CS 412 Introduction to Machine Learning

Logistic Regression

Instructor: Wei Tang

Department of Computer Science
University of Illinois at Chicago
Chicago IL 60607

https://tangw.people.uic.edu
tangw@uic.edu

Slides credit: Xinhua Zhang

Announcement

- MP #2 is available on Blackboard
- Linear regression & cross-validation
- Deadline: 10/20 Wes

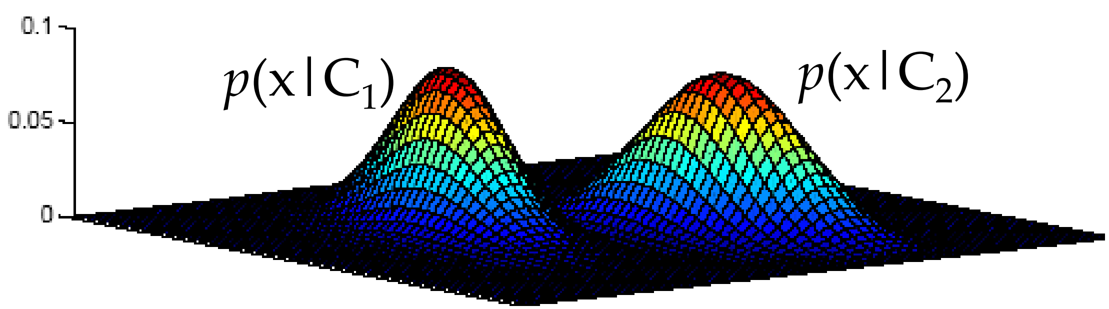
- Score of MP #1 is available on Blackboard
- Chat with TA on grading questions.

Generative- vs. Discriminative-based Classification

- Generative-based: Assume a model for $p(\mathbf{x}|C_i)$, use Bayes' rule to calculate $P(C_i|\mathbf{x})$

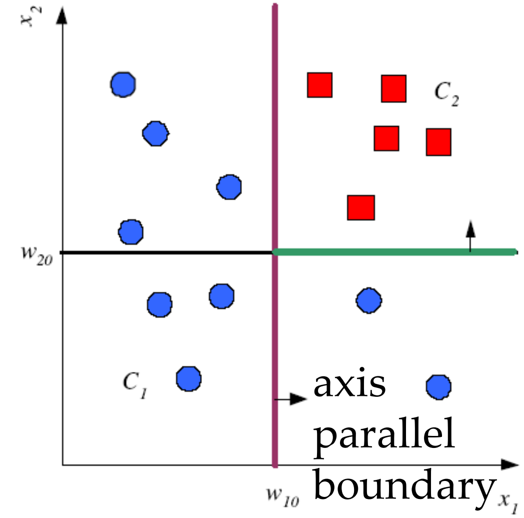
$$g_i(\mathbf{x}) = \log P(C_i|\mathbf{x})$$

Recall how to classify a new example.



- Discriminative-based: Assume a model for $g_i(\mathbf{x}|\Phi_i)$; no density estimation
- Estimating the boundaries is enough; no need to accurately estimate the densities inside the boundaries

Linear Discriminant

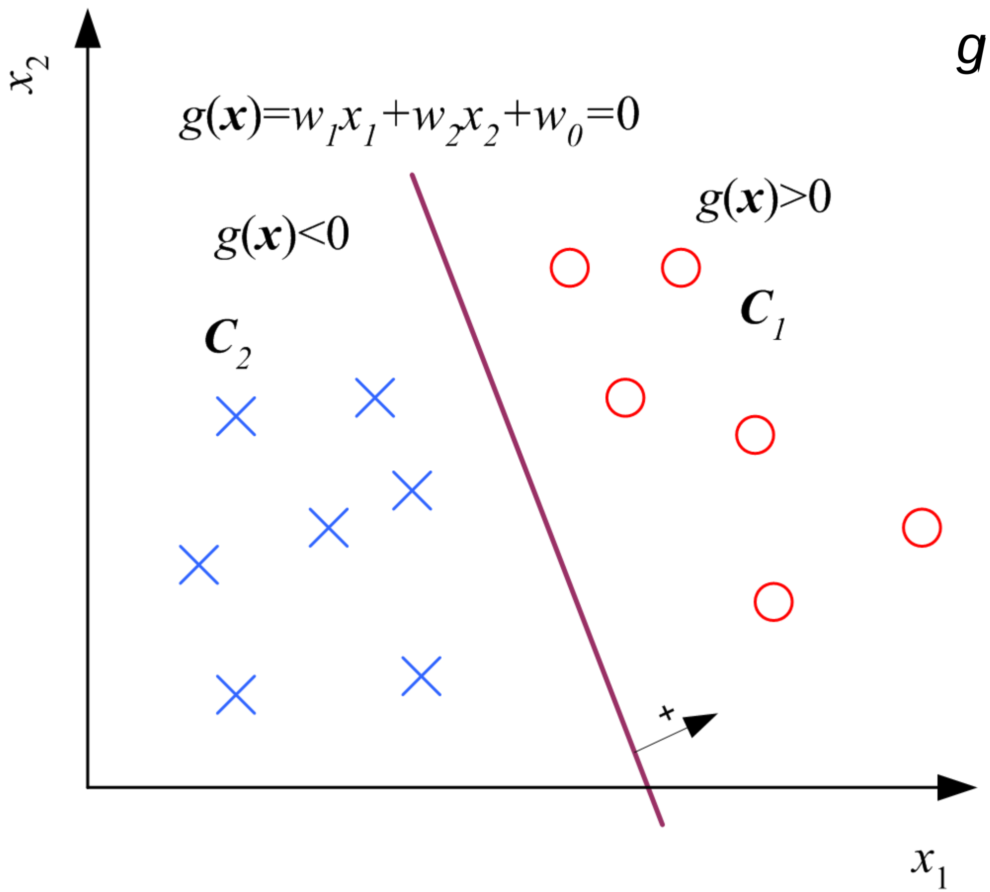


- Linear discriminant:

$$g_i(\mathbf{x}|\mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T\mathbf{x} + w_{i0} = \sum_{j=1}^{d} w_{ij}x_j + w_{i0}$$

- Advantages:
  - Simple: O(dK) space/computation for K classes
  - Knowledge extraction: Weighted sum of attributes; positive/negative weights, magnitudes (credit scoring)
  - Useful when classes are (almost) linearly separable

