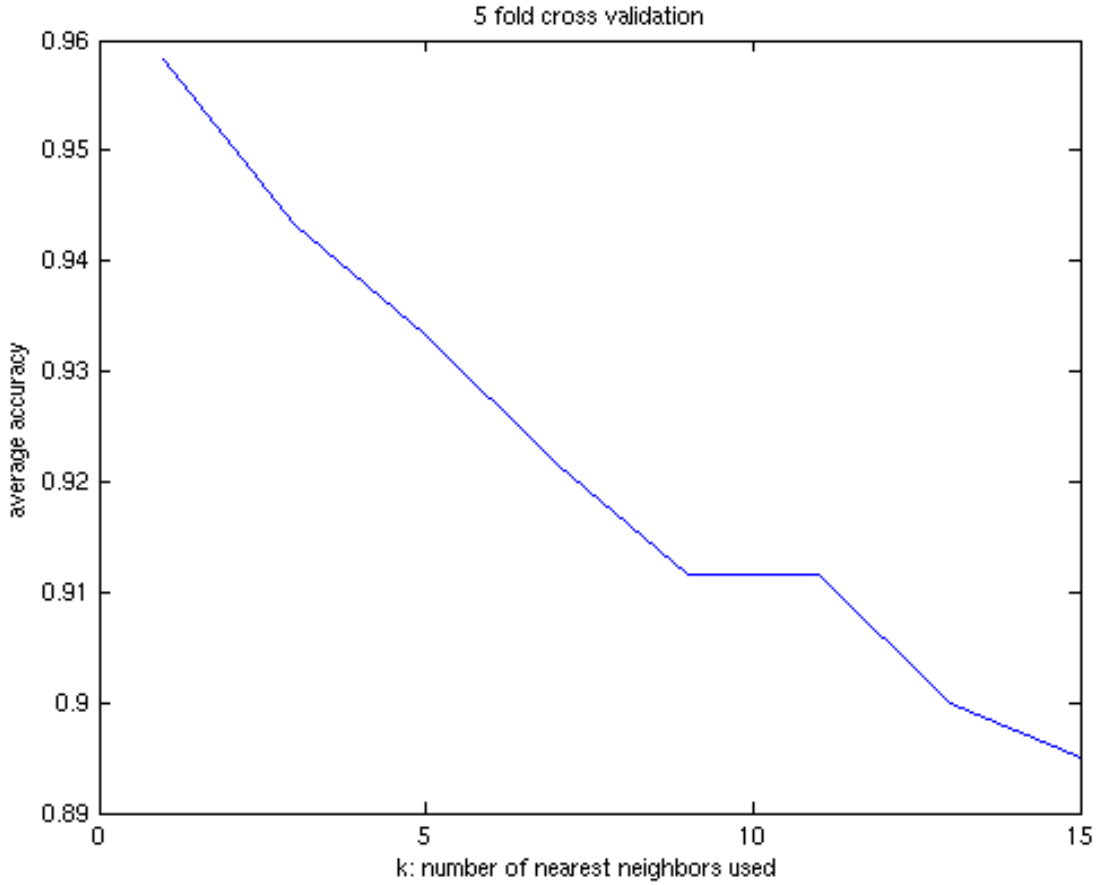
Introduction to Machine Learning

Brown University CSCI 1950-F, Spring 2012
Prof. Erik Sudderth

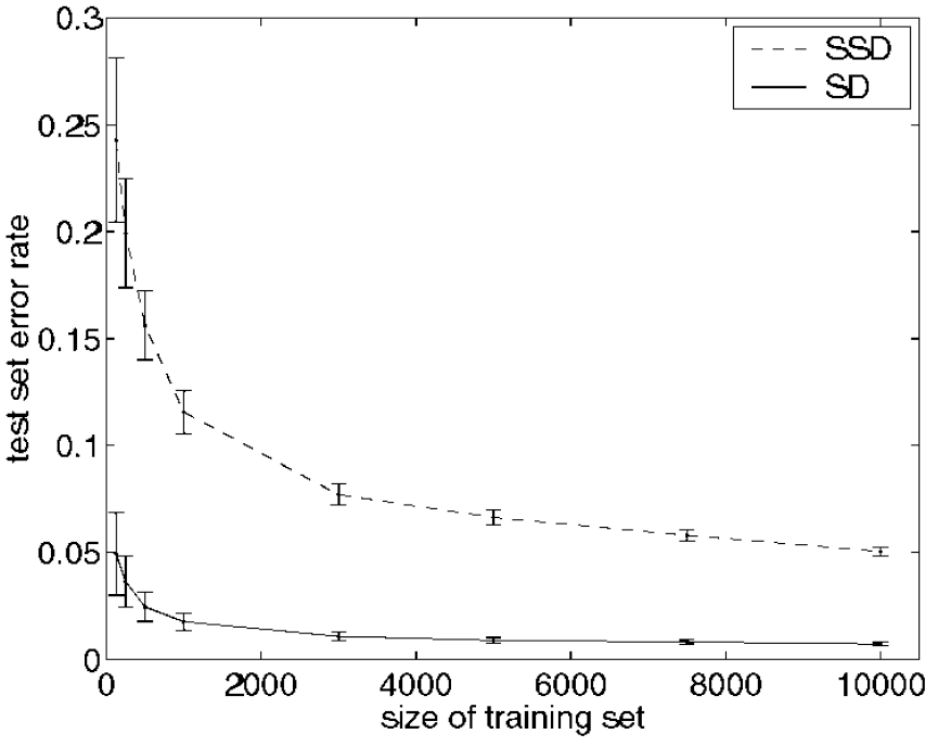
Lecture 10:
Logistic & Probit Regression
Gradient Descent & Newton's Method

Many figures courtesy Kevin Murphy's textbook,
Machine Learning: A Probabilistic Perspective

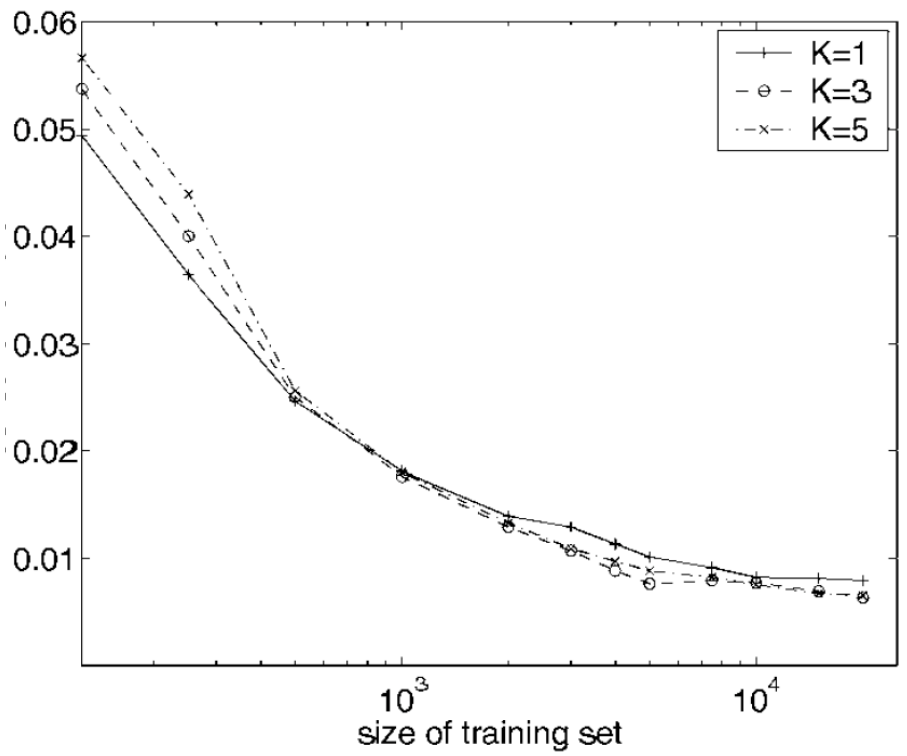
K-NN Cross-Validation: MNIST Digits (Homework 3)



