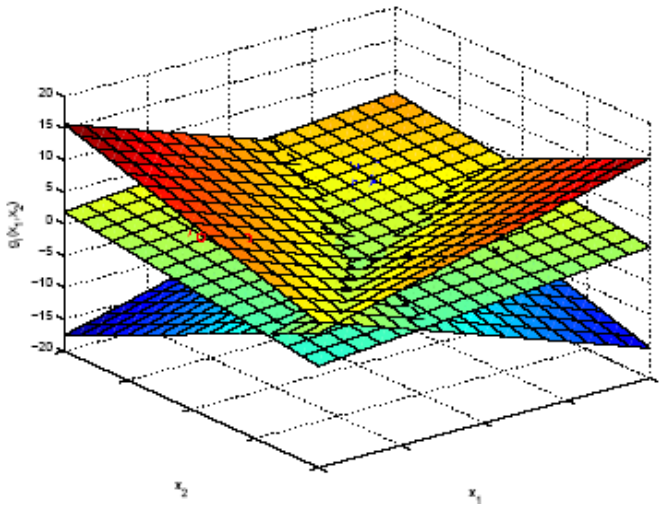
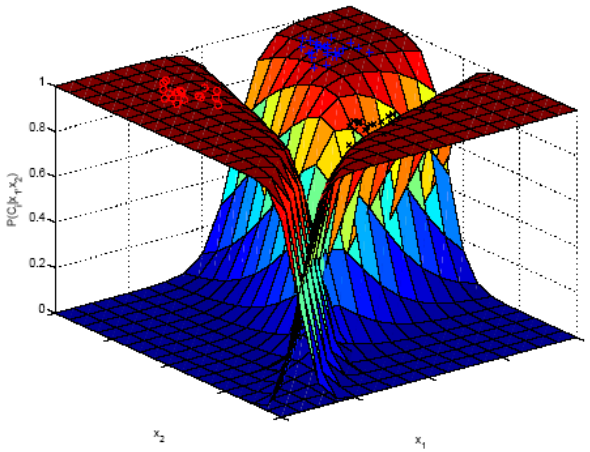
Introduction to Machine Learning





Week 2: Logistic Regression

Iasonas Kokkinos

i.kokkinos@cs.ucl.ac.uk

University College London

Feedback from Monday's lab

- Bring your laptop with you
- Keep it simple
  - "introduction to jupyter": not the main point of the assignment
- Lightweight assignment: getting started for the next one.

Week 1: Machine Learning variants

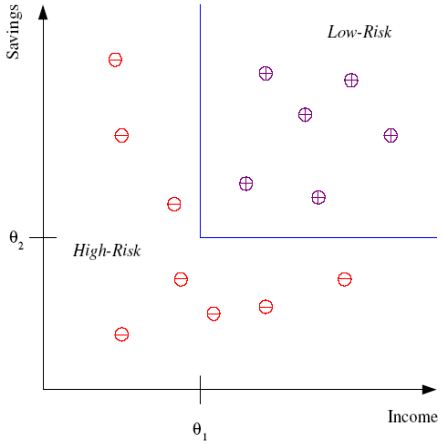
- Supervised
  - Classification
  - Regression
- Unsupervised
  - Clustering
  - Dimensionality Reduction
- Weakly supervised/semi-supervised

  Some data supervised, some unsupervised
- Reinforcement learning

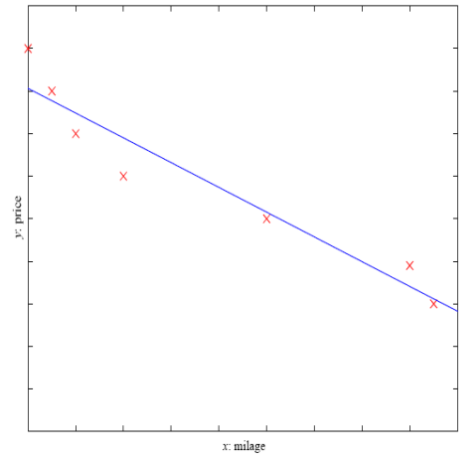
  Supervision: sparse reward for a sequence of decisions

What we want to learn: a function

Classification: $y \in \{0, 1\}$



Regression: $y \in \mathbb{R}$



prediction — **method**

$$y = f_w(x)$$

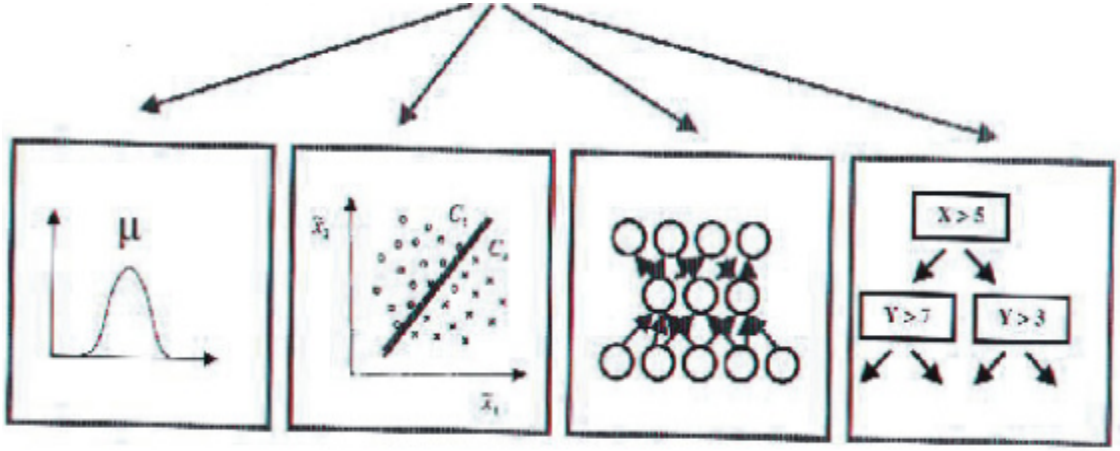
parameters — **Input**

What we want to learn: a function

prediction

method

$$y = f_w(x)$$

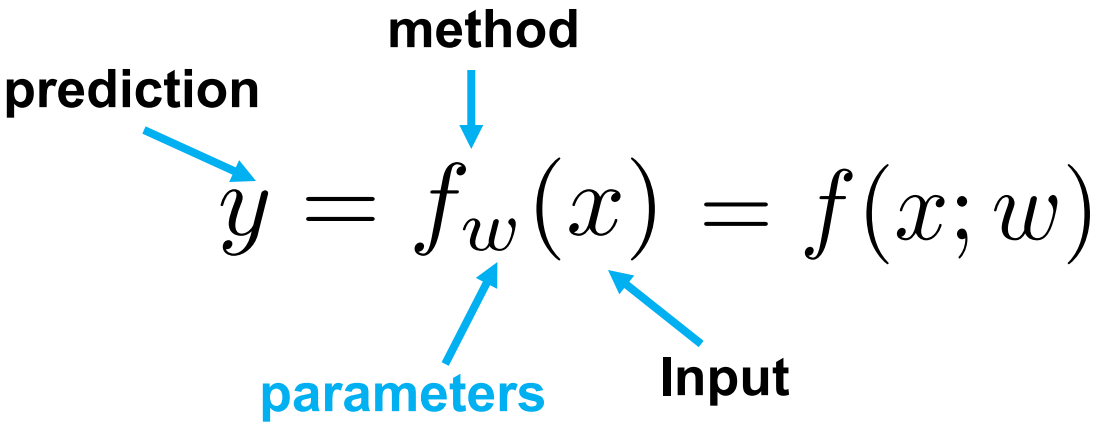


Week 1-4: linear models

$$y = f_{\mathbf{w}}(\mathbf{x}) = f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T\mathbf{x}$$

Linear classifiers, neural networks, decision trees, ensemble models, probabilistic classifiers, …

What we want to learn: a function



Sum-of-squared errors loss:

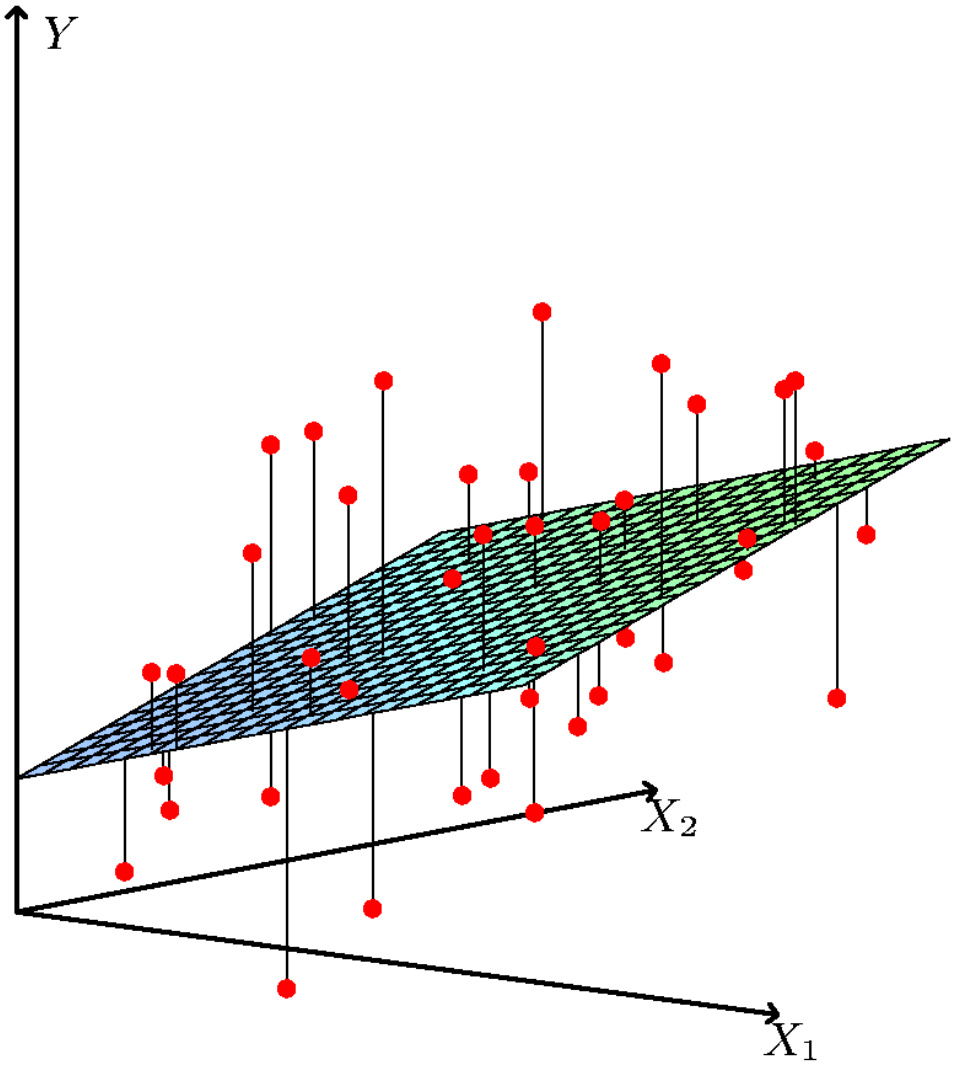
$$L(\mathbf{w}) = \sum_{i=1}^{N} (y^i - \langle \mathbf{w}, \mathbf{x}^i \rangle)^2$$

$$\mathbf{y} = \begin{bmatrix} y^1 \\ \vdots \\ y^N \end{bmatrix} \mathbf{X} = \begin{bmatrix} (\mathbf{x}^1)^T \\ \hline (\mathbf{x}^2)^T \\ \hline \vdots \\ \hline (\mathbf{x}^N)^T \end{bmatrix}$$