Two Classes

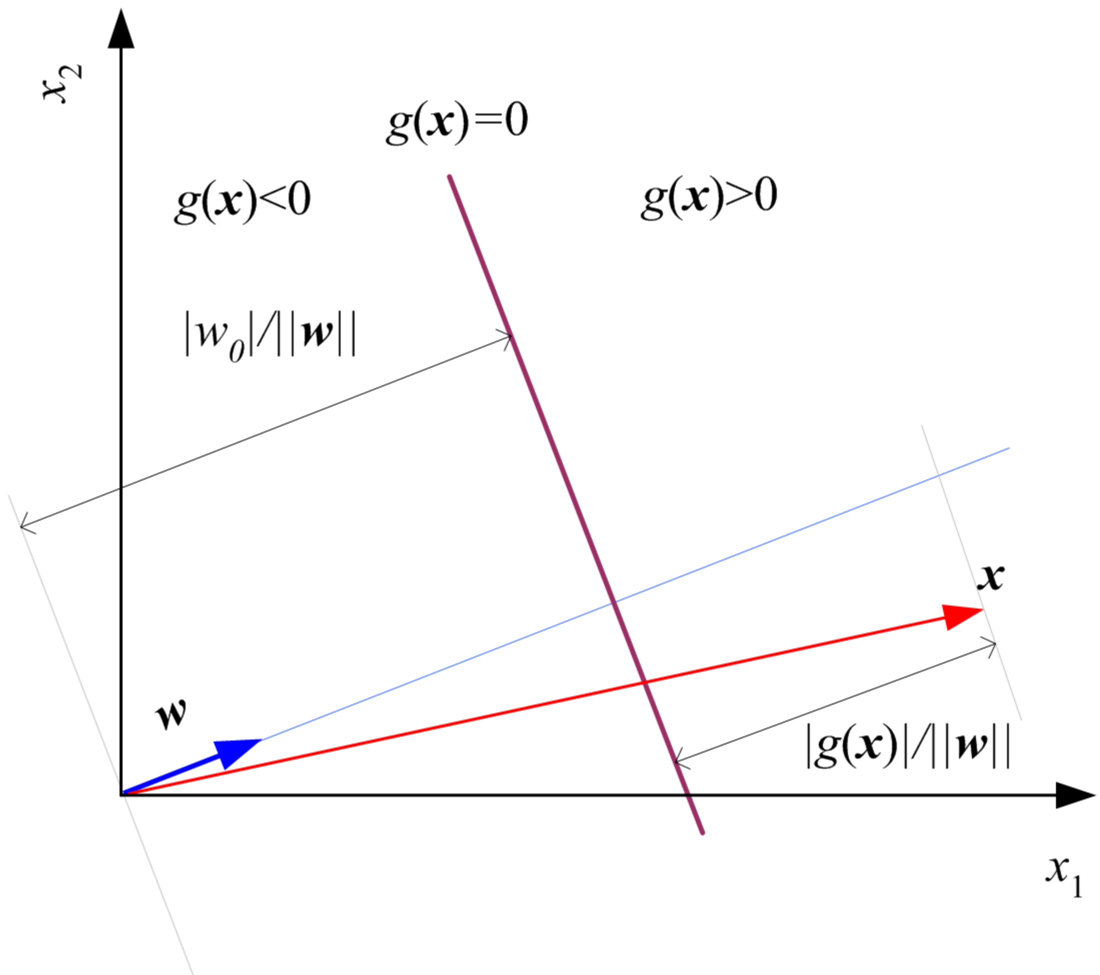


$$
\begin{aligned}
g(\mathbf{x}) &= g_1(\mathbf{x}) - g_2(\mathbf{x}) \\
&= \left(\mathbf{w}_1^T\mathbf{x} + w_{10}\right) - \left(\mathbf{w}_2^T\mathbf{x} + w_{20}\right) \\
&= \left(\mathbf{w}_1 - \mathbf{w}_2\right)^T\mathbf{x} + \left(w_{10} - w_{20}\right) \\
&= \mathbf{w}^T\mathbf{x} + w_0
\end{aligned}
$$

$$
\text{choose} \begin{cases} C_1 & \text{if } g(\mathbf{x}) > 0 \\ C_2 & \text{otherwise} \end{cases}
$$

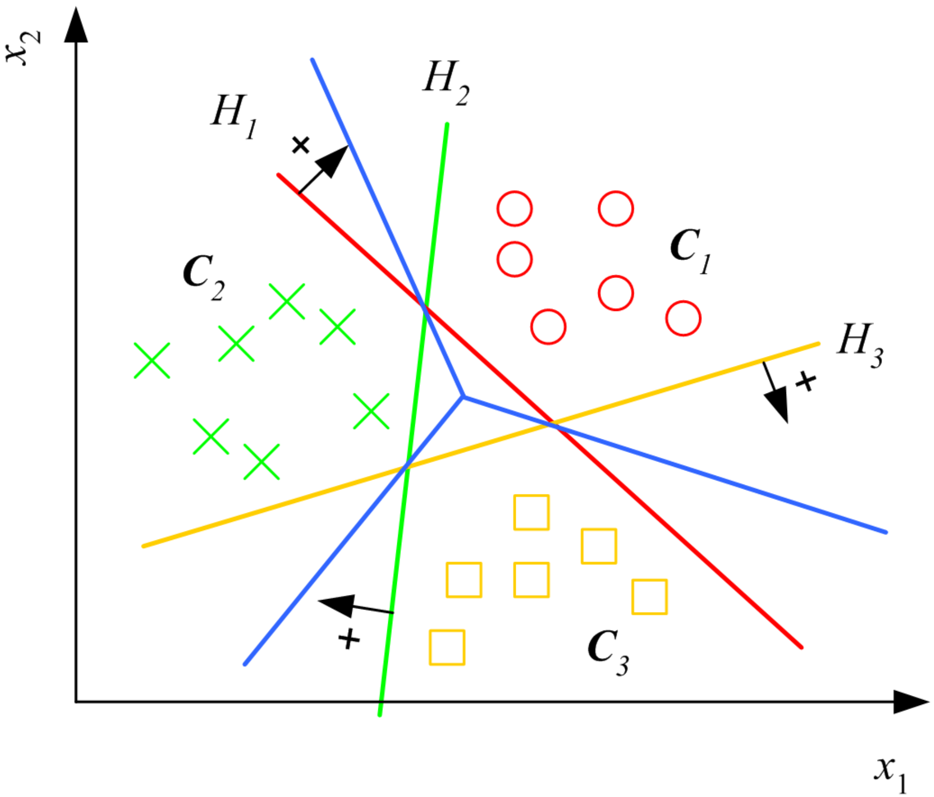
Geometry

$$\text{distance}(ax + by + c = 0, (x_0, y_0)) = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}.$$



Multiple Classes

$$g_i(\mathbf{x} \mid \mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T \mathbf{x} + w_{i0}$$



Choose C_i if

$$g_i(\mathbf{x}) = \max_{j=1}^{K} g_j(\mathbf{x})$$

Classes are
linearly separable

Logistic Regression

When $p\ (\boldsymbol{x} \mid C_i) \sim N\ (\boldsymbol{\mu}_i, \Sigma)$

$$g_i(\mathbf{x} \mid \mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

$$\mathbf{w}_i = \Sigma^{-1}\mu_i \quad w_{i0} = -\frac{1}{2}\mu_i^T \Sigma^{-1}\mu_i + \log P(C_i)$$