K-NN Performance: Shape Contexts



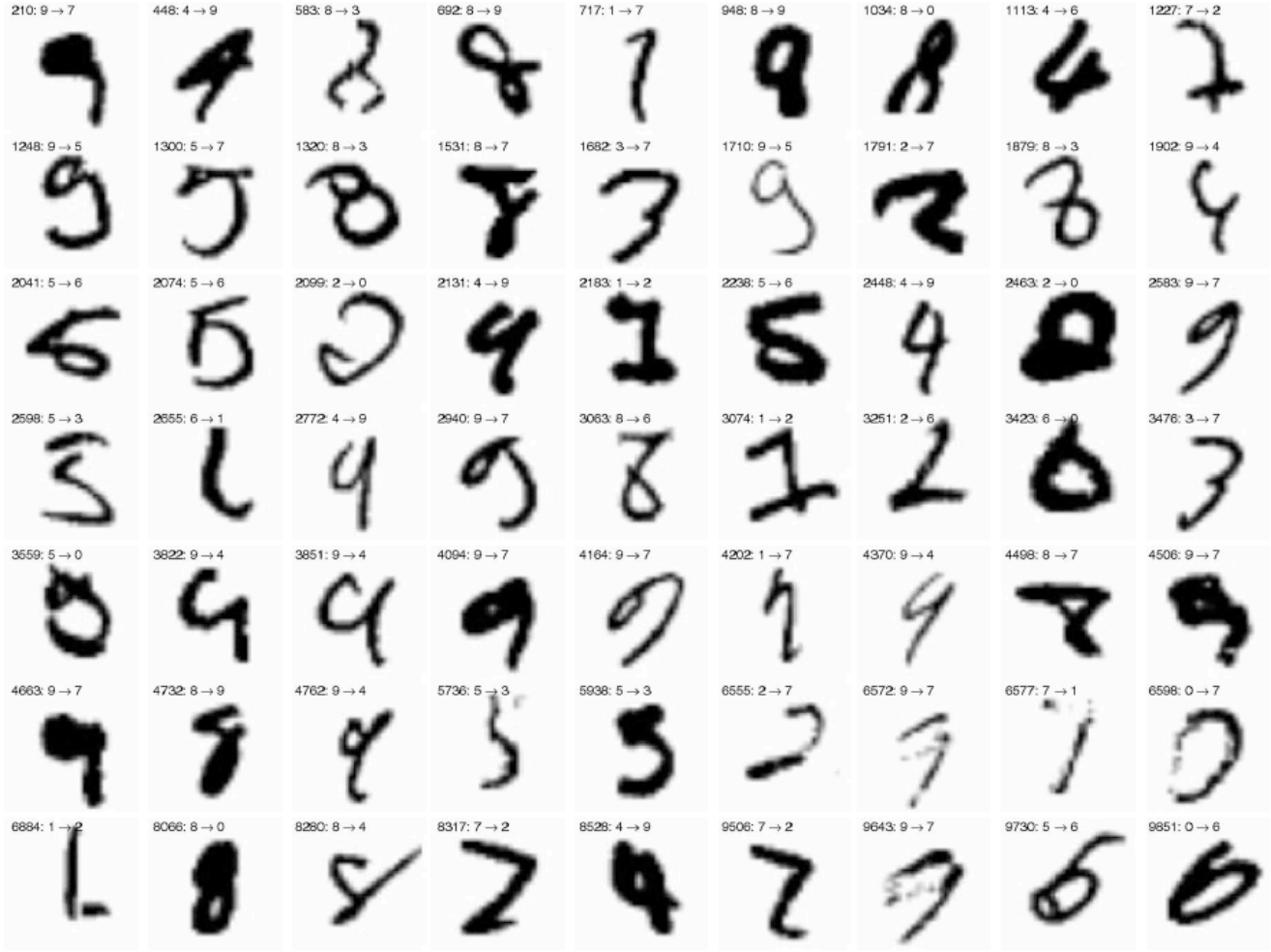
Alternative Distance Measures



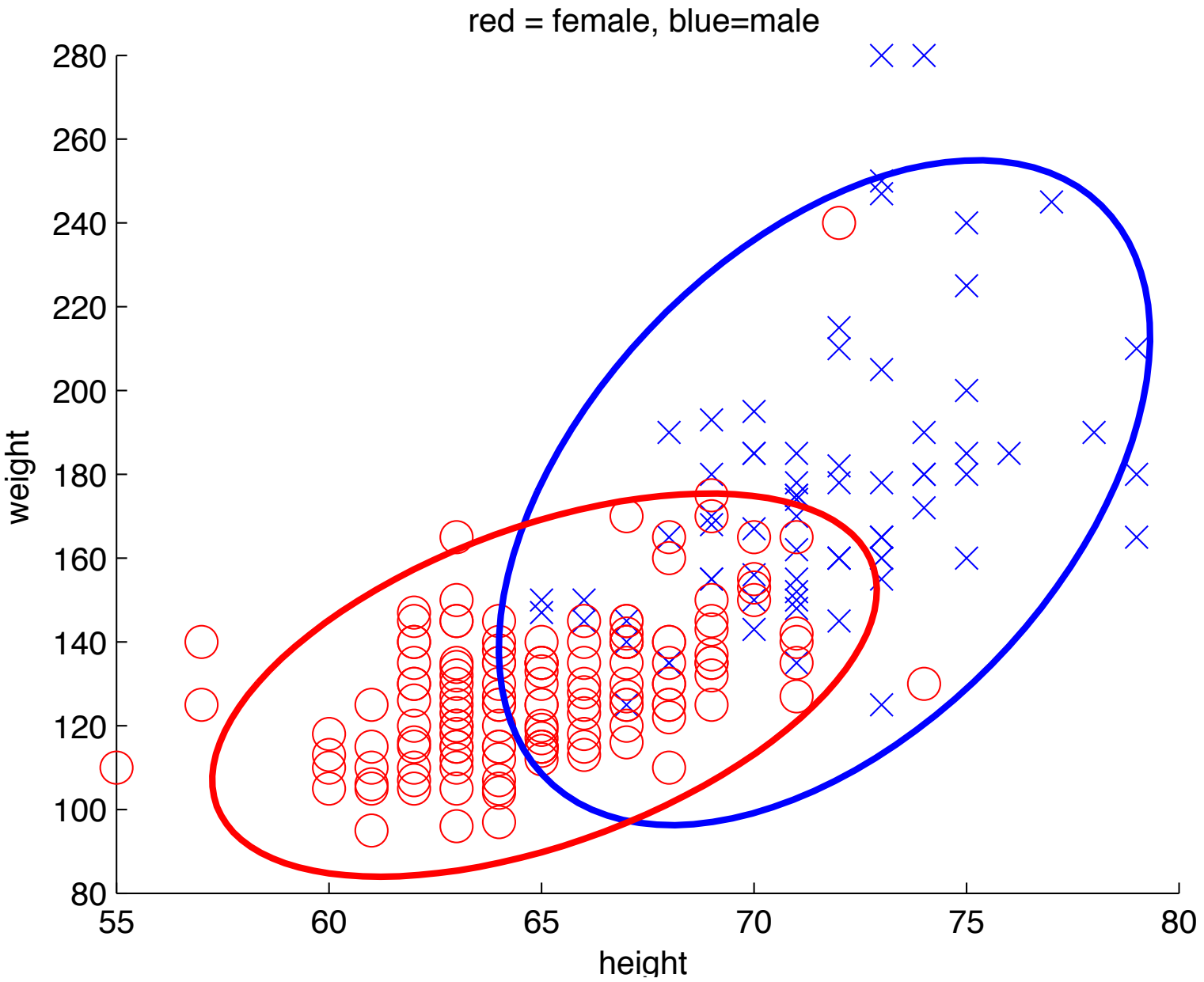
Choice of Neighborhood Size K

Belongie, Malik, & Puzicha, PAMI 2002

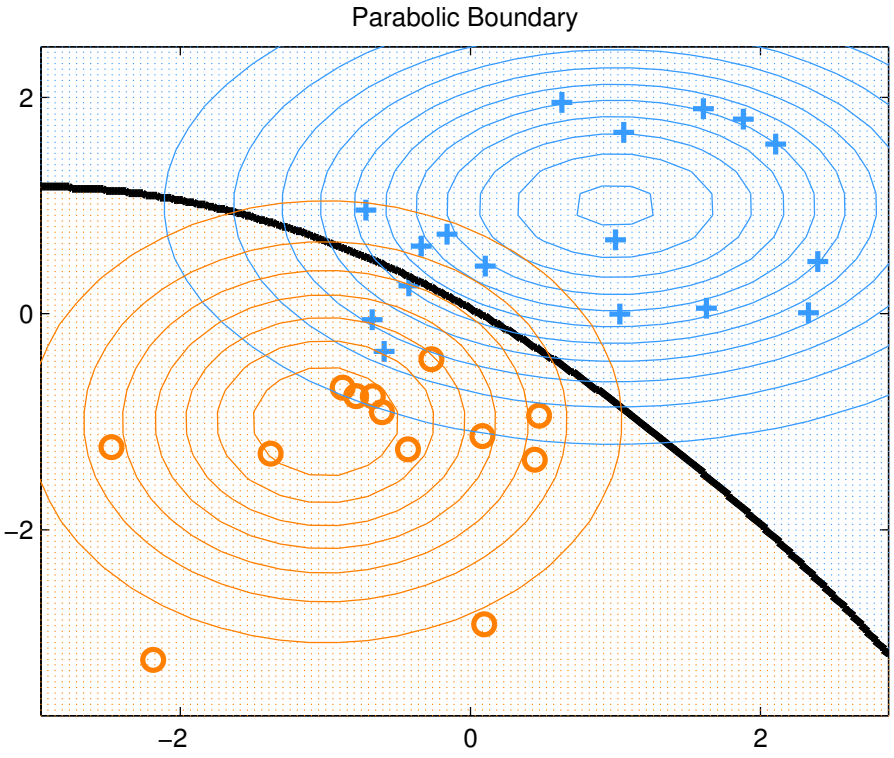
MNIST: Shape Context Errors

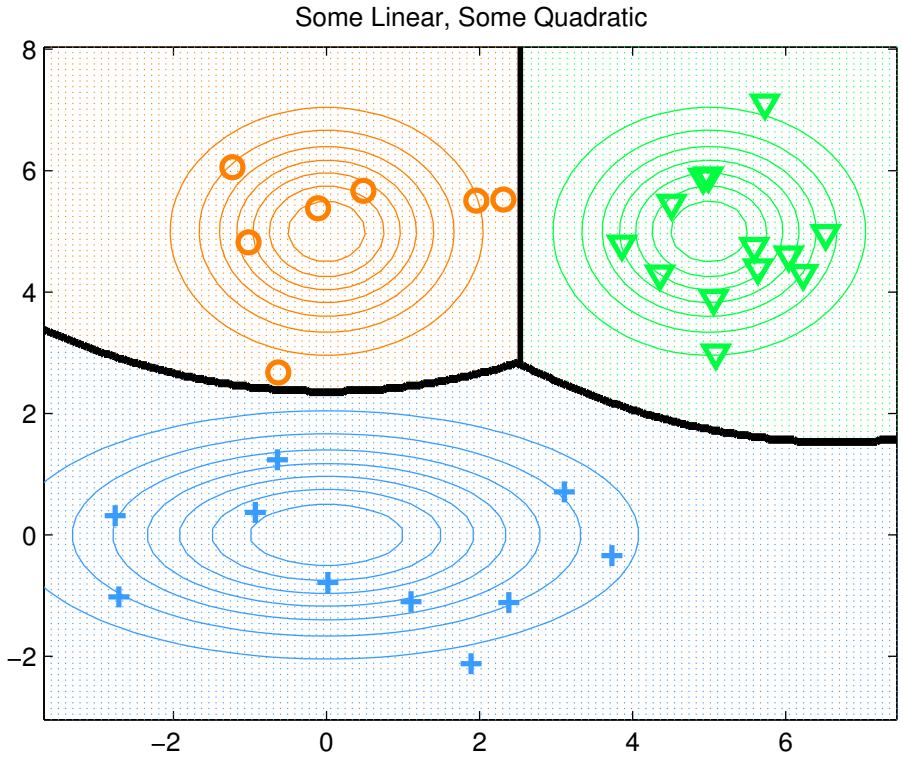


Generative Gaussian Classification



Quadratic Discriminant Analysis



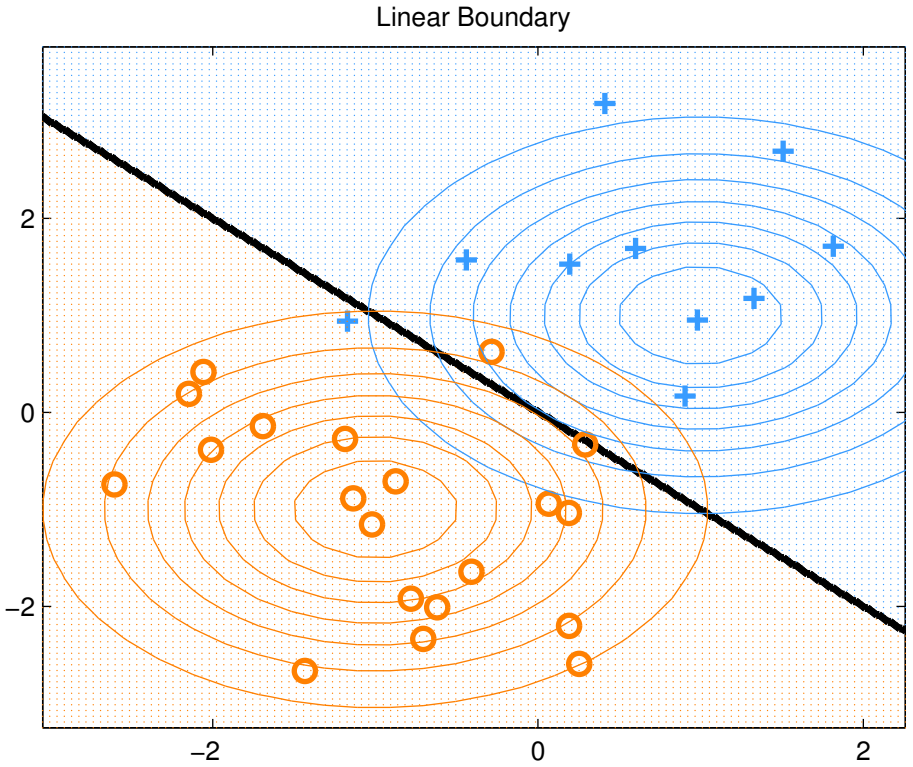


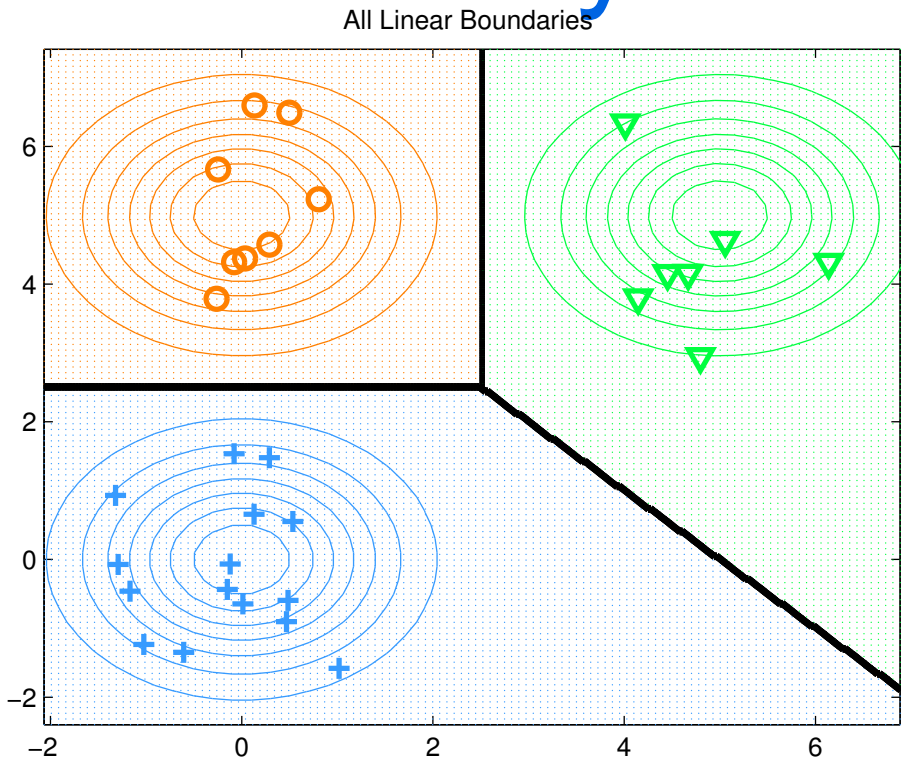
$$p(y = c|\mathbf{x}, \boldsymbol{\theta}) = \frac{\pi_c |2\pi\boldsymbol{\Sigma}_c|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1}(\mathbf{x} - \boldsymbol{\mu}_c)\right]}{\sum_{c'} \pi_{c'} |2\pi\boldsymbol{\Sigma}_{c'}|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_{c'})^T \boldsymbol{\Sigma}_{c'}^{-1}(\mathbf{x} - \boldsymbol{\mu}_{c'})\right]}$$

$$p(y \mid \pi) = \mathrm{Cat}(y \mid \pi) \qquad p(x \mid y = c, \theta) = \mathcal{N}(x \mid \mu_c, \Sigma_c)$$

Optimal decision boundaries are quadratic functions

Linear Discriminant Analysis





$$p(y=c|\mathbf{x},\boldsymbol{\theta}) \propto \pi_c \exp\left[\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c\right]$$