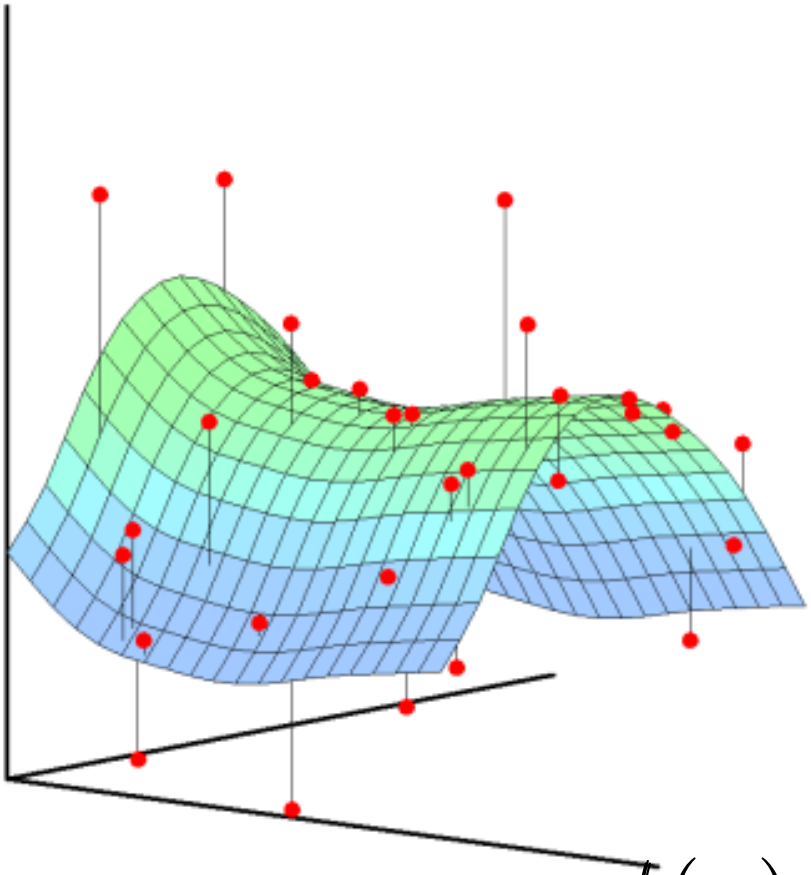
Least squares estimate:

$$\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} L(\mathbf{w}) = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

Linear regression



Generalized linear regression



$$\mathbf{x} \rightarrow \boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} \phi_1(\mathbf{x}) \\ \vdots \\ \phi_M(\mathbf{x}) \end{bmatrix}$$

Ridge regression & cross-validation

New objective:

$$L(\mathbf{w}) = \boldsymbol{\epsilon}^T\boldsymbol{\epsilon} + \lambda\mathbf{w}^T\mathbf{w}$$

"data fidelity" complexity

Setting w:

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

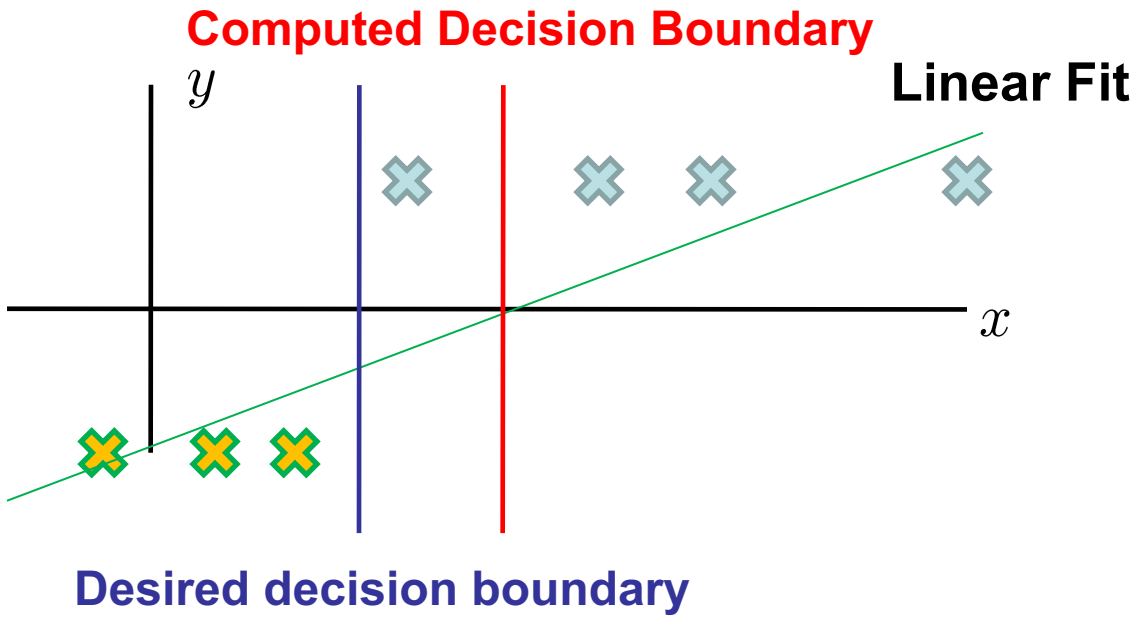
Setting λ: cross-validation

Inappropriateness of quadratic loss

We chose the quadratic cost function for convenience

Single, global minimum & closed form expression

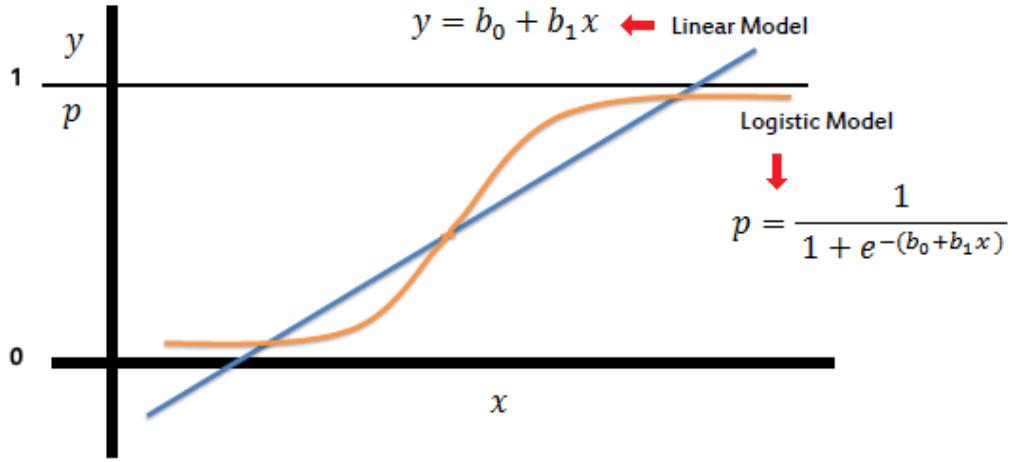
But does it indicate classification performance?

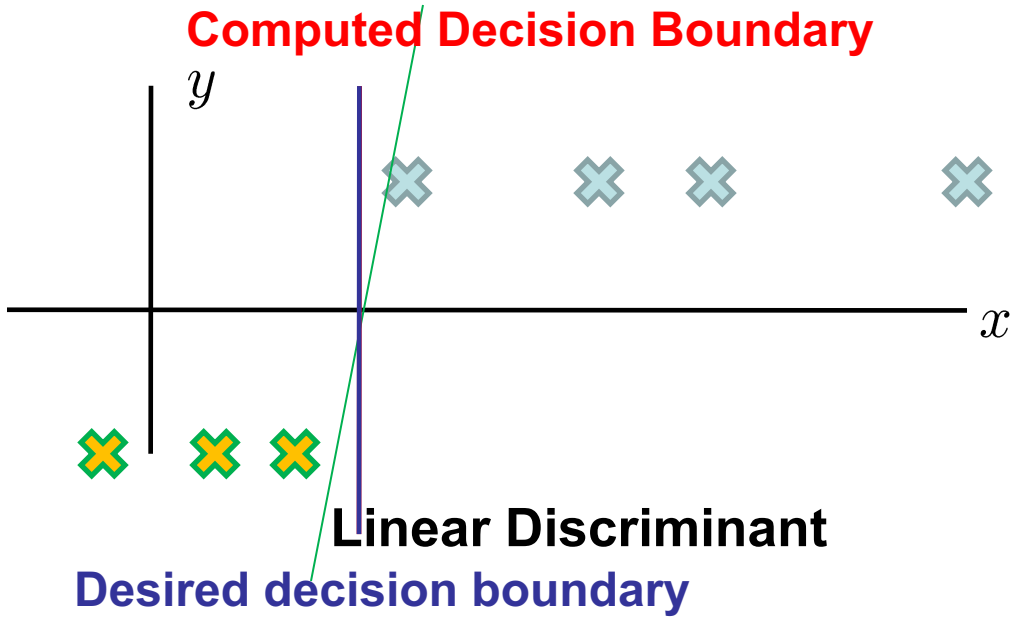


Today's basic idea

Use squashing function

The higher, the better



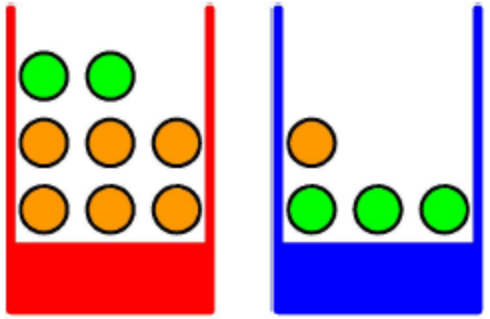


Our goal today: make this precise

Probability refresher

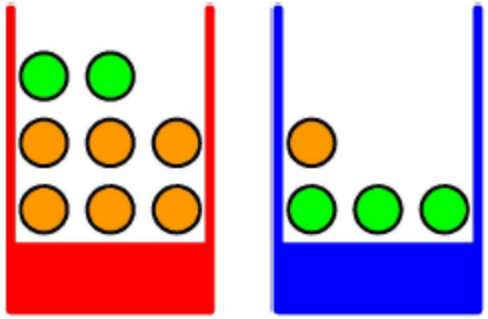
Probability Review -I

- Example: apples and oranges
  - We have two boxes to pick from.
  - Each box contains both types of fruit.
  - What is the probability of picking an apple?