Now specialize to two classes:

$$y \equiv P(C_1 \mid \mathbf{x}) \text{ and } P(C_2 \mid \mathbf{x}) = 1 - y$$

$$\text{choose } C_1 \text{ if} \begin{cases} y > 0.5 \\ y/(1-y) > 1 \\ \log[y/(1-y)] > 0 \end{cases} \text{ and } C_2 \text{ otherwise}$$

logit transformation
or log odds of y

$$\text{logit}(P(C_1 \mid \mathbf{x})) = \log \frac{P(C_1 \mid \mathbf{x})}{1 - P(C_1 \mid \mathbf{x})} = \log \frac{P(C_1 \mid \mathbf{x})}{P(C_2 \mid \mathbf{x})}$$

$$= \log \frac{p(\mathbf{x} \mid C_1)}{p(\mathbf{x} \mid C_2)} + \log \frac{P(C_1)}{P(C_2)}$$

$$= \log \frac{(2\pi)^{-d/2} |\Sigma|^{-1/2} \exp\left[-(1/2)(\mathbf{x} - \mu_1)^T \Sigma^{-1} (\mathbf{x} - \mu_1)\right]}{(2\pi)^{-d/2} |\Sigma|^{-1/2} \exp\left[-(1/2)(\mathbf{x} - \mu_2)^T \Sigma^{-1} (\mathbf{x} - \mu_2)\right]} + \log \frac{P(C_1)}{P(C_2)}$$

$$= \mathbf{w}^T \mathbf{x} + w_0$$

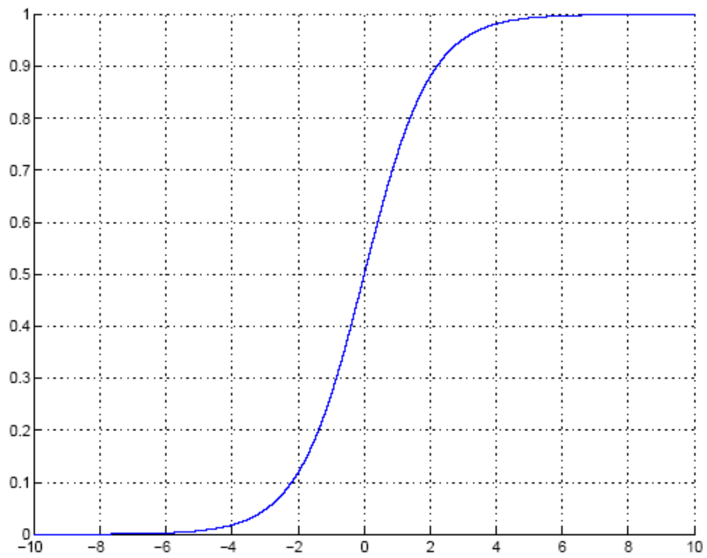
where $\mathbf{w} = \Sigma^{-1}(\mu_1 - \mu_2) \quad w_0 = -\frac{1}{2}(\mu_1 + \mu_2)^T \Sigma^{-1} (\mu_1 - \mu_2) + \log \frac{P(C_1)}{P(C_2)}$

The inverse of logit

$$\log \frac{P(C_1 \mid \mathbf{x})}{1 - P(C_1 \mid \mathbf{x})} = \mathbf{w}^T \mathbf{x} + w_0$$

$$P(C_1 \mid \mathbf{x}) = \text{sigmoid}(\mathbf{w}^T \mathbf{x} + w_0) = \frac{1}{1 + \exp\left[-(\mathbf{w}^T \mathbf{x} + w_0)\right]}$$

Sigmoid (Logistic) Function



$$S(x) = \frac{1}{1+e^{-x}}$$

In binary classification:

Calculate $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ and choose C_1 if $g(\mathbf{x}) > 0$, or

Calculate $y = \text{sigmoid}\left(\mathbf{w}^T\mathbf{x} + w_0\right)$ and choose C_1 if $y > 0.5$

$$= \text{sigmoid}(a), \text{where } a = \mathbf{w}^T\mathbf{x} + w_0 \qquad \frac{dy}{da} = y(1-y)$$

Training: Two Classes
Logistic discrimination

$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_t \quad r^t \mid \mathbf{x}^t \sim \text{Bernoulli}(y^t)$$

$$y = P(C_1 \mid \mathbf{x}) = \frac{1}{1 + \exp\left[-\left(\mathbf{w}^T \mathbf{x} + w_0\right)\right]}$$

$$l(\mathbf{w}, w_0 \mid \mathcal{X}) = \prod_t \left(y^t\right)^{\left(r^t\right)} \left(1 - y^t\right)^{\left(1 - r^t\right)}$$

$$E = -\log l$$

$$E(\mathbf{w}, w_0 \mid \mathcal{X}) = -\sum_t r^t \log y^t + \left(1 - r^t\right) \log \left(1 - y^t\right)$$

Bad news: no closed-form solution to minimize $E(\mathbf{w})$

Good news: $E(\mathbf{w})$ is a convex function of $\mathbf{w}$! no locally optimal solutions

Good news: convex functions are (relatively) easy to minimize

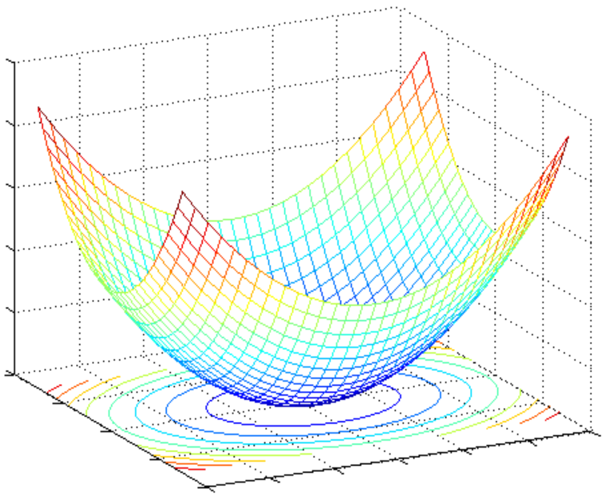
Reminder: we often divide $E(\mathbf{w}, w_0 | \mathrm{X})$ by N (number of examples)

Gradient-Descent

- E($\boldsymbol{w}$|X) is error with parameters $\boldsymbol{w}$ on sample X

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} E(\boldsymbol{w} \mid X)$$

- Gradient



$$\nabla_w E = \left[\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \ldots, \frac{\partial E}{\partial w_d} \right]^T$$