$$= \exp\left[\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log\pi_c\right]\exp[-\frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}]$$

Optimal decision boundaries are linear functions if $\Sigma_c = \Sigma$

Linear Discriminant Analysis

Further simplifying:

$$\gamma_c = -\frac{1}{2}\boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log \pi_c$$

$$\boldsymbol{\beta}_c = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c$$

$$p(y=c|\mathbf{x},\boldsymbol{\theta}) = \frac{e^{\boldsymbol{\beta}_c^T \mathbf{x}+\gamma_c}}{\sum_{c'} e^{\boldsymbol{\beta}_{c'}^T \mathbf{x}+\gamma_{c'}}} = \mathcal{S}(\boldsymbol{\eta})_c$$

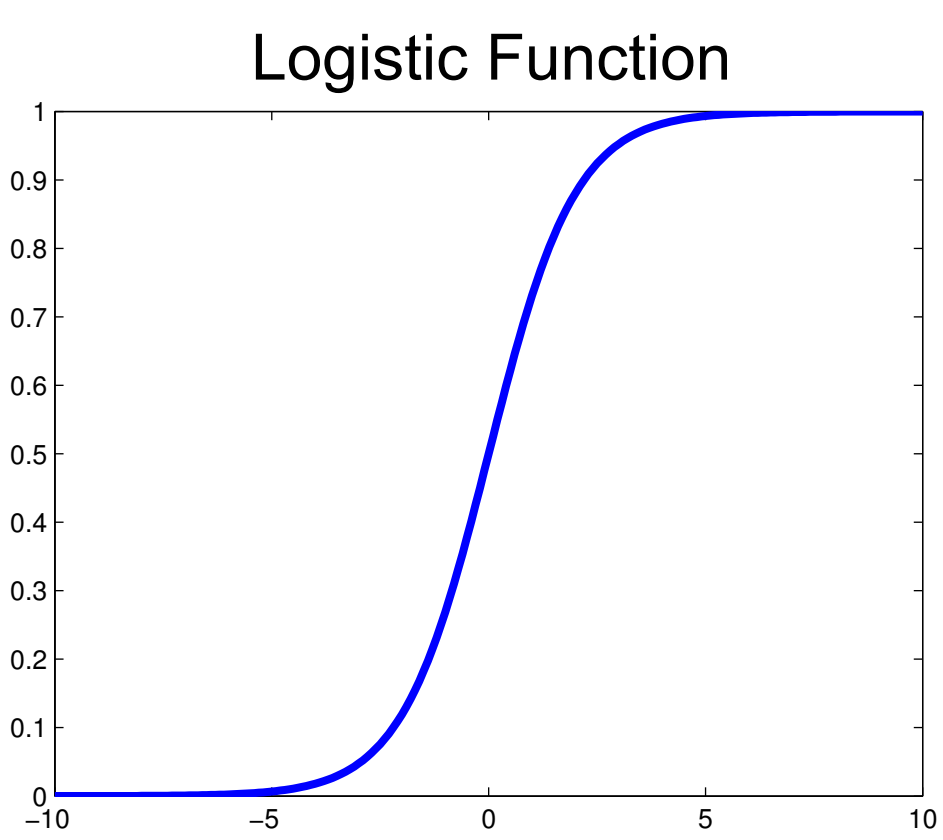
$$\boldsymbol{\eta} = [\boldsymbol{\beta}_1^T \mathbf{x} + \gamma_1, \ldots, \boldsymbol{\beta}_C^T \mathbf{x} + \gamma_C]$$

$$p(y=c|\mathbf{x},\boldsymbol{\theta}) \propto \pi_c \exp\left[\boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c\right]$$

$$= \exp\left[\boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log \pi_c\right] \exp[-\frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}]$$