Probability Review -I

- Example: apples and oranges
  - We have two boxes to pick from.
  - Each box contains both types of fruit.
  - What is the probability of picking an apple?



- Formalization
  - Let $B \in \{r, b\}$ be a random variable for the box we pick.
  - Let $F \in \{a, o\}$ be a random variable for the type of fruit we get.
  - Suppose we pick the red box 40% of the time. We write this as

$$p(B = r) = 0.4 \qquad p(B = b) = 0.6$$

  - The probability of picking an apple *given* a choice for the box is

$$p(F = a \mid B = r) = 0.25 \qquad p(F = a \mid B = b) = 0.75$$

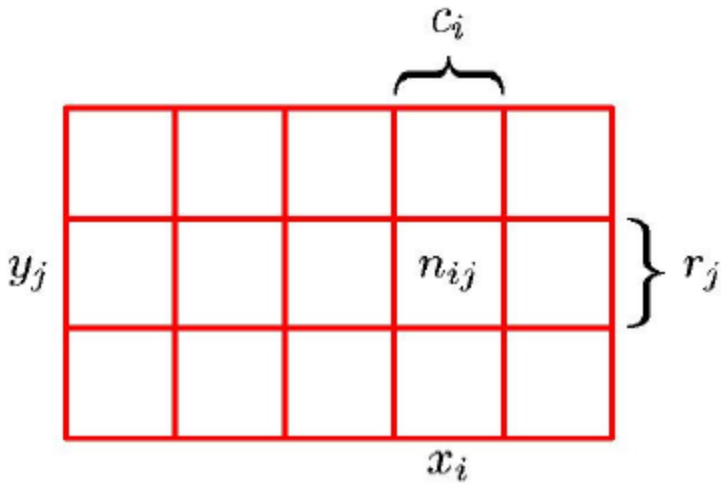
  - What is the probability of picking an apple?

$$p(F = a) = ?$$

Joint, Marginal, Conditional Probability

- More general case
  - Consider two random variables $X \in \{x_i\}$ and $Y \in \{y_j\}$
  - Consider N trials and let

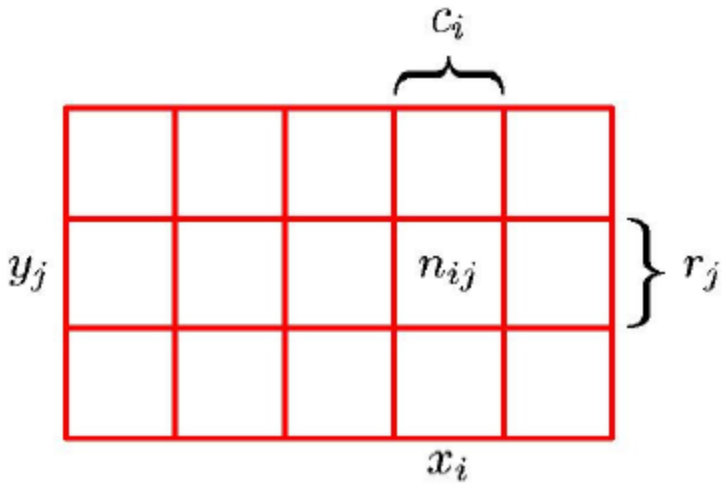
$$n_{ij} = \#\{X = x_i \wedge Y = y_j\}$$
$$c_i = \#\{X = x_i\}$$
$$r_j = \#\{Y = y_j\}$$



Joint, Marginal, Conditional Probability

- More general case
  - Consider two random variables $X \in \{x_i\}$ and $Y \in \{y_j\}$
  - Consider N trials and let

$$n_{ij} = \#\{X = x_i \wedge Y = y_j\}$$
$$c_i = \#\{X = x_i\}$$
$$r_j = \#\{Y = y_j\}$$

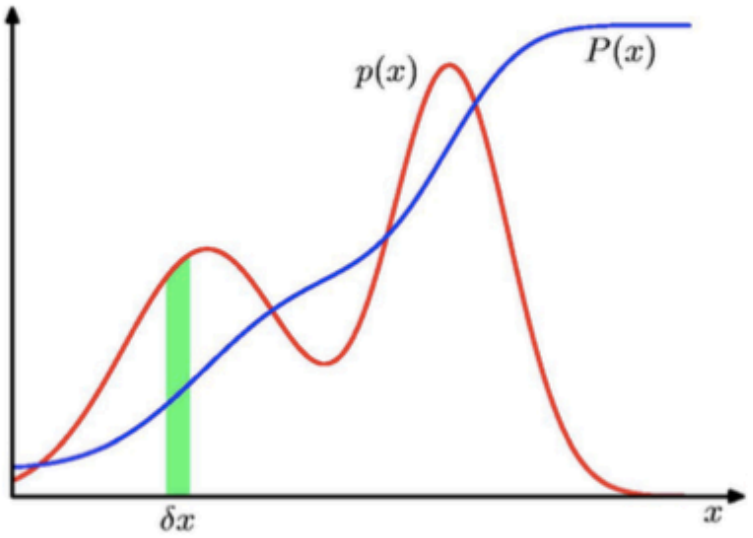


- Then we can derive
  - Joint probability $$p(X = x_i, Y = y_j) = \frac{n_{ij}}{N}$$
  - Marginal probability $$p(X = x_i) = \frac{c_i}{N}.$$
  - Conditional probability $$p(Y = y_j | X = x_i) = \frac{n_{ij}}{c_i}$$

Continuous variables

- **Probabilities over continuous variables are defined over their probability density function (pdf)** $p(x)$**.**

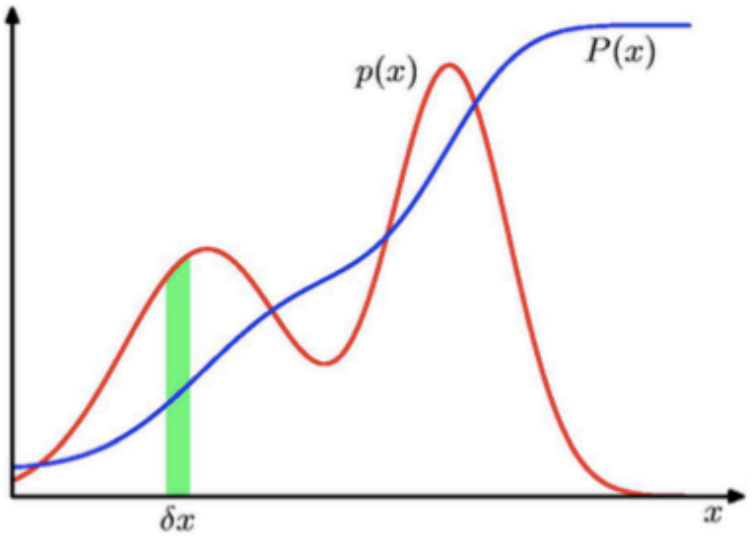
$$p(x \in (a, b)) = \int_a^b p(x)\,\mathrm{d}x$$



Continuous variables

- Probabilities over continuous variables are defined over their probability density function (pdf) $p(x)$.

$$p(x \in (a, b)) = \int_a^b p(x)\,\mathrm{d}x$$



- The probability that x lies in the interval $(-\infty, z)$ is given by the cumulative distribution function

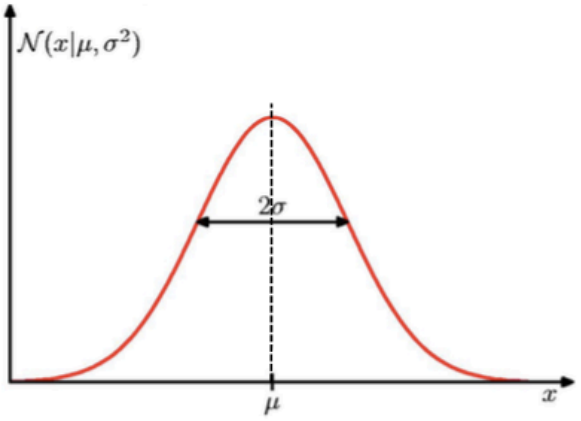
$$P(z) = \int_{-\infty}^{z} p(x)\,\mathrm{d}x$$

Gaussian (or Normal) distribution

One-dimensional case

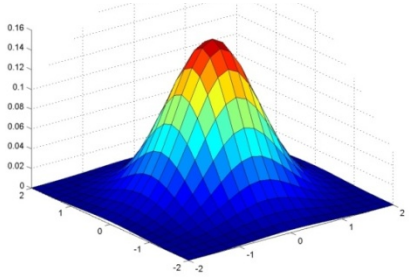
- Mean μ
- Variance σ^2

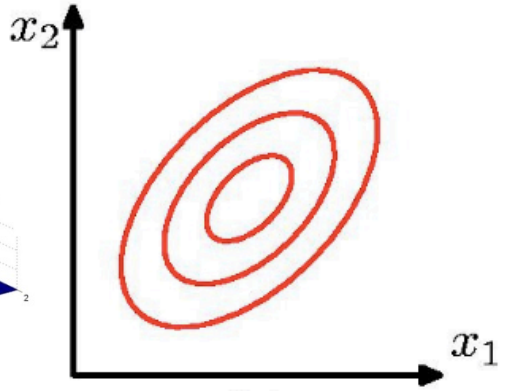
$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}$$



Multi-dimensional case

- Mean $\boldsymbol{\mu}$
- Covariance $\boldsymbol{\Sigma}$



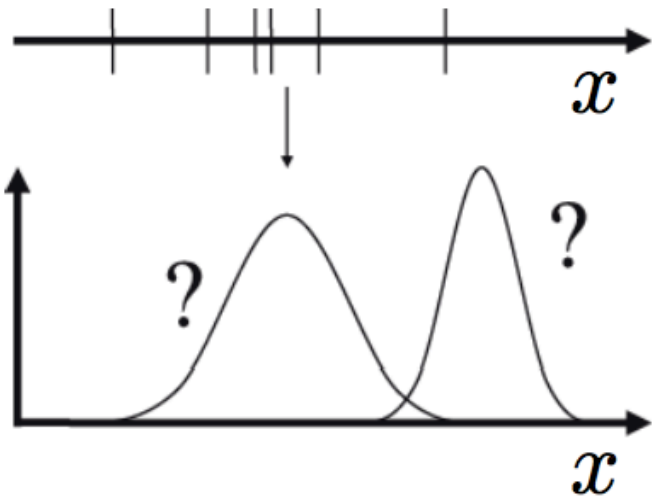


$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

Parameter estimation

Given

- Data $X = \{x_1, x_2, \ldots, x_N\}$
- Parametric form of the distribution with parameters θ
- E.g. for Gaussian distrib.: $\theta = (\mu, \sigma)$