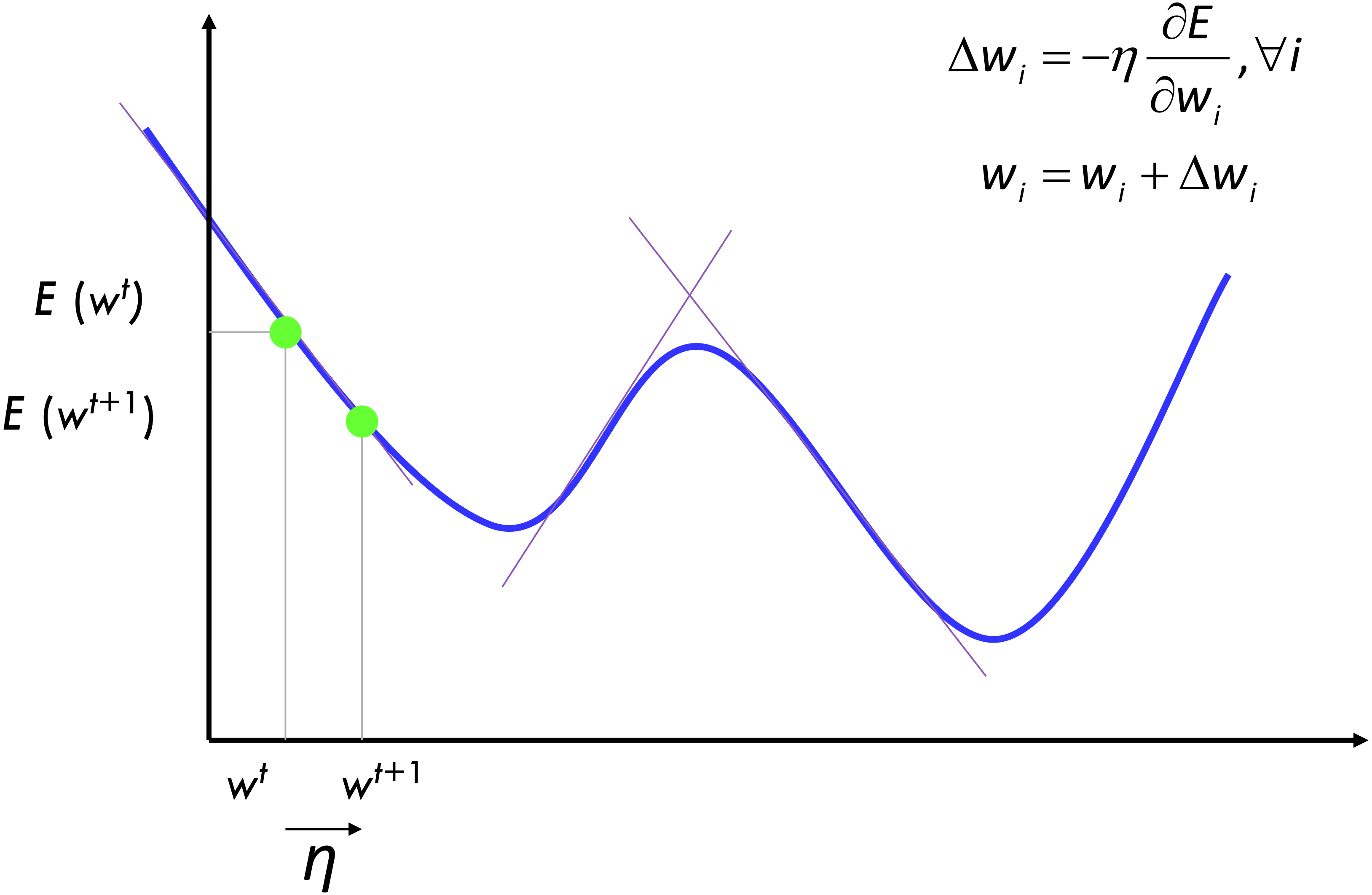
- Gradient-descent:

  Starts from random $\boldsymbol{w}$ and updates $\boldsymbol{w}$ iteratively in the negative direction of gradient

Gradient-Descent

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \forall i$$

$$w_i = w_i + \Delta w_i$$



Training: Gradient-Descent
Formula of the gradient (two classes)

$$E(\mathbf{w},w_0 \mid \mathcal{X}) = -\sum_t r^t \log y^t + \left(1-r^t\right)\log\left(1-y^t\right)$$

If $y = \text{sigmoid}(o)$, then $\frac{dy}{do} = y(1-y)$

$$\Delta w_j = -\eta \frac{\partial E}{\partial w_j} = \eta \sum_t \left(\frac{r^t}{y^t} - \frac{1-r^t}{1-y^t}\right) y^t\left(1-y^t\right)x_j^t$$

$$= \eta \sum_t \left(r^t - y^t\right)x_j^t, j = 1,\ldots,d$$

$$\Delta w_0 = -\eta \frac{\partial E}{\partial w_0} = \eta \sum_t \left(r^t - y^t\right)$$

handle w_0 in a unified fashion
by setting $x_0^t = 1$

$$y = P(C_1 \mid \mathbf{x}) = \frac{1}{1+\exp\left[-\left(\mathbf{w}^T\mathbf{x}+w_0\right)\right]}$$

$= \text{sigmoid}(o)$,
where $o = \mathbf{w}^T\mathbf{x} + w_0$

Training: Gradient-Descent (two classes)

For $j = 0, \ldots, d$
    $w_j \leftarrow \text{rand}(-0.01, 0.01)$
Repeat
    For $j = 0, \ldots, d$
        $\Delta w_j \leftarrow 0$
    For $t = 1, \ldots, N$
        $o \leftarrow 0$
        For $j = 0, \ldots, d$
            $o \leftarrow o + w_j x_j^t$
        $y \leftarrow \text{sigmoid}(o)$
        For $j = 0, \ldots, d$
            $\Delta w_j \leftarrow \Delta w_j + (r^t - y) x_j^t$
    For $j = 0, \ldots, d$
        $w_j \leftarrow w_j + \eta \Delta w_j$
Until convergence

$$o^t = \sum_{j=0}^{d} w_j x_j^t$$

$$y^t = sigmoid(o^t)$$

$$\Delta w_j = \eta \sum_{t=1}^{N} (r^t - y^t) x_j^t$$

$j = 1, \ldots, d$

$X \in \mathbb{R}^{N \times d}$ $\quad r \in \{0,1\}^N \xrightarrow{\text{Learning}} w \in \mathbb{R}^d$

Repeat

$o = Xw \qquad o \in \mathbb{R}^N$

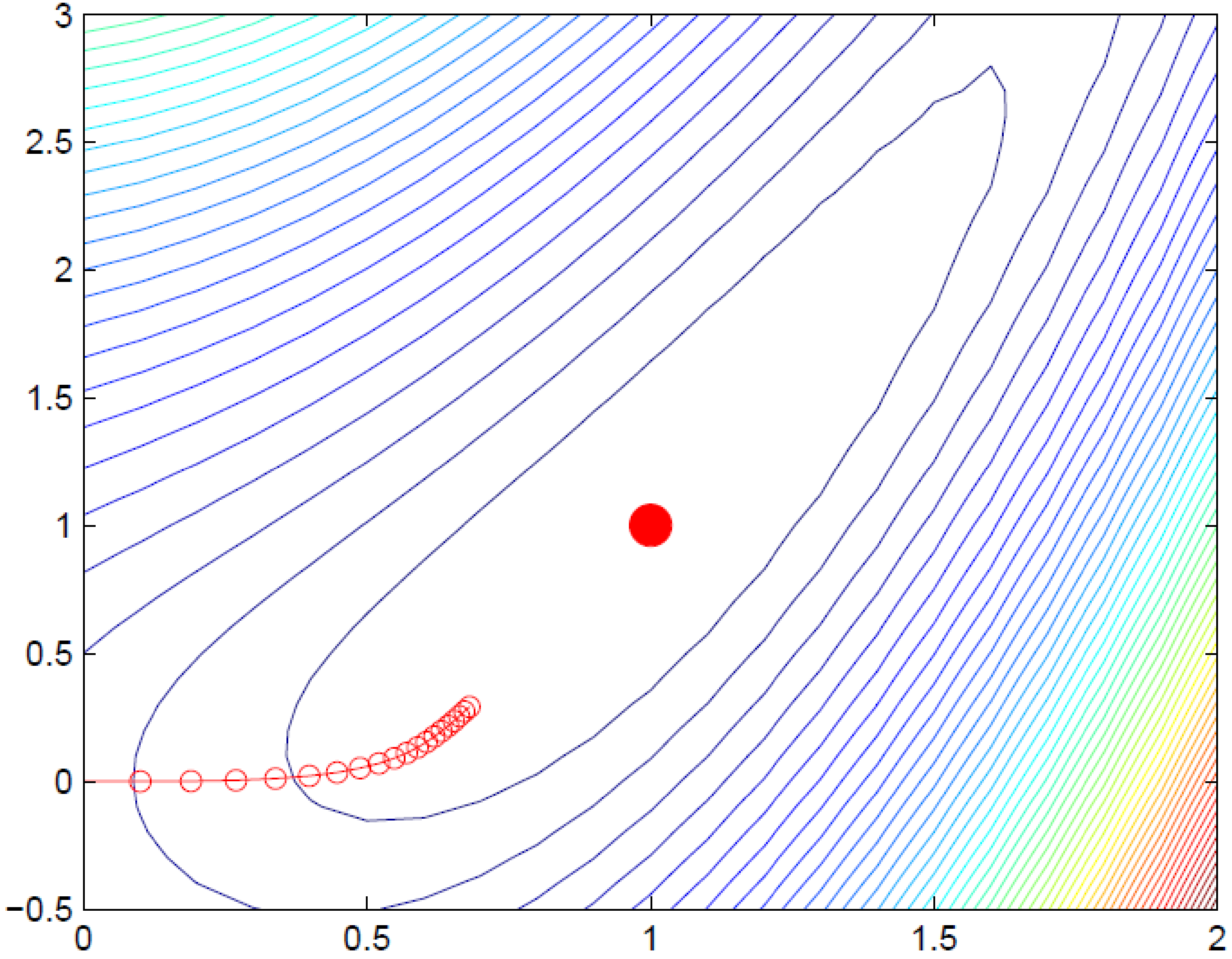
$y = \text{sigmoid}(o) \qquad y \in [0,1]^N$