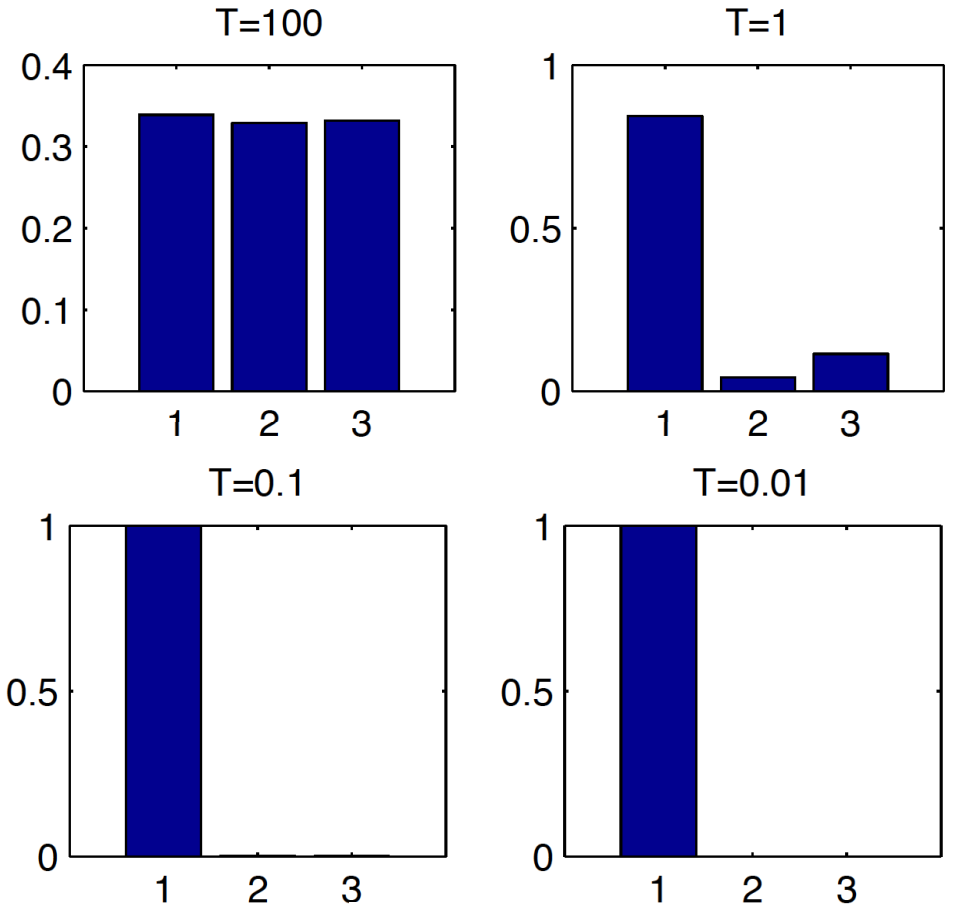
Optimal decision boundaries are linear functions if $\Sigma_c = \Sigma$

Logistic & Softmax Functions



$$\text{sigm}(\eta) := \frac{1}{1+\exp(-\eta)} = \frac{e^{\eta}}{e^{\eta}+1}$$

$$\mathcal{S}(\boldsymbol{\eta})_c = \frac{e^{\eta_c}}{\sum_{c'=1}^{C} e^{\eta_{c'}}}$$

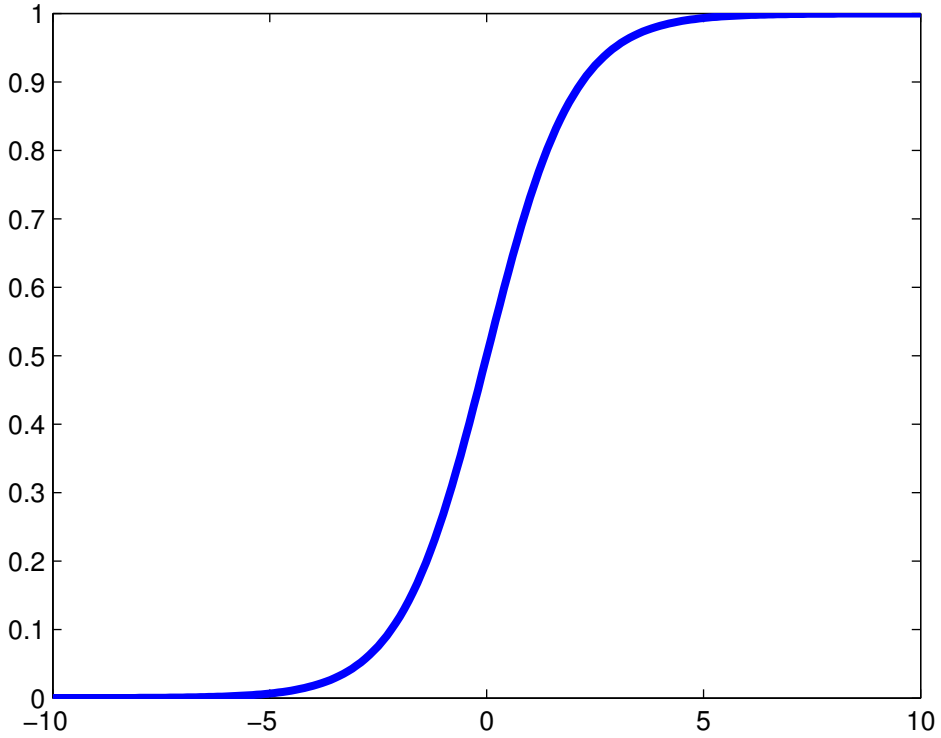


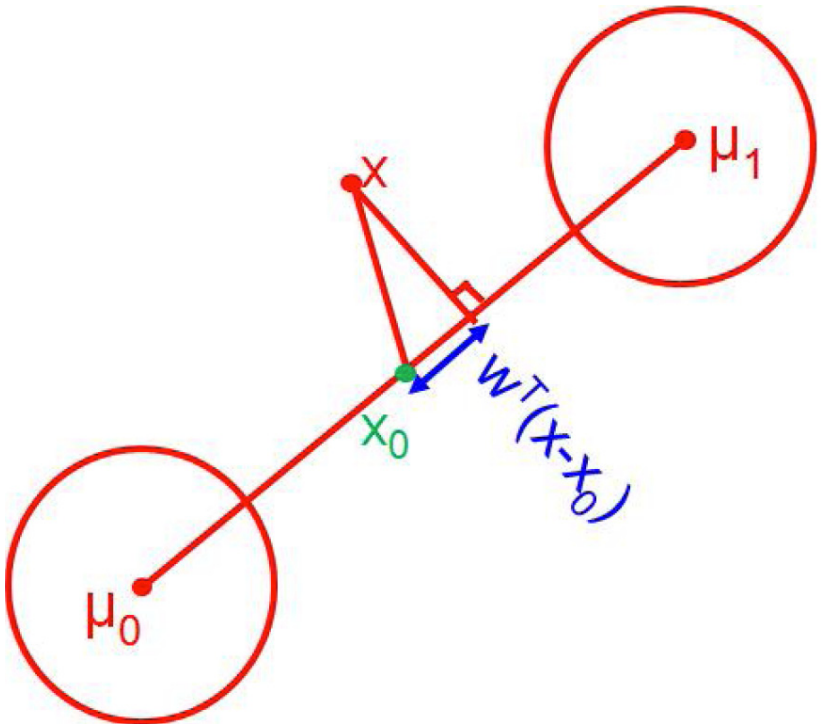
$$\mathcal{S}(\boldsymbol{\eta}/T)$$

$$\boldsymbol{\eta} = (3, 0, 1)$$

Binary Discriminant Analysis

Logistic Function





$$\text{sigm}(\eta) := \frac{1}{1+\exp(-\eta)} = \frac{e^{\eta}}{e^{\eta}+1}$$

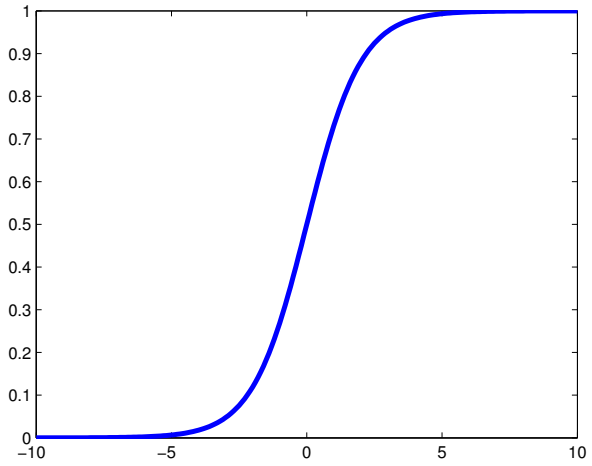
$$p(y=1|\mathbf{x},\boldsymbol{\theta}) = \sigma(\mathbf{w}^T(\mathbf{x}-\mathbf{x}_0))$$

$$\mathbf{w} = \boldsymbol{\beta}_1 - \boldsymbol{\beta}_0 = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

$$\mathbf{x}_0 = \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) - (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)\frac{\log(\pi_1/\pi_0)}{(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)}$$

Logistic Regression

$$p(y \mid x, w) = \mathrm{Ber}(y \mid \mathrm{sigm}(w^T \phi(x)))$$



$$\mathrm{sigm}(\eta) := \frac{1}{1+\exp(-\eta)} = \frac{e^{\eta}}{e^{\eta}+1}$$

- Linear discriminant analysis:

$$\phi(x_i) = [1, x_1, x_2, \ldots, x_d]$$

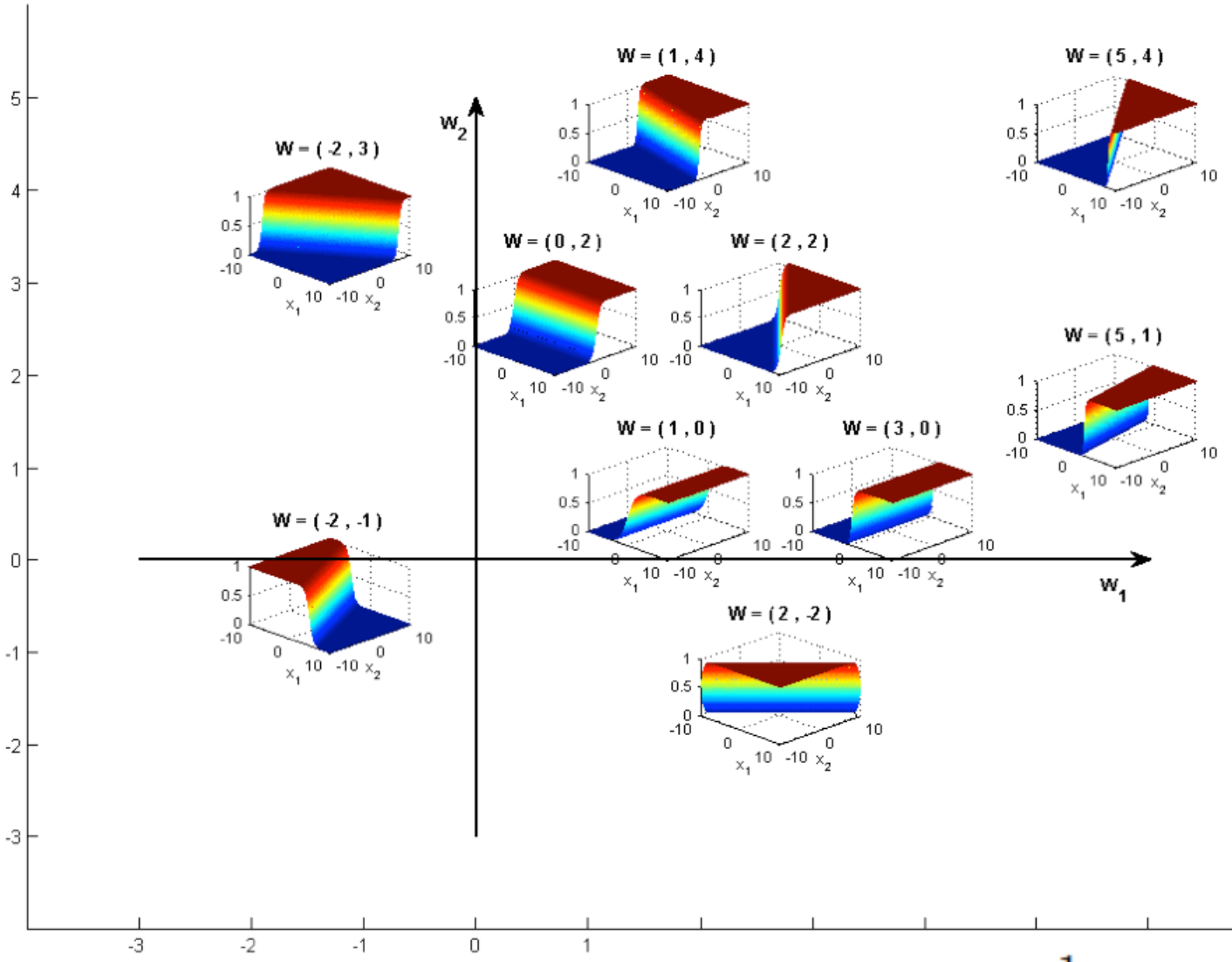
- Quadratic discriminant analysis:

$$\phi(x_i) = [1, x_1, \ldots, x_d, x_1^2, x_1 x_2, x_2^2, \ldots]$$

- Can derive weights from Gaussian generative model if that happens to be known, but more generally:
  - Choose any convenient feature set $\phi(x)$
  - Do discriminative Bayesian learning:

$$p(w \mid x, y) \propto p(w) \prod_{i=1}^{N} \mathrm{Ber}(y_i \mid \mathrm{sigm}(w^T \phi(x_i)))$$

Logistic Regression

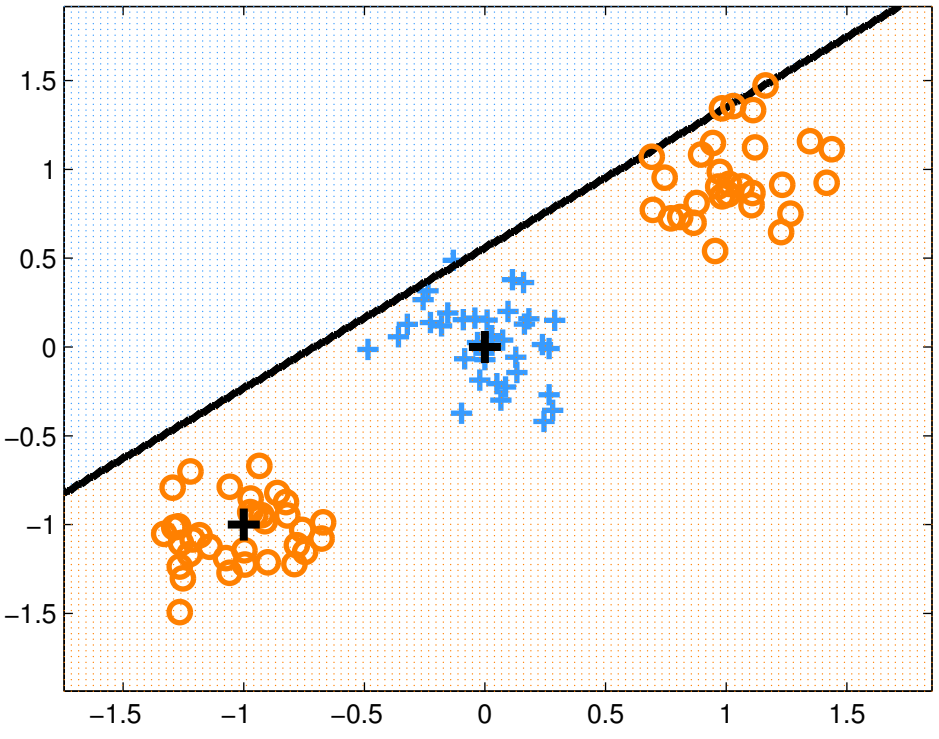


$$p(y|\mathbf{x}, \mathbf{w}) = \mathrm{Ber}(y|\mathrm{sigm}(\mathbf{w}^T\mathbf{x}))$$

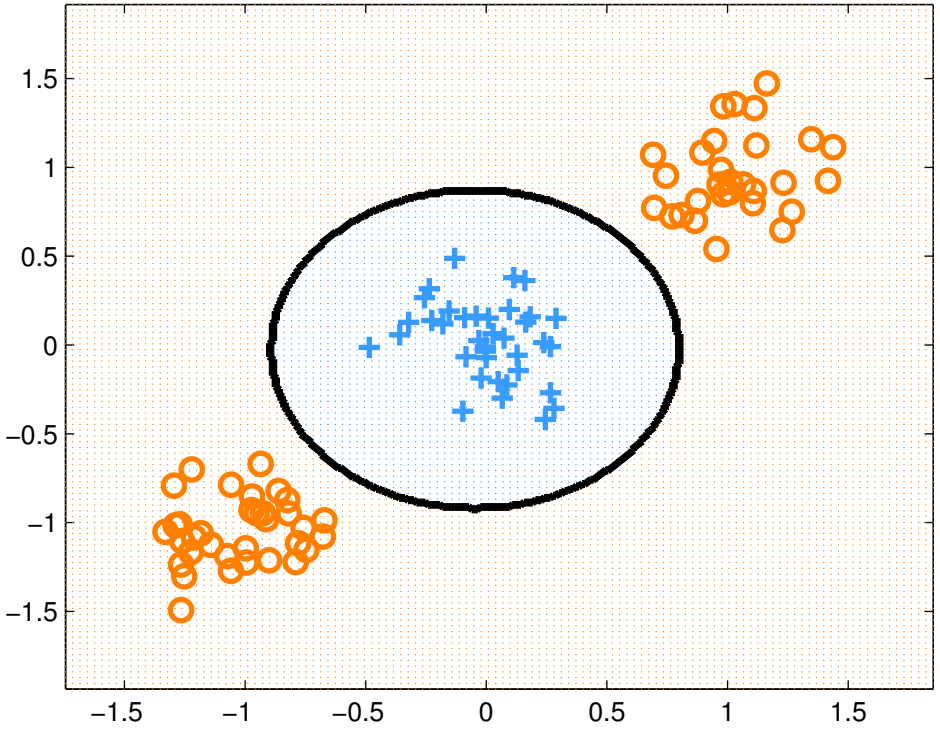
$$\mathrm{sigm}(\eta) := \frac{1}{1+\exp(-\eta)} = \frac{e^{\eta}}{e^{\eta}+1}$$

Decision Boundaries

By assumption, a linear function of input features.

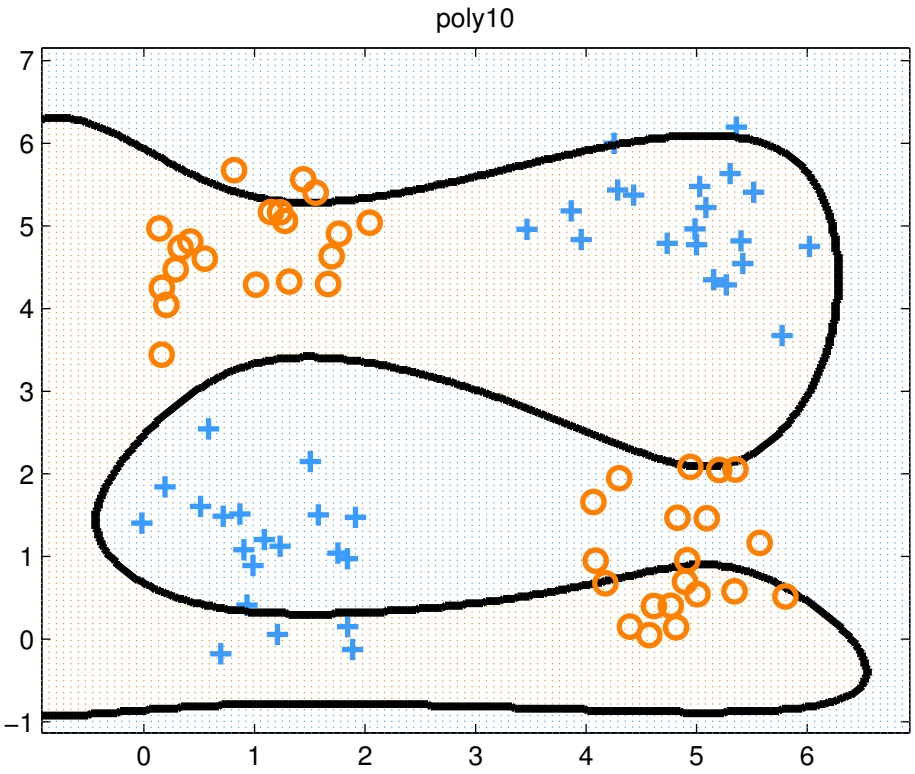


Raw features.