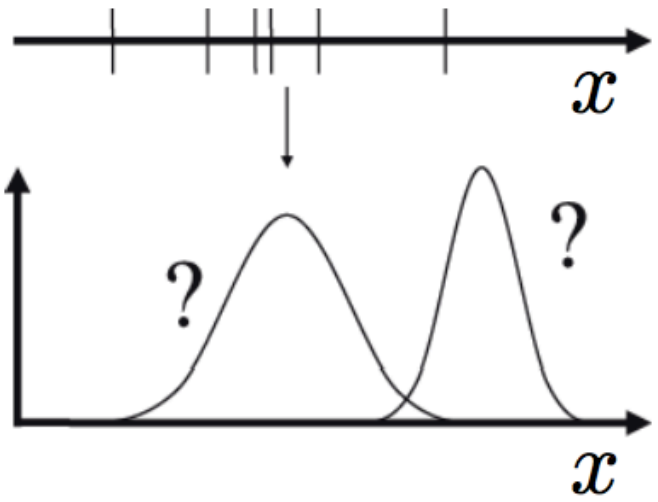
Learning

- Estimation of the parameters θ

Parameter estimation

Given

- Data $X = \{x_1, x_2, \ldots, x_N\}$
- Parametric form of the distribution with parameters θ
- E.g. for Gaussian distrib.: $\theta = (\mu, \sigma)$



Learning

- Estimation of the parameters θ

Likelihood of θ

- Probability that the data X have indeed been generated from a probability density with parameters θ

$$L(\theta) = p(X|\theta)$$

Computation of the likelihood

- Single data point: $p(x_n|\theta)$

Computation of the likelihood

- Single data point: $p(x_n|\theta)$

- Assumption: all data points are independent

$$L(\theta) = p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta)$$

Computation of the likelihood

- Single data point: $p(x_n|\theta)$

- Assumption: all data points are independent

$$L(\theta) = p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta)$$

- Negative (of) log-likelihood

$$E(\theta) = -\ln L(\theta) = -\sum_{n=1}^{N} \ln p(x_n|\theta)$$

Computation of the likelihood

- Single data point: $p(x_n|\theta)$

- Assumption: all data points are independent

$$L(\theta) = p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta)$$

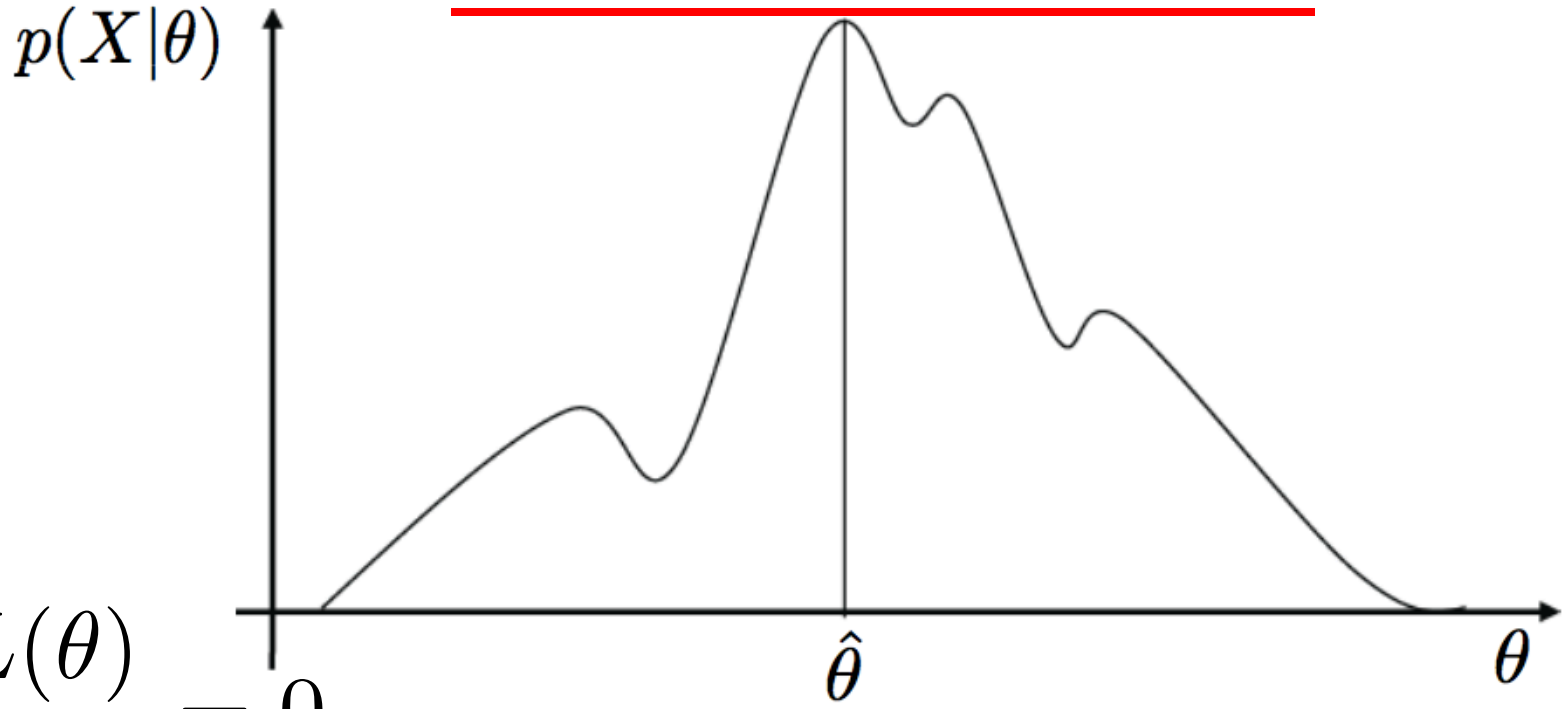
- Negative (of) log-likelihood

$$E(\theta) = -\ln L(\theta) = -\sum_{n=1}^{N} \ln p(x_n|\theta)$$

- Estimation of the parameters θ (Learning)
  - Maximize the likelihood
  - Minimize the negative log-likelihood

Likelihood: $L(\theta) = p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta)$

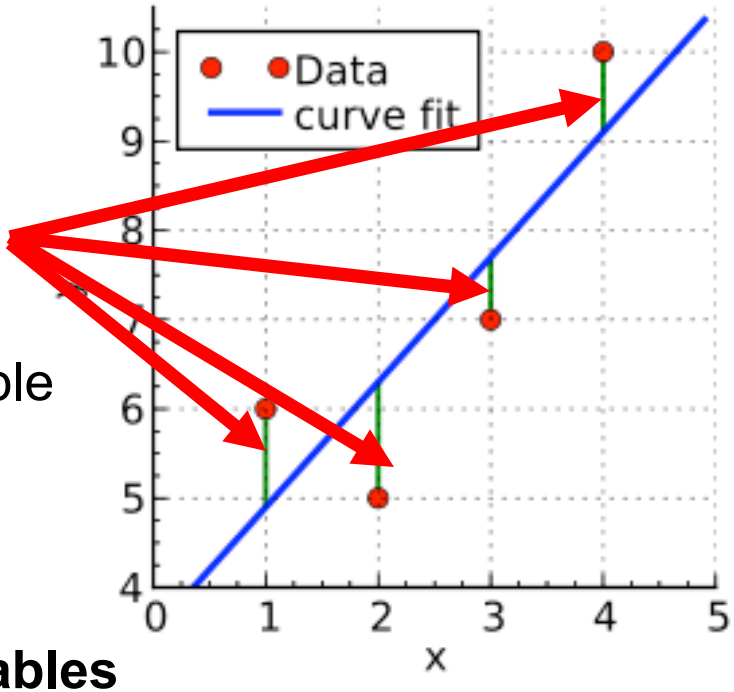
We want to obtain $\hat{\theta}$ such that $L(\hat{\theta})$ is maximized.



$$\frac{\partial L(\theta)}{\partial \theta} = 0$$

Probabilistic formulation of linear regression-I

$$\epsilon^i = y^i - \mathbf{w}^T \mathbf{x}^i$$



Residual: zero-mean Gaussian random variable

$$p(E^i = \epsilon^i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\epsilon^i)^2}{2\sigma^2}\right)$$

The R.V. its value

random variables

$$Y^i = \mathbf{w}^T \mathbf{x}^i + E^i$$

deterministic

Conditional model on observations:

$$p(Y^i = y^i | \mathbf{x}^i; \mathbf{w}^T) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^i - \mathbf{w}^T \mathbf{x}^i)^2}{2\sigma^2}\right)$$

Probabilistic interpretation of linear regression

Training set: $\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}, \mathbf{x} \in \mathbb{R}^D, y \in \mathbb{R}$

$$p(y^1, \ldots, y^n | \mathbf{x}^1, \ldots, \mathbf{x}^N) =$$

Independence assumption

$$= \prod_{i=1}^{N} p(y^i | \mathbf{x}^i)$$

$$= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^i - \mathbf{w}^T \mathbf{x}^i)^2}{2\sigma^2}\right)$$

$$= \frac{1}{(\sqrt{2\pi}\sigma)^N} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{N} (y^i - \mathbf{w}^T \mathbf{x}^i)^2\right)$$

=> Least squares: Maximum Conditional Likelihood estimation of w

Great, but only for real-valued outputs!

Training set: $\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}, \mathbf{x} \in \mathbb{R}^D, y \in \mathbb{R}$

$$p(y^1, \ldots, y^n | \mathbf{x}^1, \ldots, \mathbf{x}^N) =$$

Independence assumption

$$= \prod_{i=1}^{N} p(y^i | \mathbf{x}^i)$$

$$= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^i - \mathbf{w}^T \mathbf{x}^i)^2}{2\sigma^2}\right)$$

$$= \frac{1}{(\sqrt{2\pi}\sigma)^N} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{N} (y^i - \mathbf{w}^T \mathbf{x}^i)^2\right)$$

From regression to classification

Training set: $\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}, \mathbf{x} \in \mathbb{R}^D, y \in \{0, 1\}$

$$p(y^1, \ldots, y^n | \mathbf{x}^1, \ldots, \mathbf{x}^N) =$$

Independence assumption

$$= \prod_{i=1}^{N} p(y^i | \mathbf{x}^i)$$

?

From regression to classification

Training set: $\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}, \mathbf{x} \in \mathbb{R}^D, y \in \{0, 1\}$

$$p(y^1, \ldots, y^n | \mathbf{x}^1, \ldots, \mathbf{x}^N) =$$

Independence assumption

$$= \prod_{i=1}^{N} p(y^i | \mathbf{x}^i)$$

?