$\Delta w = X^T(r - y) \qquad \Delta w \in \mathbb{R}^d \leftarrow$ negative gradient!

$w = w + \eta \Delta w$

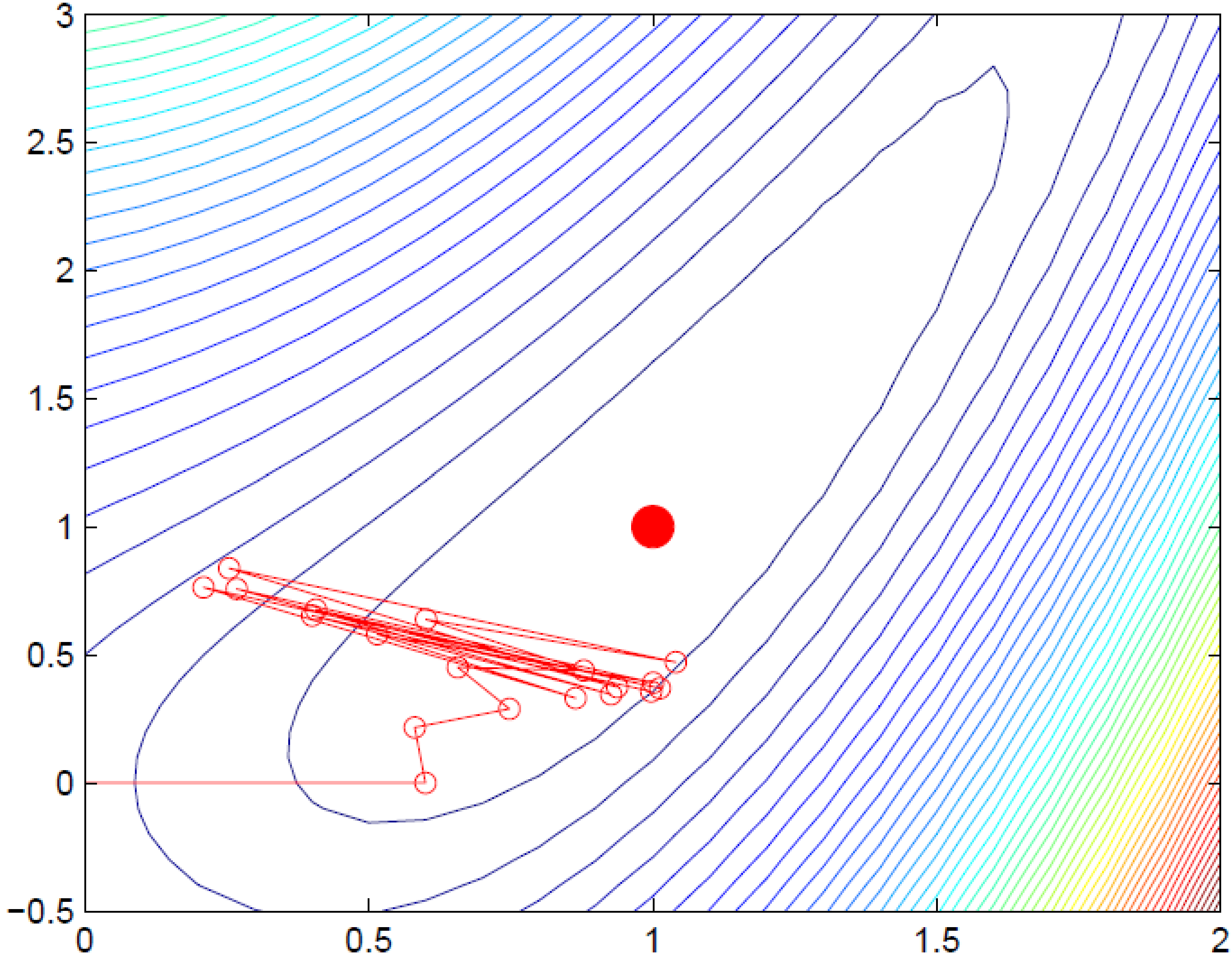
Learning for logistic regression

$\eta = 0.1$

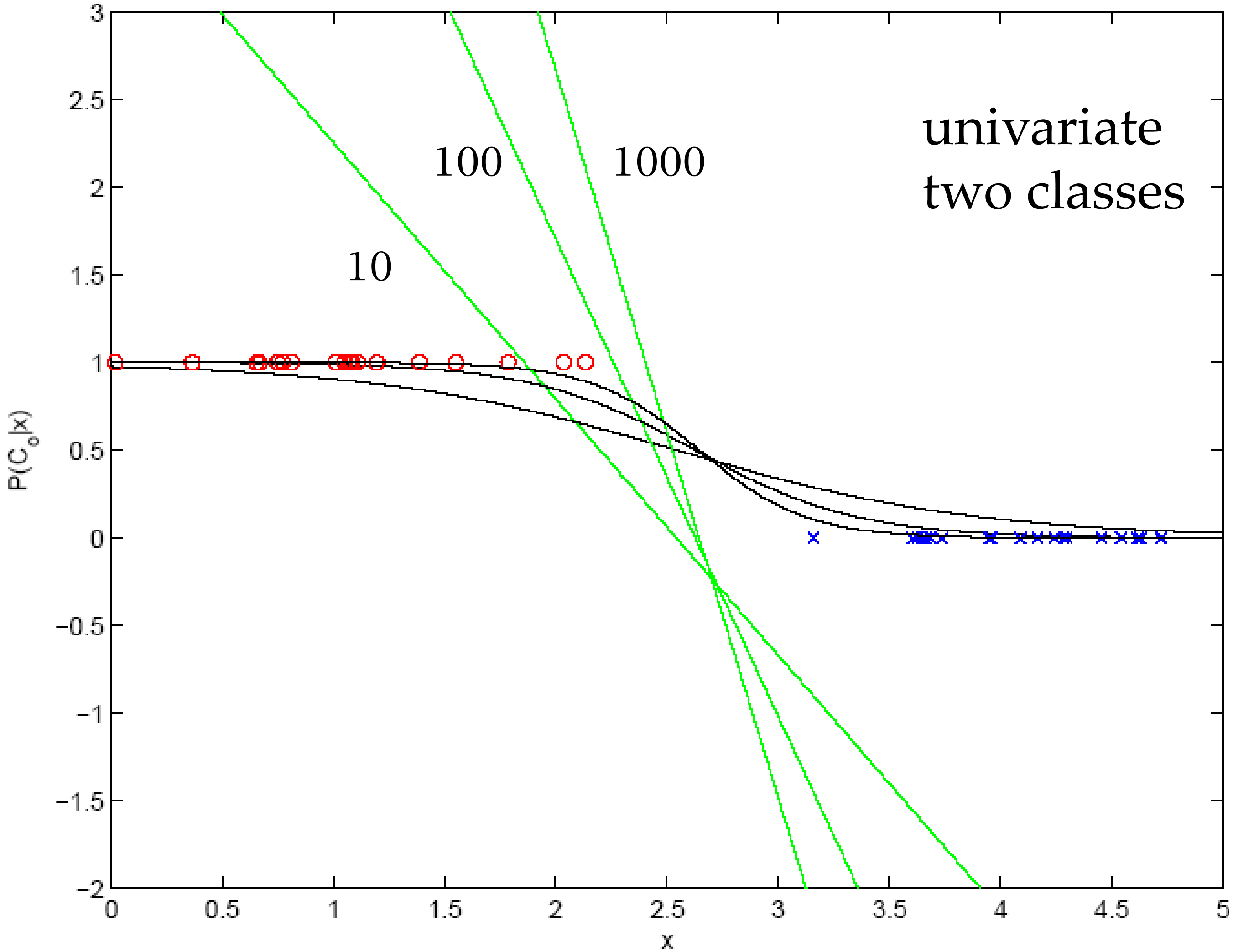


Learning for logistic regression

$\eta = 0.6$



after 10, 100, 1000 iterations



K>2 Classes

$$\mathcal{X} = \{\mathbf{x}^t, \mathbf{r}^t\}_t \quad r^t \mid \mathbf{x}^t \sim \text{Mult}_K(1, \mathbf{y}^t)$$

$$y_i = \hat{P}(C_i \mid \mathbf{x}) = \frac{\exp\left[\mathbf{w}_i^T \mathbf{x} + w_{i0}\right]}{\sum_{j=1}^{K} \exp\left[\mathbf{w}_j^T \mathbf{x} + w_{j0}\right]}, i = 1, \ldots, K$$

softmax

$$l(\{\mathbf{w}_i, w_{i0}\}_i \mid \mathcal{X}) = \prod_t \prod_i \left(y_i^t\right)^{\left(r_i^t\right)}$$

$$E(\{\mathbf{w}_i, w_{i0}\}_i \mid \mathcal{X}) = -\sum_{t,\, i} r_i^t \log y_i^t$$