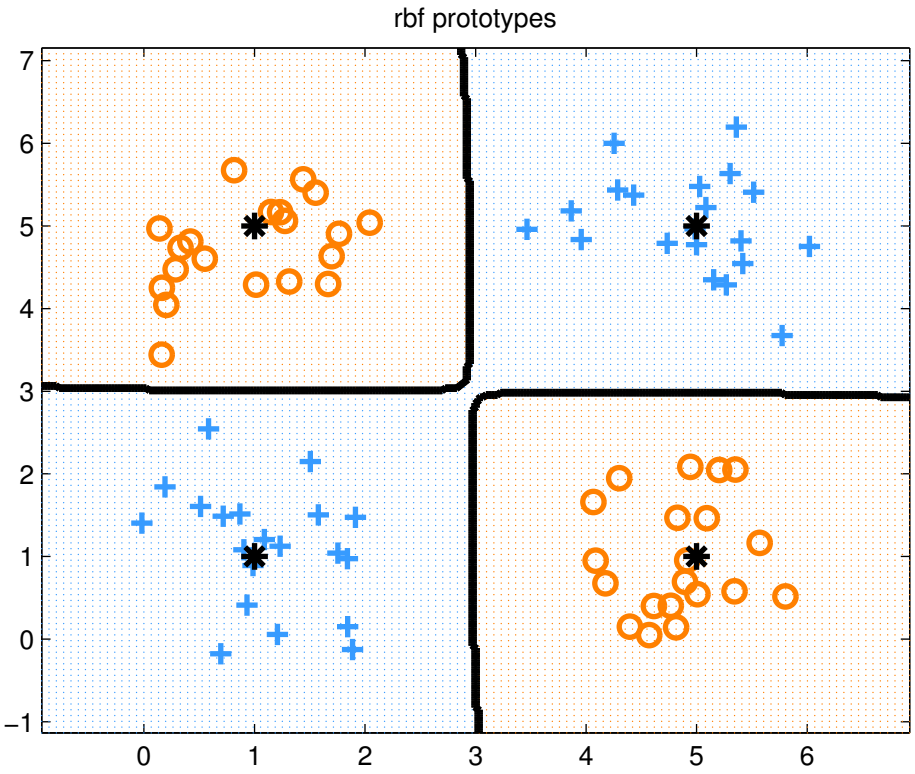


2^{nd}-order polynomial expansion.

Importance of Features: XOR



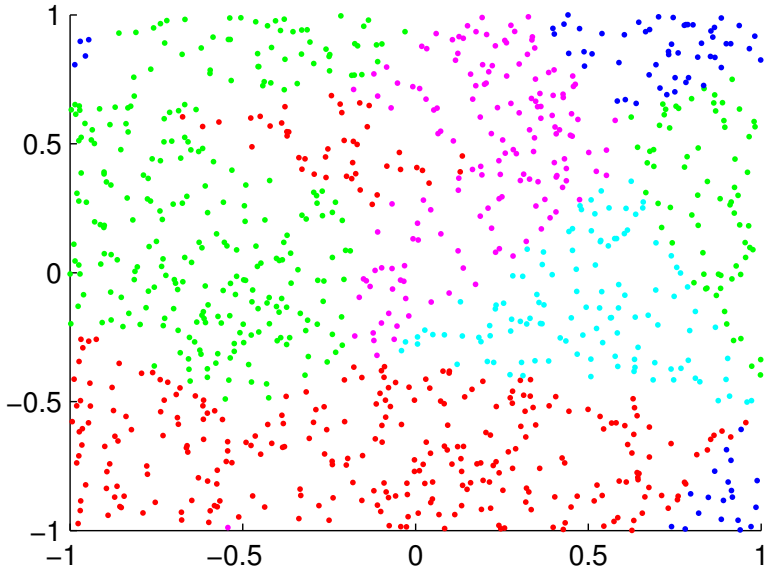
10^{th}-order polynomial.



4 radial basis functions.

Avoid RBF placement via kernel methods…

Multinomial Logistic Regression



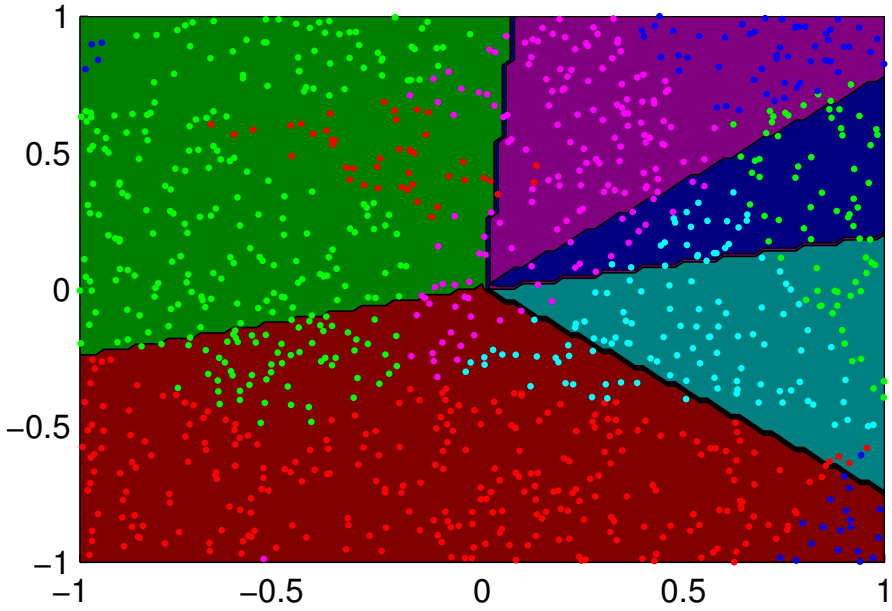
$$p(y|\mathbf{x}, \mathbf{W}) = \text{Cat}(y|\mathcal{S}(\mathbf{W}^T\mathbf{x}))$$

$$\mathcal{S}(\eta)_c = \frac{e^{\eta_c}}{\sum_{c'=1}^{C} e^{\eta_{c'}}}$$

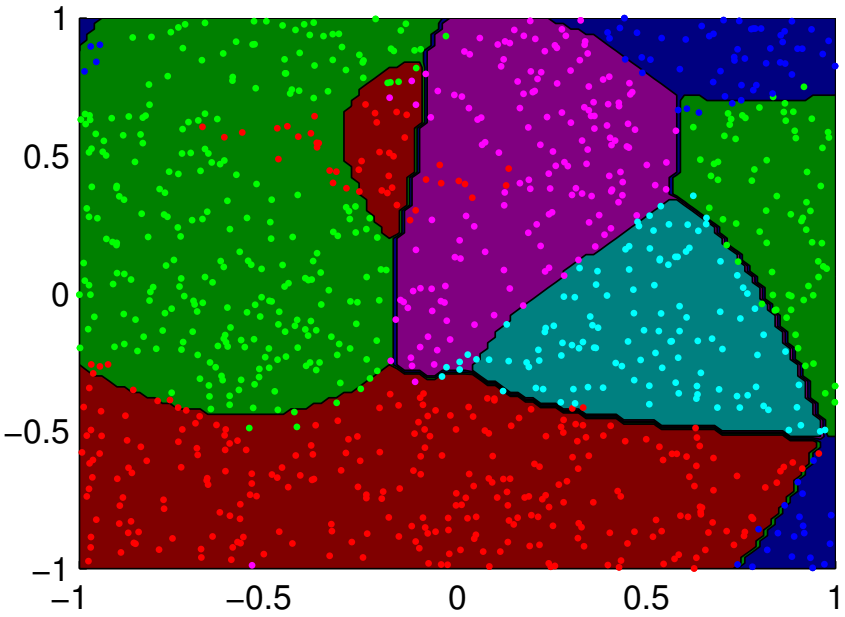
as $T \to 0$

$$\mathcal{S}(\eta/T)_c = \begin{cases} 1.0 & \text{if } c = \arg\max_{c'} \eta_{c'} \\ 0.0 & \text{otherwise} \end{cases}$$

Linear Multinomial Logistic Regression



Kernel−RBF Multinomial Logistic Regression



Learning via Optimization

ML Estimate: $\hat{w} = \arg\min_w -\sum_i \log p(y_i \mid x_i, w)$

MAP Estimate: $\hat{w} = \arg\min_w -\log p(w) - \sum_i \log p(y_i \mid x_i, w)$

Gradient vectors:

$$f : \mathbb{R}^M \to \mathbb{R}$$
$$\nabla_w f : \mathbb{R}^M \to \mathbb{R}^M$$
$$(\nabla_w f(w))_k = \frac{\partial f(w)}{\partial w_k}$$

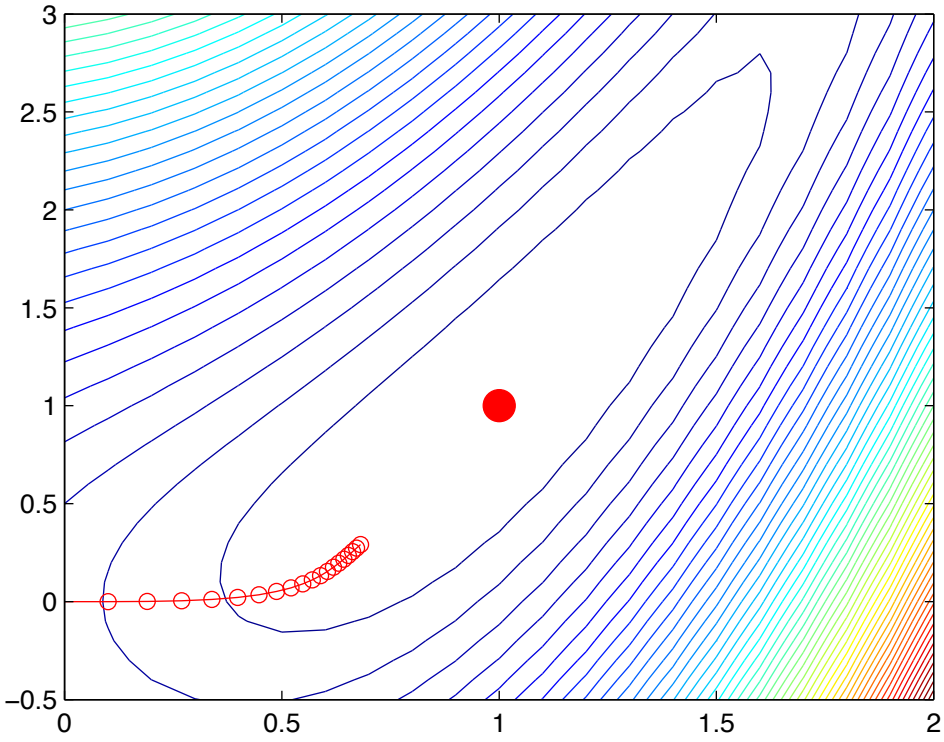
Hessian matrices:

$$\nabla_w^2 f : \mathbb{R}^M \to \mathbb{R}^{M \times M}$$
$$(\nabla_w f(w))_{k,\ell} = \frac{\partial^2 f(w)}{\partial w_k \partial w_\ell}$$