Bernoulli distribution

Discrete random variable $Y \in \{0, 1\}$

1x2 table, 1 parameter:

$$P(Y = 1) = p$$
$$P(Y = 0) = 1 - P(Y = 1) = 1 - p$$

Compact form:

$$P(Y = c) = \begin{cases} p & c = 1 \\ 1 - p & c = 0 \end{cases}$$
$$= p^c (1 - p)^{1-c}$$

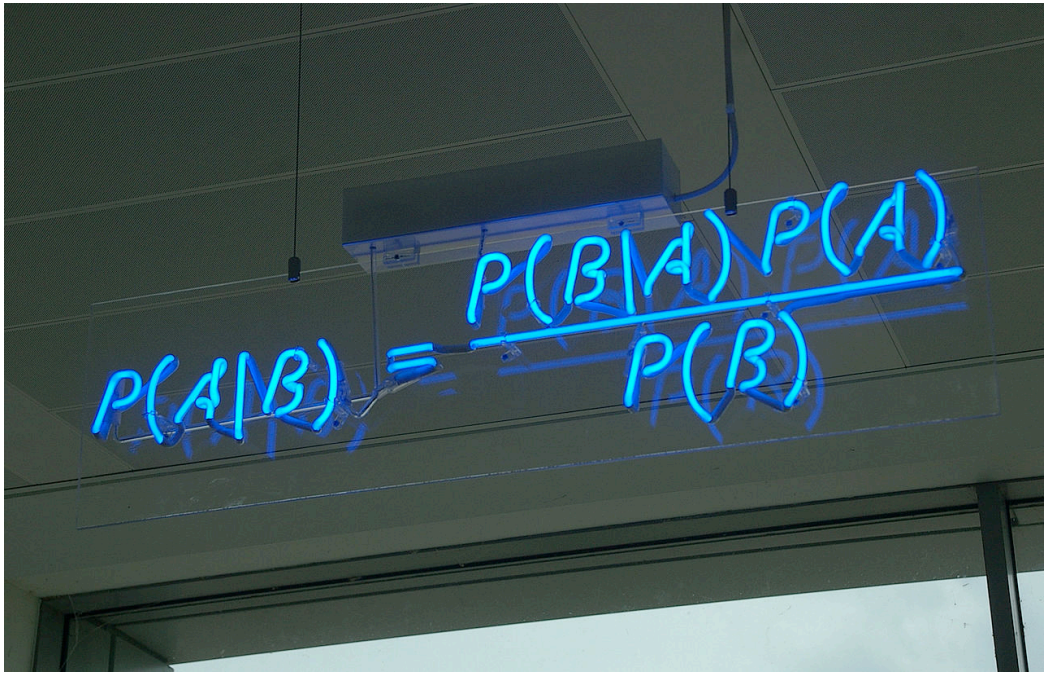
Parametric model for posterior

$$P(Y = 1|X = \mathbf{x}; \mathbf{w}) = f(\mathbf{x}, \mathbf{w})$$
$$P(Y = 0|X = \mathbf{x}; \mathbf{w}) = 1 - f(\mathbf{x}, \mathbf{w})$$

$$P(Y = y|X = \mathbf{x}; \mathbf{w}) = f(\mathbf{x}, \mathbf{w})^y (1 - f(\mathbf{x}, \mathbf{w}))^{1-y}$$

What would be a reasonable expression for f?

Bayes' rule



Sum and product rule

Sum Rule $$p(X) = \sum_{Y} p(X, Y)$$

Product Rule $$p(X, Y) = p(Y|X)p(X)$$

Bayes' theorem

$$P(A = a, B = b) = P(A = a, B = b)$$

Product rule:

$$P(A = a|B = b)P(B = b) = P(B = b|A = a)P(A = a)$$

$$P(A = a|B = b) = \frac{P(B=b|A=a)P(A=a)}{P(B=b)}$$

Sum rule:

$$P(A = a|B = b) = \frac{P(B = b|A = a)P(A = a)}{\sum_{a'} P(B = b, A = a')}$$

Product rule:

$$P(A = a|B = b) = \frac{P(B = b|A = a)P(A = a)}{\sum_{a'} P(B = b|A = a')P(A = a')}$$

Bayes' theorem

Sum Rule $$p(X) = \sum_Y p(X, Y)$$

Product Rule $$p(X, Y) = p(Y|X)p(X)$$

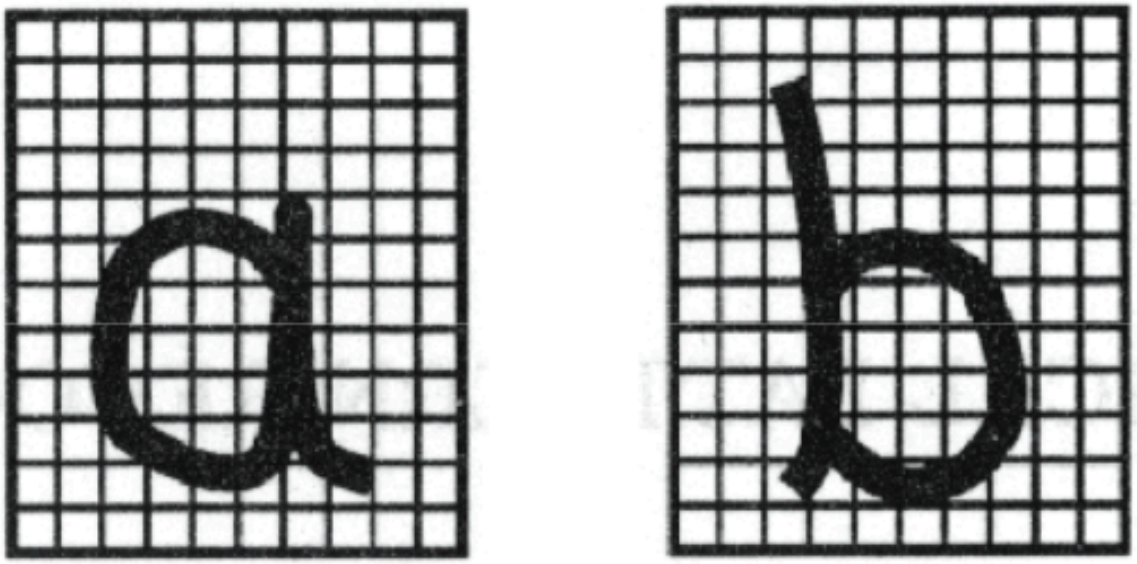
- From those, we can derive

Bayes' Theorem $$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

where $$p(X) = \sum_Y p(X|Y)p(Y)$$

Binary classification problem

- Example: handwritten character recognition



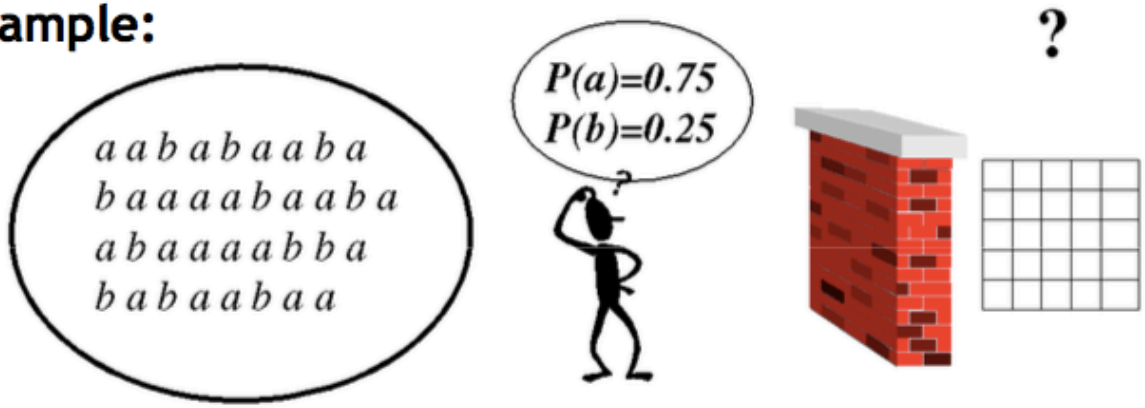
- Goal:
  - Classify a new letter such that the probability of misclassification is minimized.

Binary classification problem

- Concept 1: Priors (a priori probabilities) $p(C_k)$
  - What we can tell about the probability *before seeing the data*.

Binary classification problem

- **Concept 1: Priors (a priori probabilities)** $p(C_k)$
  - What we can tell about the probability *before seeing the data*.
  - Example:

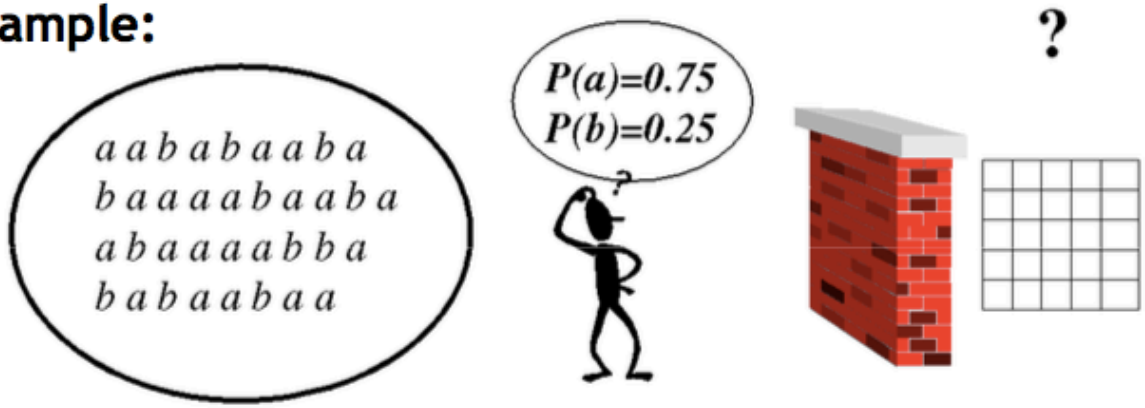


$$C_1 = a \qquad p(C_1) = 0.75$$

$$C_2 = b \qquad p(C_2) = 0.25$$

Binary classification problem

- **Concept 1: Priors (a priori probabilities)** $p(C_k)$
  - What we can tell about the probability *before seeing the data*.
  - Example:



$$C_1 = a \qquad p(C_1) = 0.75$$

$$C_2 = b \qquad p(C_2) = 0.25$$

- **In general:** $\sum_k p(C_k) = 1$

Binary classification problem

Concept 2: Conditional probabilities

$$p(x \mid C_k)$$

Binary classification problem

Concept 2: Conditional probabilities

$$p(x \mid C_k)$$

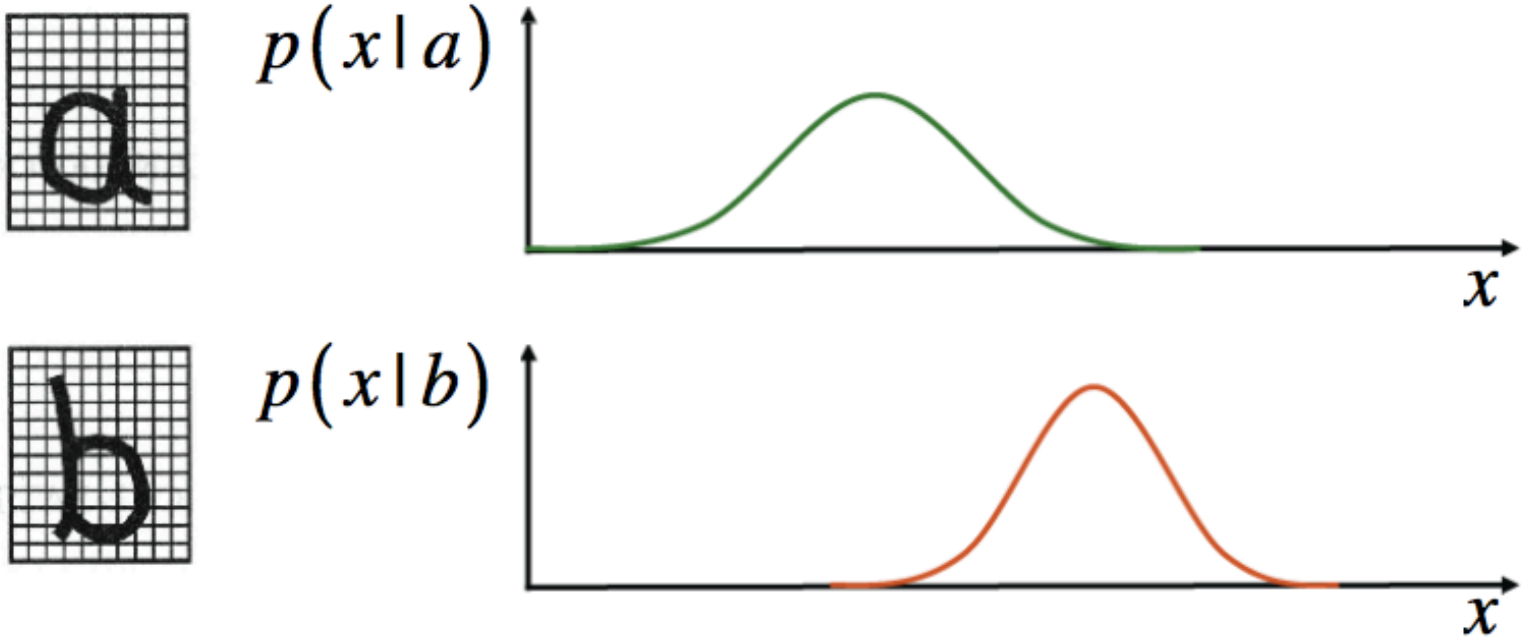
- Let x be a feature vector.
- x measures/describes certain properties of the input.
  - E.g. number of black pixels, aspect ratio, ...
- $p(x|C_k)$ describes its likelihood for class C_k.

Binary classification problem

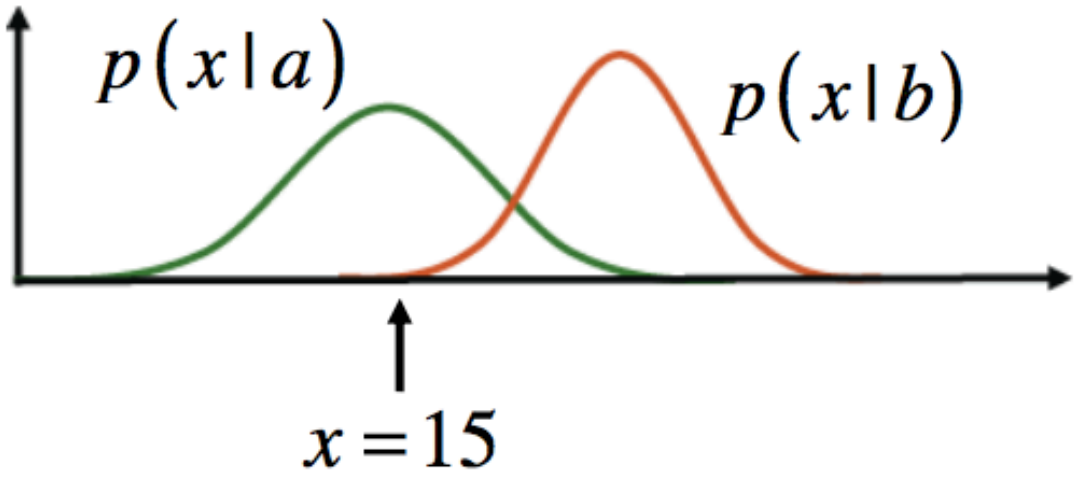
Concept 2: Conditional probabilities

$p(x \mid C_k)$

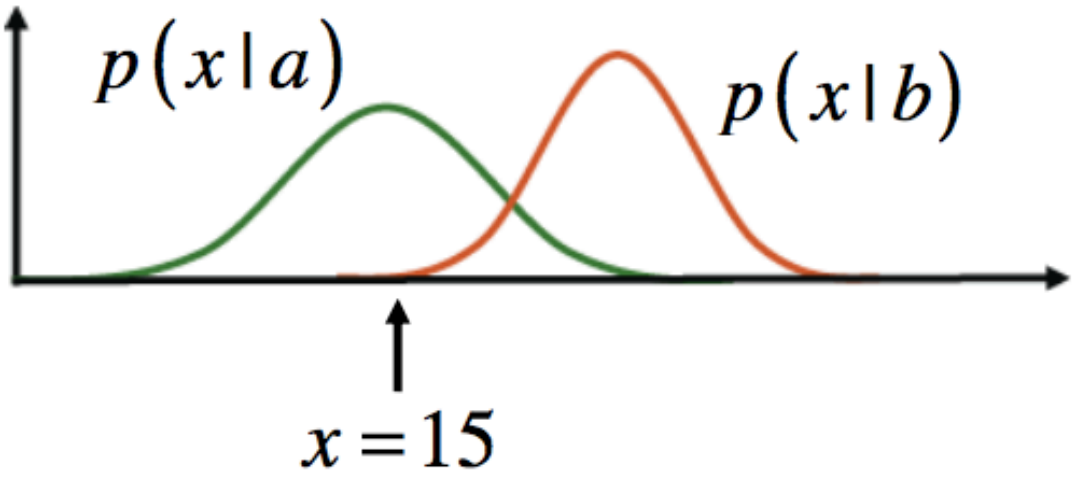
- Let x be a feature vector.
- x measures/describes certain properties of the input.
  - E.g. number of black pixels, aspect ratio, ...
- $p(x|C_k)$ describes its likelihood for class C_k.



Example:



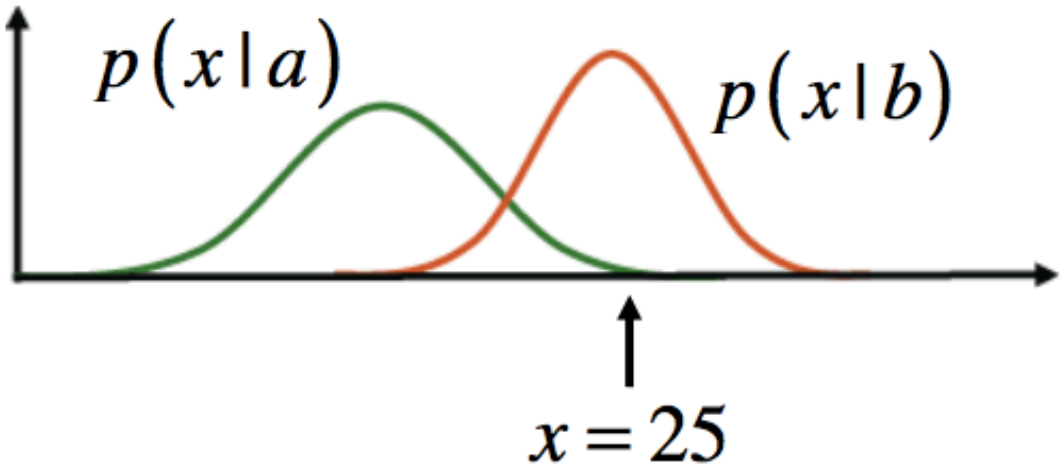
Example:



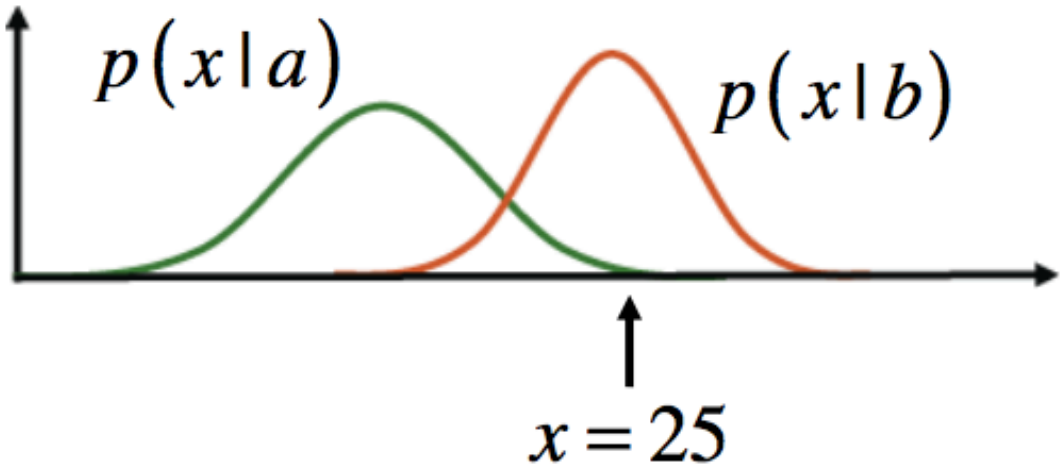
Question:

- Which class?
- The decision should be ‘a’ here.

Example:



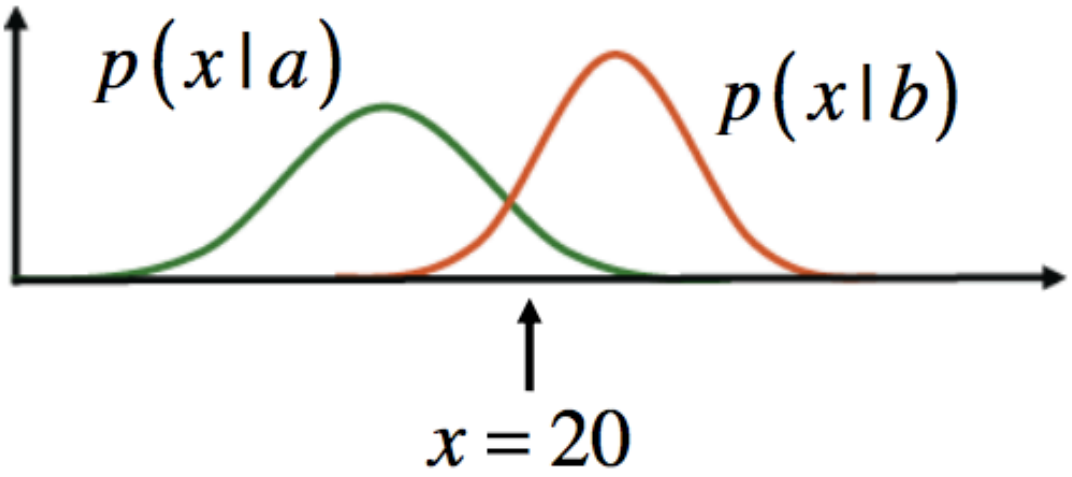
Example:



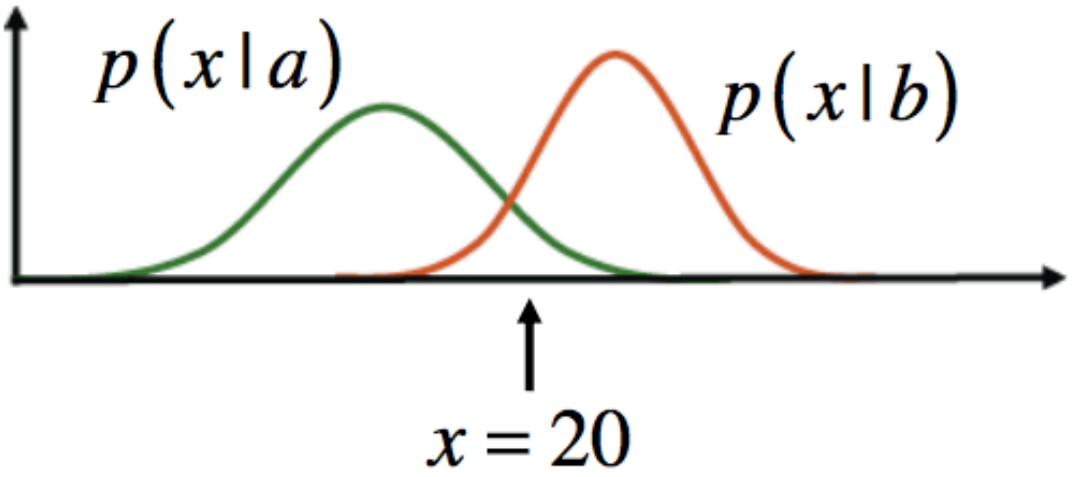
Question:

- Which class?
- Since $p(x \mid a)$ is much smaller than $p(x \mid b)$, the decision should be 'b' here.

Example:



Example:



Question:

- Which class?
- Remember that $p(a) = 0.75$ and $p(b) = 0.25$...
- I.e., the decision should be again ‘a’.

$\Rightarrow$ How can we formalize this?

Concept 3: Posterior probabilities

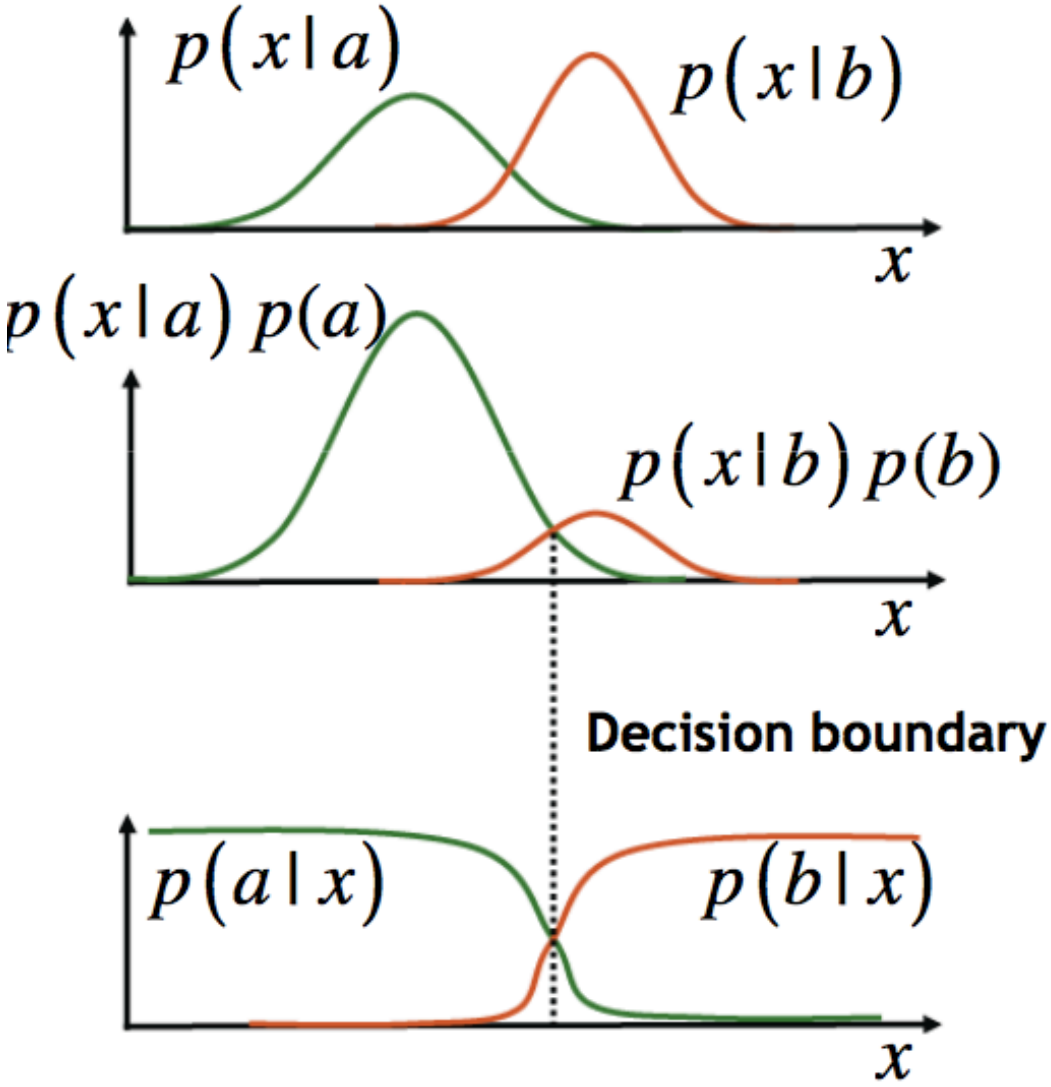
$p(C_k \mid x)$

- We are typically interested in the *a posteriori* probability, i.e. the probability of class C_k given the measurement vector x.

Bayes' Theorem:

$$p(C_k \mid x) = \frac{p(x \mid C_k)\, p(C_k)}{p(x)} = \frac{p(x \mid C_k)\, p(C_k)}{\sum_i p(x \mid C_i)\, p(C_i)}$$

Bayes' rule for binary classification problem



Probability of observation, conditioned on class

Probability of class, conditioned on observation ('posterior')

Binary Classification for Gaussian distributions

Assumption: within each class, features follow a Gaussian distribution

$$p(\mathbf{X} = \mathbf{x}|y = c) = \frac{1}{\sqrt{2\pi}^N |\Sigma_c|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_c)^T \Sigma_c^{-1} (\mathbf{x} - \mu_c)^T\right)$$

Shortcut notation: $p(\mathbf{x}|c)$ instead of $p(\mathbf{X} = \mathbf{x}|C = c)$

Posterior: $$p(1|\mathbf{x}) = \frac{p(\mathbf{x}|1)p(1)}{\sum_{c\in\{0,1\}} p(\mathbf{x}|c)p(c)}$$

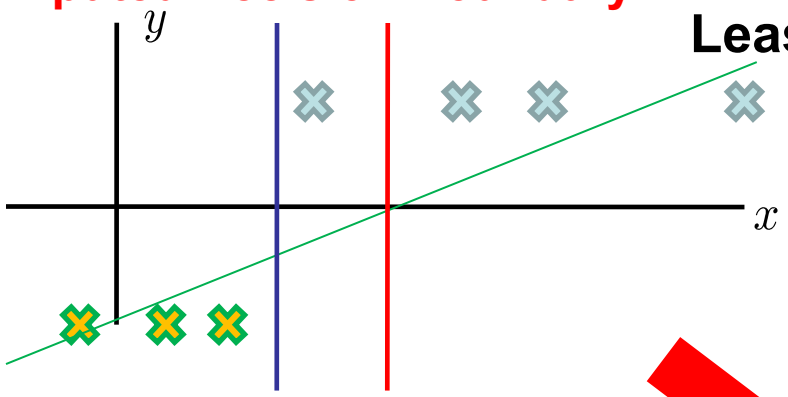
Special case: $\Sigma = \Sigma_0 = \Sigma_1$

$$p(1|\mathbf{x}) = \frac{1}{1 + \exp\left(-(\mathbf{w}^T\mathbf{x} + b)\right)}$$

$$\mathbf{w} = \Sigma^{-1}(\mu_1 - \mu_0) \qquad b = \frac{1}{2}(\mu_0^T\Sigma^{-1}\mu_0 - \mu_1^T\Sigma^{-1}\mu_1)$$

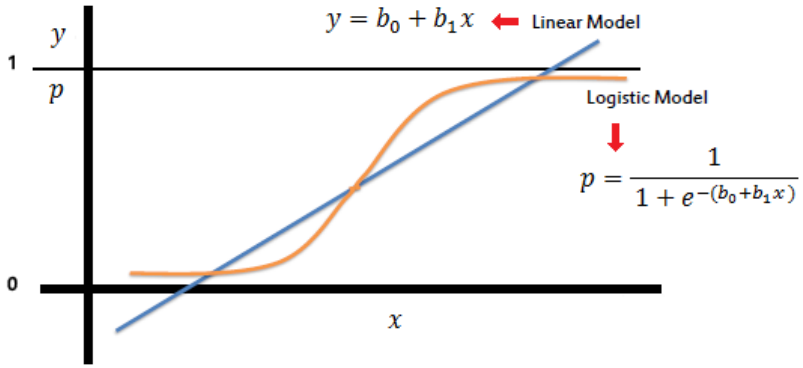
Back to classification

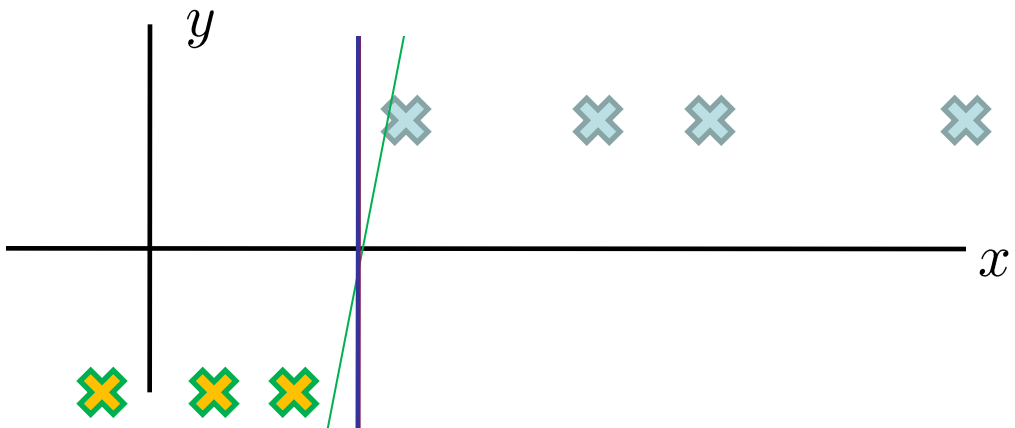
Computed Decision Boundary



Desired decision boundary

Least Squares Fit





Linear Discriminant

Desired & computed decision boundary

From regression to classification

Training set: $\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}, \mathbf{x} \in \mathbb{R}^D, y \in \{0, 1\}$

$$p(y^1, \ldots, y^n | \mathbf{x}^1, \ldots, \mathbf{x}^N) =$$

Independence assumption

$$= \prod_{i=1}^{N} p(y^i | \mathbf{x}^i)$$

?