$$\Delta \mathbf{w}_j = \eta \sum_t \left(r_j^t - y_j^t\right)\mathbf{x}^t \quad \Delta w_{j0} = \eta \sum_t \left(r_j^t - y_j^t\right)$$

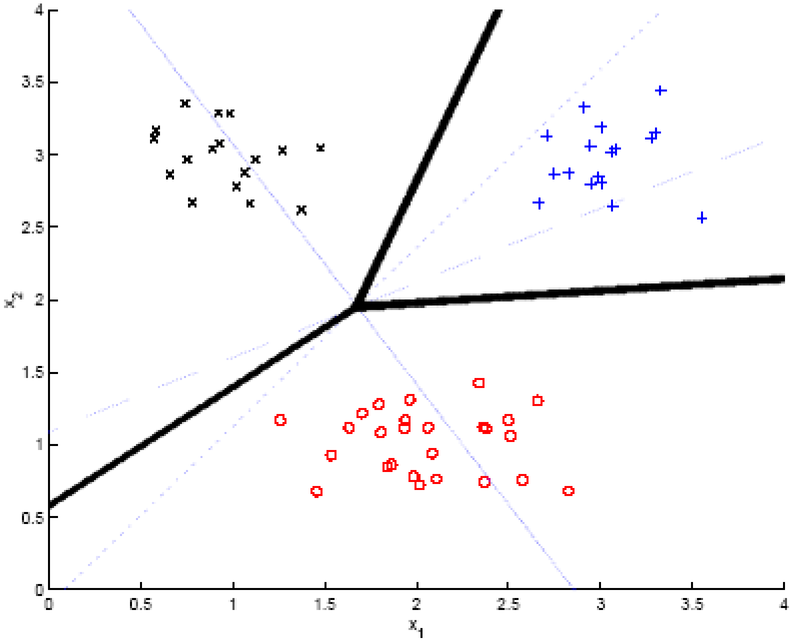
Gradient-Descent for multiple classes

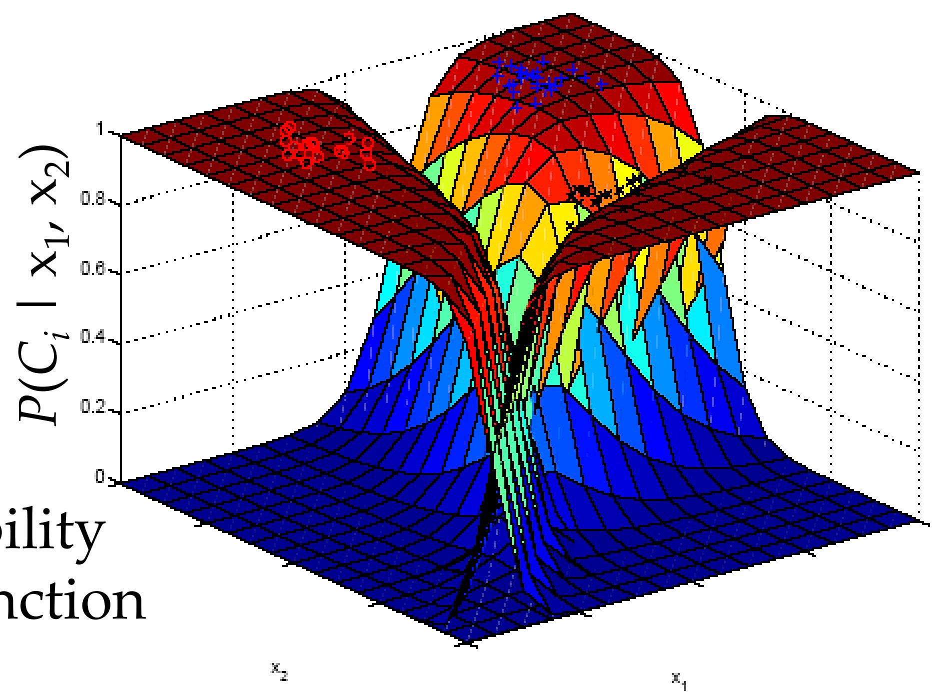
For $i = 1, \ldots, K$, For $j = 0, \ldots, d$, $w_{ij} \leftarrow \text{rand}(-0.01, 0.01)$
Repeat
  For $i = 1, \ldots, K$, For $j = 0, \ldots, d$, $\Delta w_{ij} \leftarrow 0$
  For $t = 1, \ldots, N$
    For $i = 1, \ldots, K$
      $o_i \leftarrow 0$
      For $j = 0, \ldots, d$
        $o_i \leftarrow o_i + w_{ij} x_j^t$
    For $i = 1, \ldots, K$
      $y_i \leftarrow \exp(o_i) / \sum_k \exp(o_k)$
    For $i = 1, \ldots, K$
      For $j = 0, \ldots, d$
        $\Delta w_{ij} \leftarrow \Delta w_{ij} + (r_i^t - y_i) x_j^t$
  For $i = 1, \ldots, K$
    For $j = 0, \ldots, d$
      $w_{ij} \leftarrow w_{ij} + \eta \Delta w_{ij}$
Until convergence