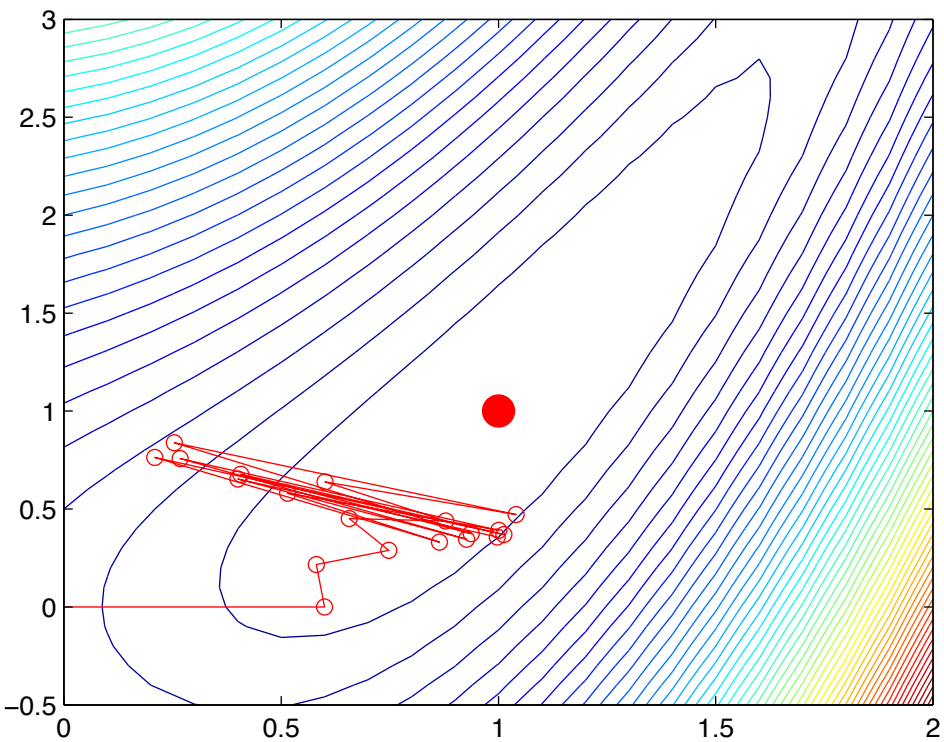
Optimization of Smooth Functions:

- *Closed form:* Find zero gradient points, check curvature
- *Iterative:* Initialize somewhere, use gradients to take steps towards better (by convention, smaller) values