$$y_i = \hat{P}(C_i \mid \mathbf{x}) = \frac{\exp\left[\mathbf{w}_i^T \mathbf{x} + w_{i0}\right]}{\sum_{j=1}^K \exp\left[\mathbf{w}_j^T \mathbf{x} + w_{j0}\right]}$$

$$\Delta \mathbf{w}_j = \eta \sum_t \left(r_j^t - y_j^t\right) \mathbf{x}^t$$

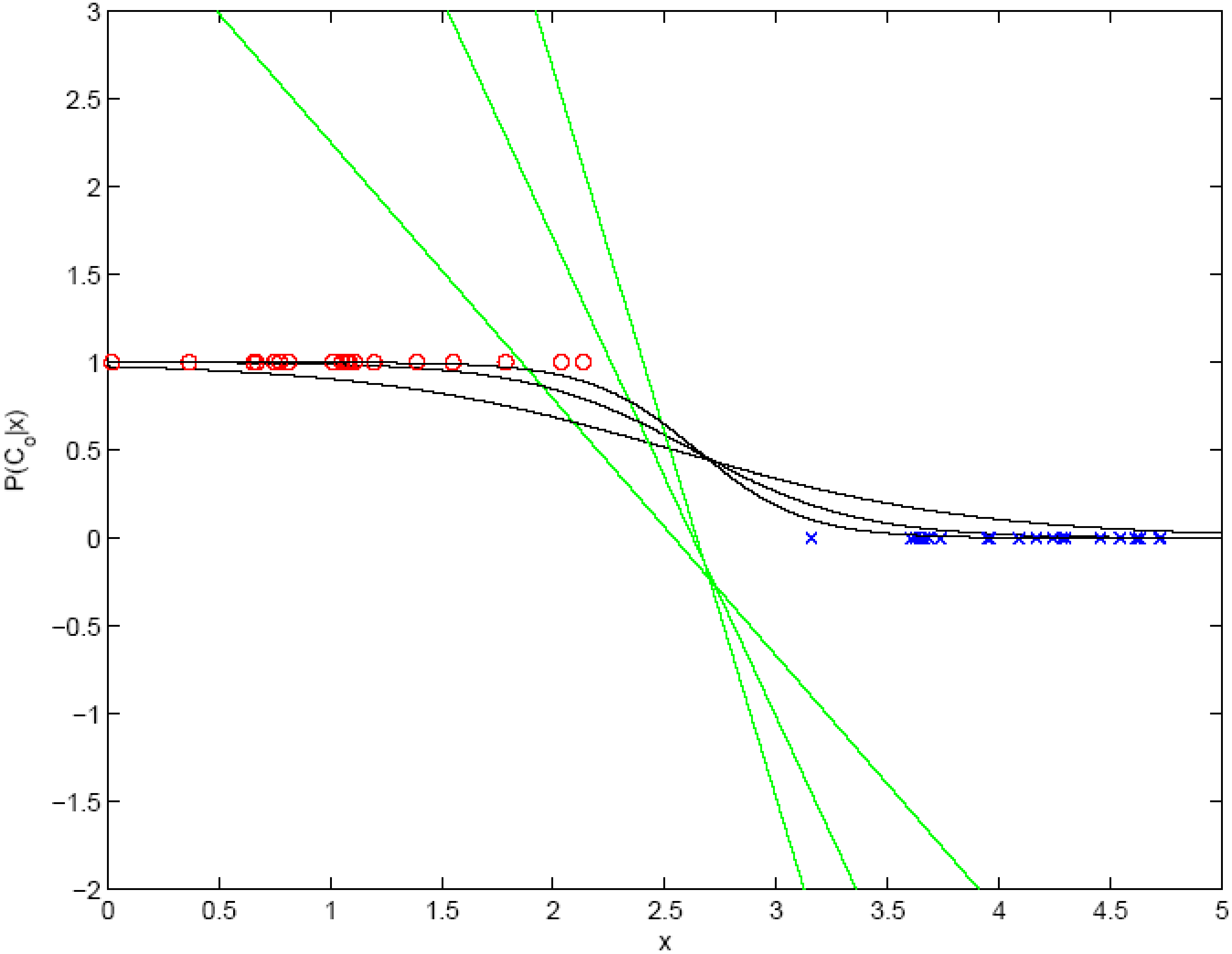
Example





posterior probability
using logistic function

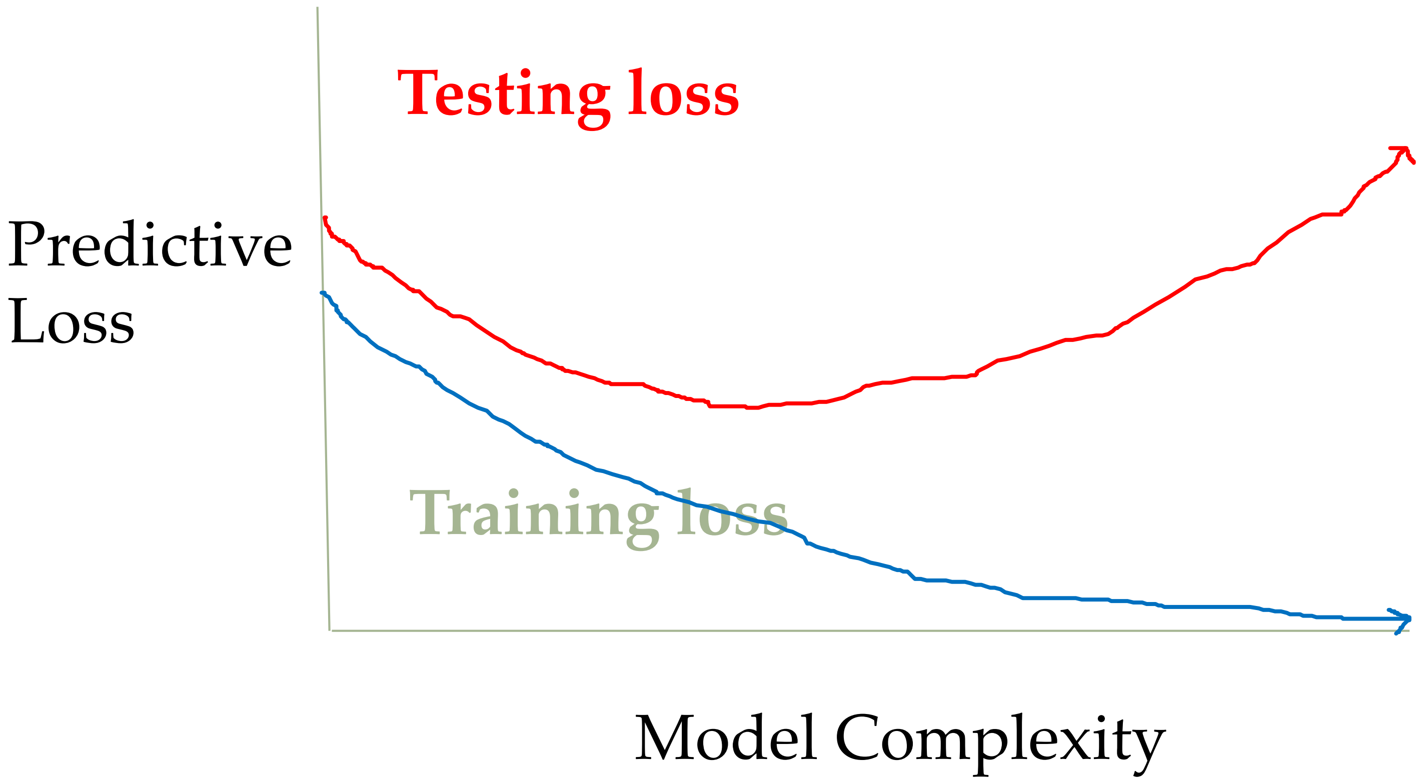
Perfectly predictable training data?



LR and overfitting

- Overfitting
  - Occurs when very few instances and feature space is high dimensional
- To avoid, a common approach is defining a prior on **w**
  - Corresponds to ***Regularization***
  - Helps with avoiding large weights
  - “Pushes” parameters to zero

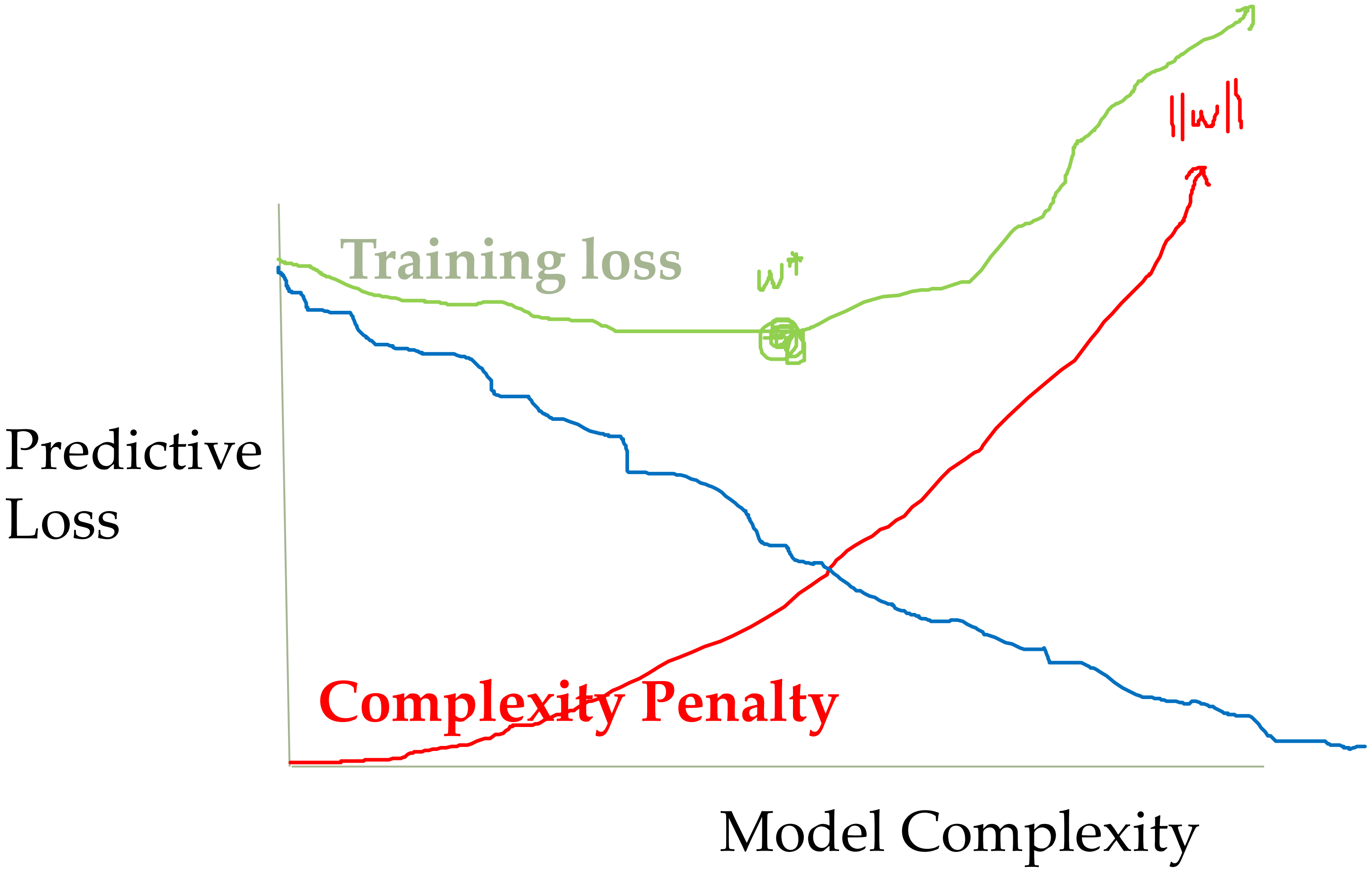
Overfitting



Need to prevent complex hypotheses

- Overfitting
  - Occurs when very few instances and feature space is high dimensional
- **Idea #1**: Restrict the number of features considered
  - Cross-validation
- **Idea #2:** Penalize complex hypotheses in the model search
  - **Regularization!**

Regularization



Regularization

- Recall the objective of logistic regression:

$$E(\mathbf{w}, w_0 \mid \mathcal{X}) = -\sum_t r^t \log y^t + (1 - r^t) \log(1 - y^t)$$

- L_2 regularization

$$\underset{\mathbf{w}}{argmin} \quad E(\mathbf{w}, w_0|X) + \lambda \sum_i w_i^2$$

- $\lambda > 0$ is a weight, chosen by, e.g., cross validation

Summary

- Generative model vs. discriminative model
- Model binary and multi-class classification
- Logistic discrimination
  - Maximum likelihood estimation
  - Gradient descent optimization
  - How to compute the gradient
- Regularization