Gradient (Steepest) Descent



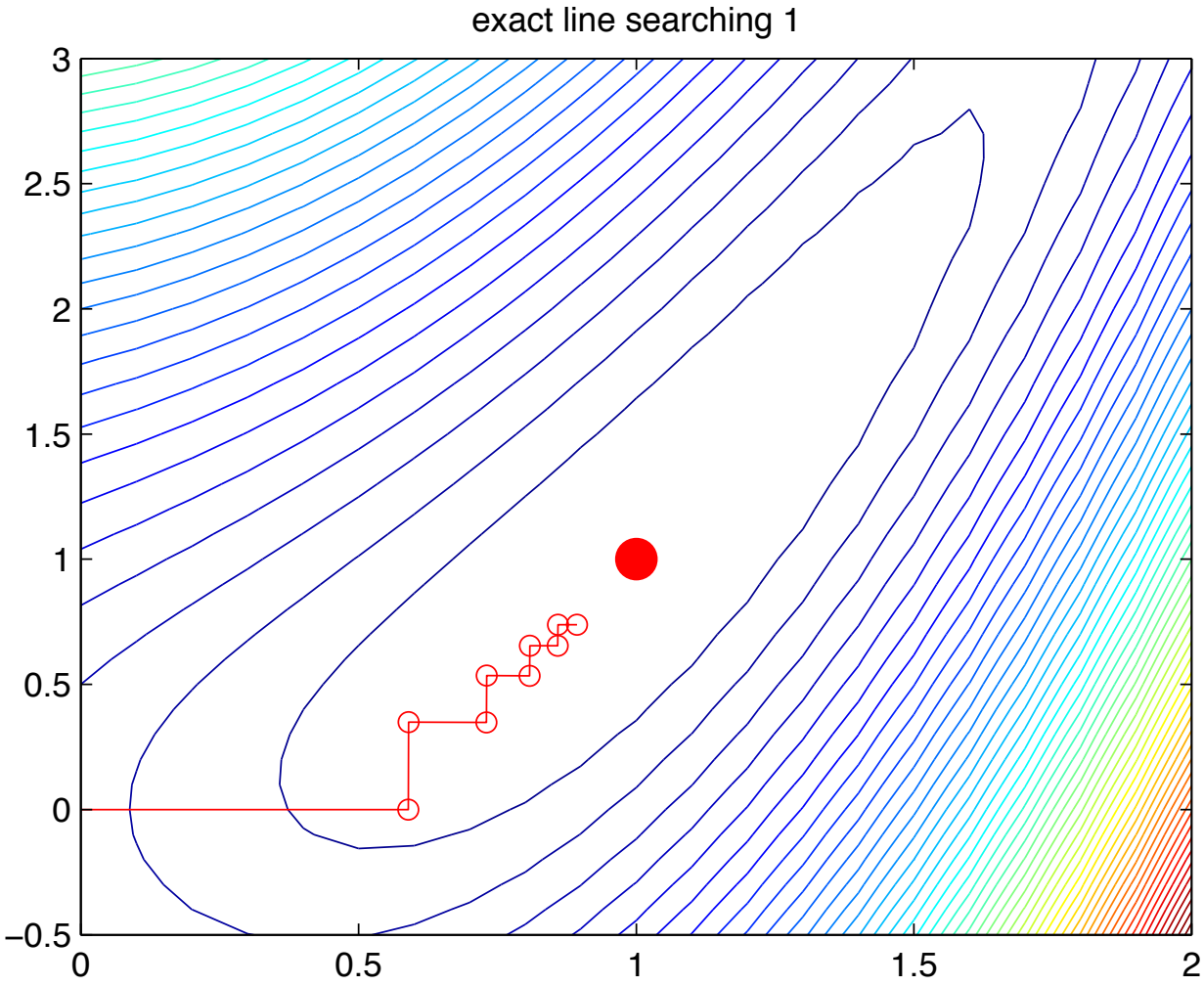
stepsize = 0.1



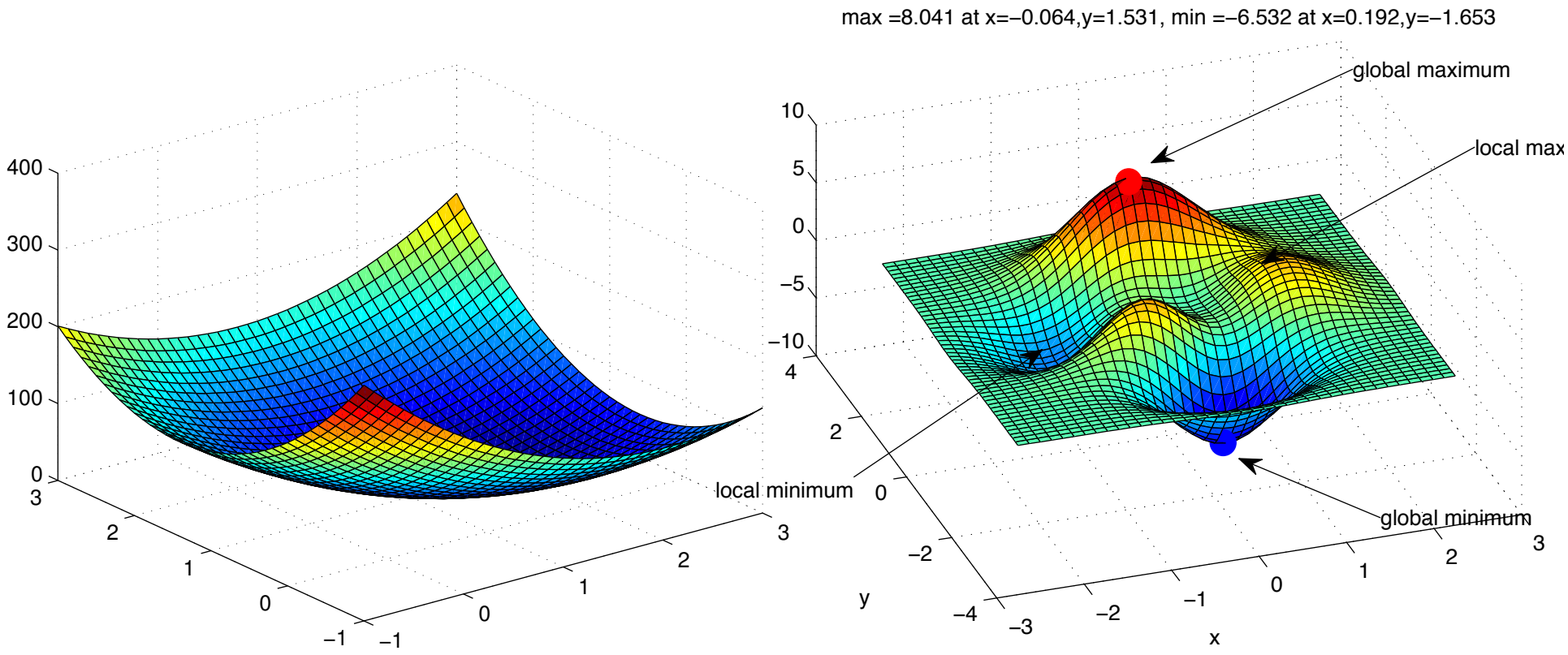
stepsize = 0.6

$$w_{k+1} = w_k - \eta_k \nabla_w f(w_k)$$

Descent via Line Search



Global & Local Optima

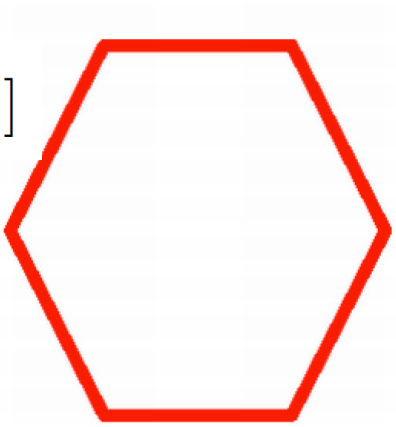


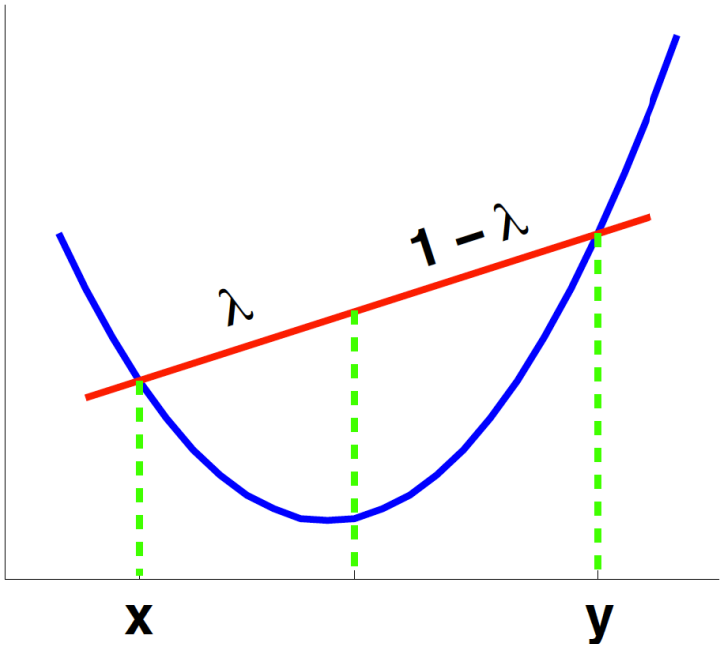
- A *globally convergent* algorithm is guaranteed to converge to a *local optimum* from any initialization
- Convergence to a *global optimum* is another issue…

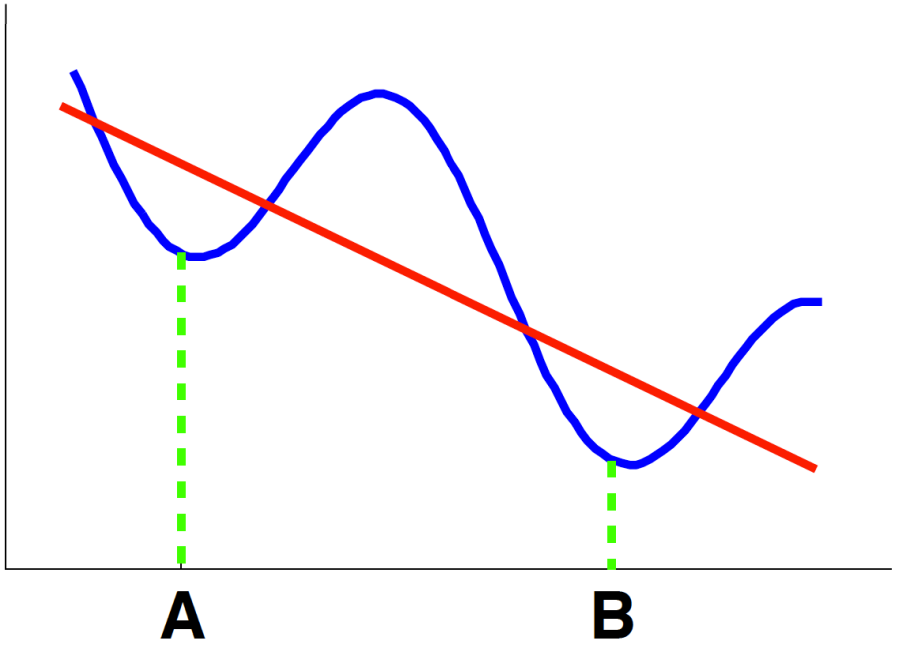
Convexity

$$\lambda\boldsymbol{\theta} + (1-\lambda)\boldsymbol{\theta}' \in \mathcal{S}, \quad \forall\, \lambda \in [0,1]$$

$$\boldsymbol{\theta}, \boldsymbol{\theta}' \in \mathcal{S}$$

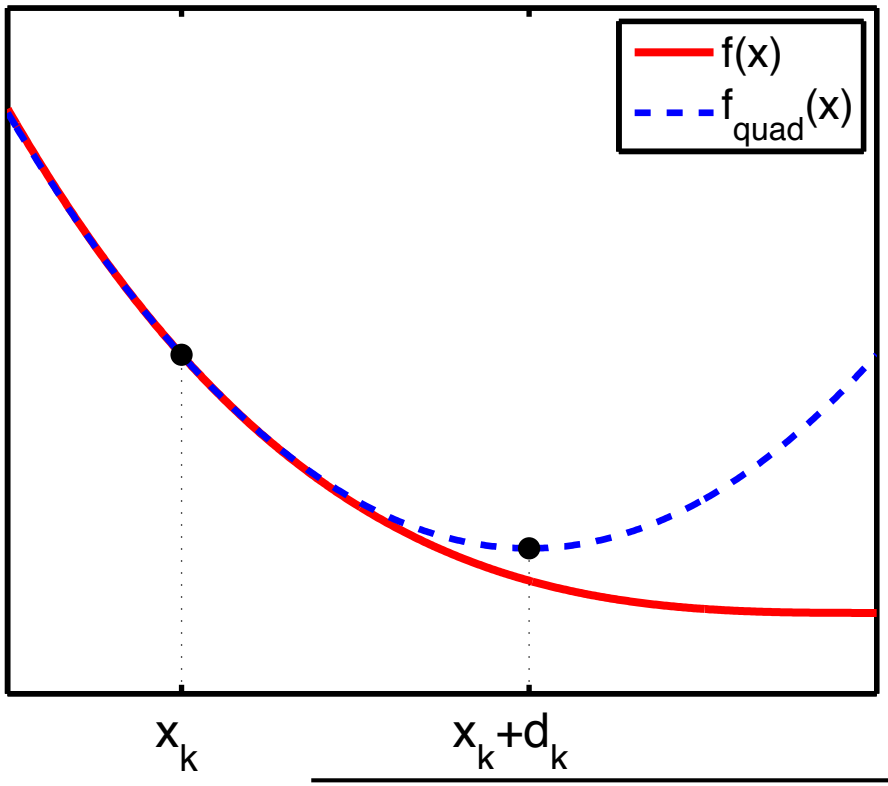


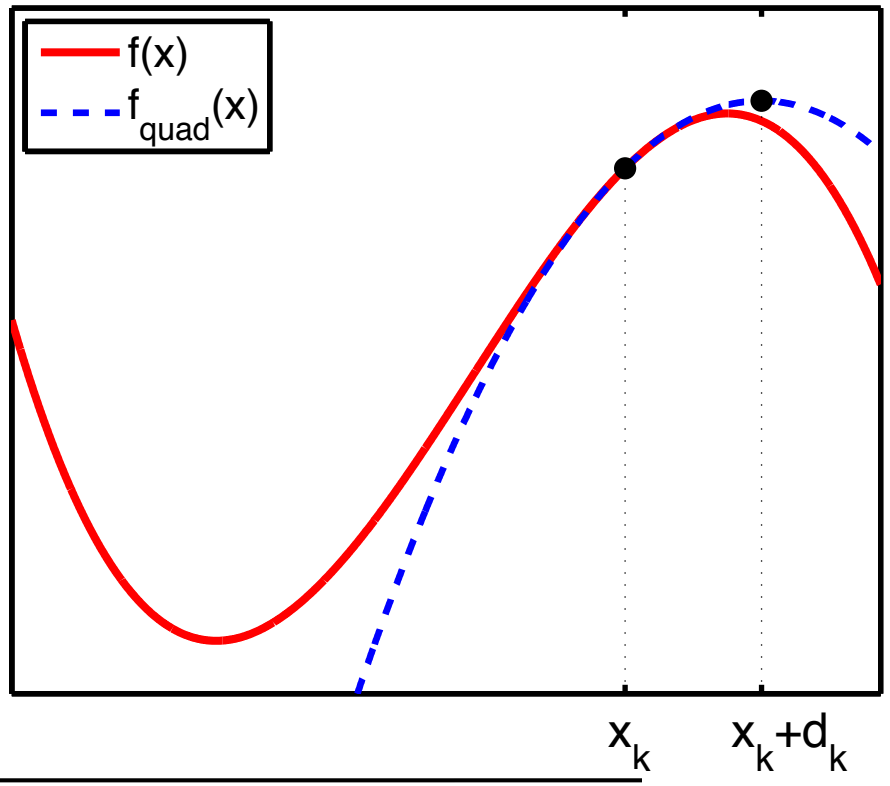




$$f(\lambda\boldsymbol{\theta} + (1-\lambda)\boldsymbol{\theta}') \leq \lambda f(\boldsymbol{\theta}) + (1-\lambda) f(\boldsymbol{\theta}')$$

Newton's Method





Algorithm 6.1: Newton's method for minimizing a strictly convex function

Initialize $\boldsymbol{\theta}_0$;
for $k = 1, 2, \ldots$ *until convergence* **do**
  Evaluate $\mathbf{g}_k = \nabla f(\boldsymbol{\theta}_k)$;
  Evaluate $\mathbf{H}_k = \nabla^2 f(\boldsymbol{\theta}_k)$;
  Solve $\mathbf{H}_k \mathbf{d}_k = -\mathbf{g}_k$ for $\mathbf{d}_k$;
  Use line search to find stepsize η_k along $\mathbf{d}_k$;
  $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \eta_k \mathbf{d}_k$;