Form of posterior distribution

Bernoulli-type conditional distribution

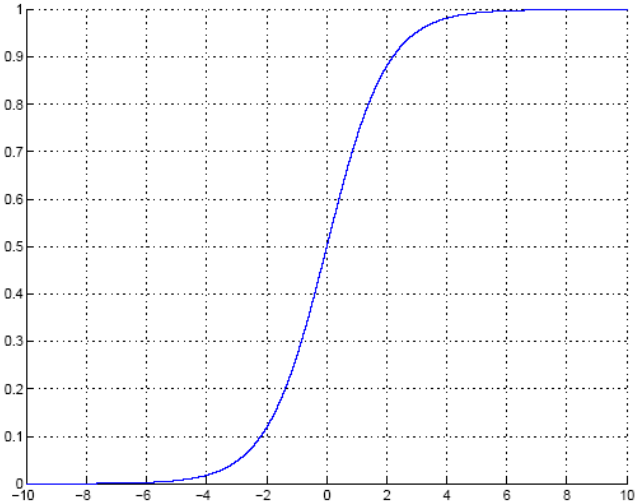
$$\left.\begin{array}{rl} P(Y=1|X=\mathbf{x};\mathbf{w}) = & f(\mathbf{x},\mathbf{w}) \\ P(Y=0|X=\mathbf{x};\mathbf{w}) = & 1-f(\mathbf{x},\mathbf{w}) \end{array}\right\} \rightarrow$$

$$P(Y=y|X=\mathbf{x};\mathbf{w}) = f(\mathbf{x},\mathbf{w})^{y}(1-f(\mathbf{x},\mathbf{w}))^{1-y}$$

Particular choice of form of f:

$$P(Y=1|X=\mathbf{x};\mathbf{w}) = g(\mathbf{w}^{T}\mathbf{x})$$

Sigmoidal: $g(\alpha) = \dfrac{1}{1+\exp(-a)}$



"squashing function":

$$-\infty \rightarrow 0$$
$$+\infty \rightarrow 1$$

From regression to classification, continued

Training set: $\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}, \mathbf{x} \in \mathbb{R}^D, y \in \{0, 1\}$

$$p(y^1, \ldots, y^n | \mathbf{x}^1, \ldots, \mathbf{x}^N) =$$

$$\overset{\text{Independence assumption}}{=} \prod_{i=1}^{N} P(y^i | \mathbf{x}^i)$$

$$= \prod_{i=1}^{N} g(\mathbf{w}^T \mathbf{x}^i)^{y^i} (1 - g(\mathbf{w}^T \mathbf{x}^i))^{1-y^i}$$

$$\log P(\mathbf{y} | \mathbf{X}; \mathbf{w}) = \sum_{i=1}^{N} \log \left(g(\mathbf{w}^T \mathbf{x}^i)^{y^i} \right) + \log \left(\left(1 - g(\mathbf{w}^T \mathbf{x}^i)\right)^{(1-y^i)} \right)$$

$$= \sum_{i=1}^{N} y^i \log g(\mathbf{w}^T \mathbf{x}^i) + (1 - y^i) \log(1 - g(\mathbf{w}^T \mathbf{x}^i))$$

Q1: How does this behave? Q2: How to optimize it with respect to w?

Loss function for linear regression

Training: given $S = \{(\mathbf{x}^i, y^i)\}, i = 1, \ldots, N$, estimate optimal $\mathbf{w}$

Loss function: quantify appropriateness of $\mathbf{w}$

$$L(S, \mathbf{w}) = \sum_{i=1}^{N} l(y^i, f_{\mathbf{w}}(\mathbf{x}^i))$$

$$= \sum_{i=1}^{N} (y^i - \mathbf{w}^T \mathbf{x}^i)^2 = \sum_{i=1}^{N} (\epsilon^i)^2$$

Loss function for classification

Training: given $S = \{(\mathbf{x}^i, y^i)\}, i = 1, \ldots, N$, estimate optimal $\mathbf{w}$

Loss function: quantify appropriateness of $\mathbf{w}$

$$
\begin{aligned}
L(S, \mathbf{w}) &= -\log P(\mathbf{y}|\mathbf{X}; \mathbf{w}) \\
&= -\sum_{\substack{i=1 \\ N}}^{N} \log P(y^i|\mathbf{x}^i; \mathbf{w}) \\
&= -\sum_{\substack{i=1 \\ N}}^{N} y^i \log g(\mathbf{w}^T\mathbf{x}^i) + (1 - y^i) \log(1 - g(\mathbf{w}^T\mathbf{x}^i)) \\
&= \sum_{i=1}^{} l(y^i, f_{\mathbf{w}}(\mathbf{x}^i))
\end{aligned}
$$

Linear discriminant: $f_{\mathbf{w}}(\mathbf{x}^i) = \mathbf{w}^T\mathbf{x}^i$

Rewriting the quadratic loss

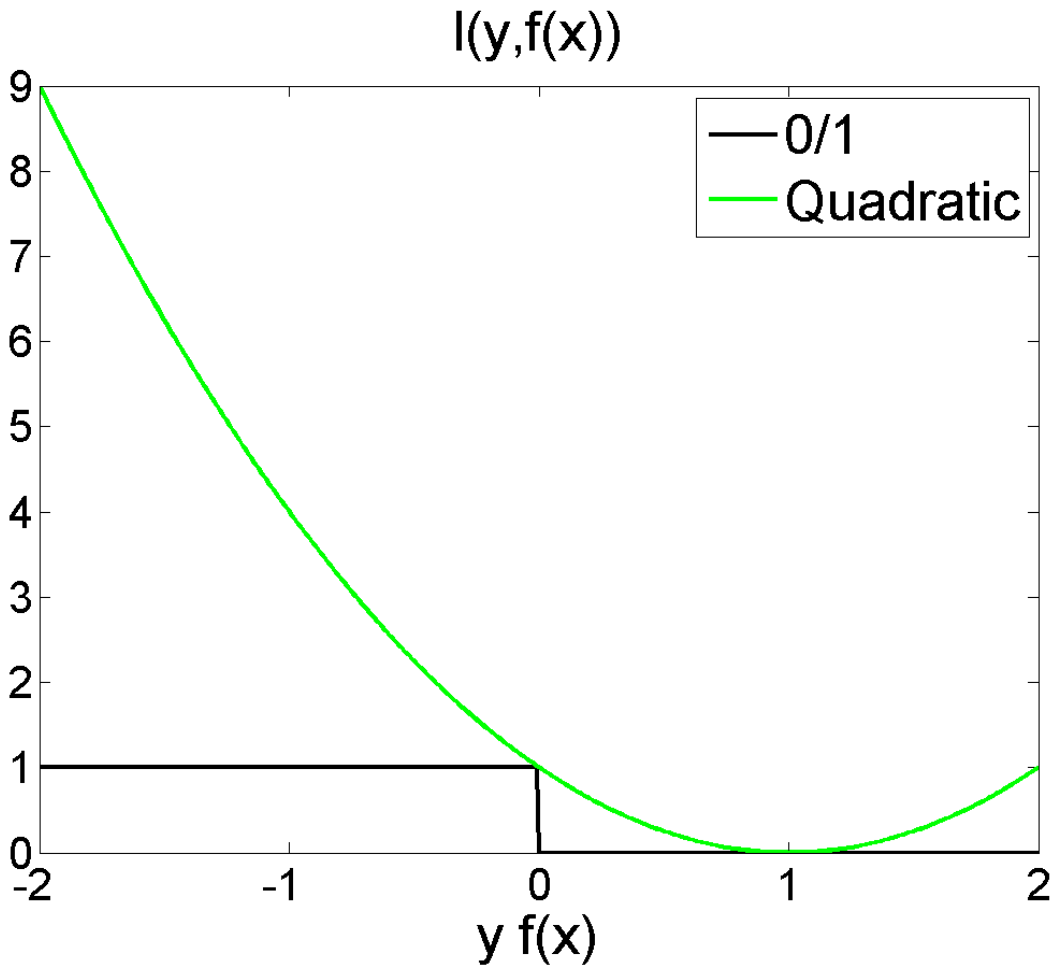
Consider transformation: $y_{\pm} = 2y_b - 1$

$y_b \in \{0, 1\} \qquad y_{\pm} \in \{-1, 1\}$

$$
\begin{aligned}
l(y, f_{\mathbf{w}}(\mathbf{x})) &= (y - f_{\mathbf{w}}(\mathbf{x}))^2 \\
&\overset{y^2=1}{=} y^2(y - f_{\mathbf{w}}(\mathbf{x}))^2 \\
&= (y^2 - y f_{\mathbf{w}}(\mathbf{x}))^2 \\
&\overset{y^2=1}{=} (1 - y f_{\mathbf{w}}(\mathbf{x}))^2
\end{aligned}
$$

Inappropriateness of quadratic loss for classification

Last slide: $l(y, f(x)) = (1 - yf(x))^2$



Quadratic loss is not robust to outliers and penalizes outputs that are `too good'

Rewriting the cross-entropy loss

As before: $h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ $\quad y_{\pm} = 2y_b - 1 \quad y_b \in \{0, 1\}$

$y_{\pm} \in \{-1, 1\}$

$$P(Y = 1|X = \mathbf{x}; \mathbf{w}) = \frac{1}{1 + \exp(-h_{\mathbf{w}}(\mathbf{x}))}$$

$$P(Y = -1|X = \mathbf{x}; \mathbf{w}) = 1 - P(Y = 1|X = \mathbf{x}; \mathbf{w})$$

$$= 1 - \frac{1}{1 + \exp(-h_{\mathbf{w}}(\mathbf{x}))}$$

$$= \frac{\exp(-h_{\mathbf{w}}(\mathbf{x}))}{1 + \exp(-h_{\mathbf{w}}(\mathbf{x}))}$$

$$= \frac{1}{1 + \exp(h_{\mathbf{w}}(\mathbf{x}))}$$

Rewriting the cross-entropy loss

As before: $h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ $\quad y_{\pm} = 2y_b - 1$ $\quad y_b \in \{0, 1\}$

$y_{\pm} \in \{-1, 1\}$

Last slide:

$$P(Y = 1|X = \mathbf{x}; \mathbf{w}) = \frac{1}{1 + \exp(-h_{\mathbf{w}}(\mathbf{x}))}$$

$$P(Y = -1|X = \mathbf{x}; \mathbf{w}) = \frac{1}{1 + \exp(h_{\mathbf{w}}(\mathbf{x}))}$$

Compact form:

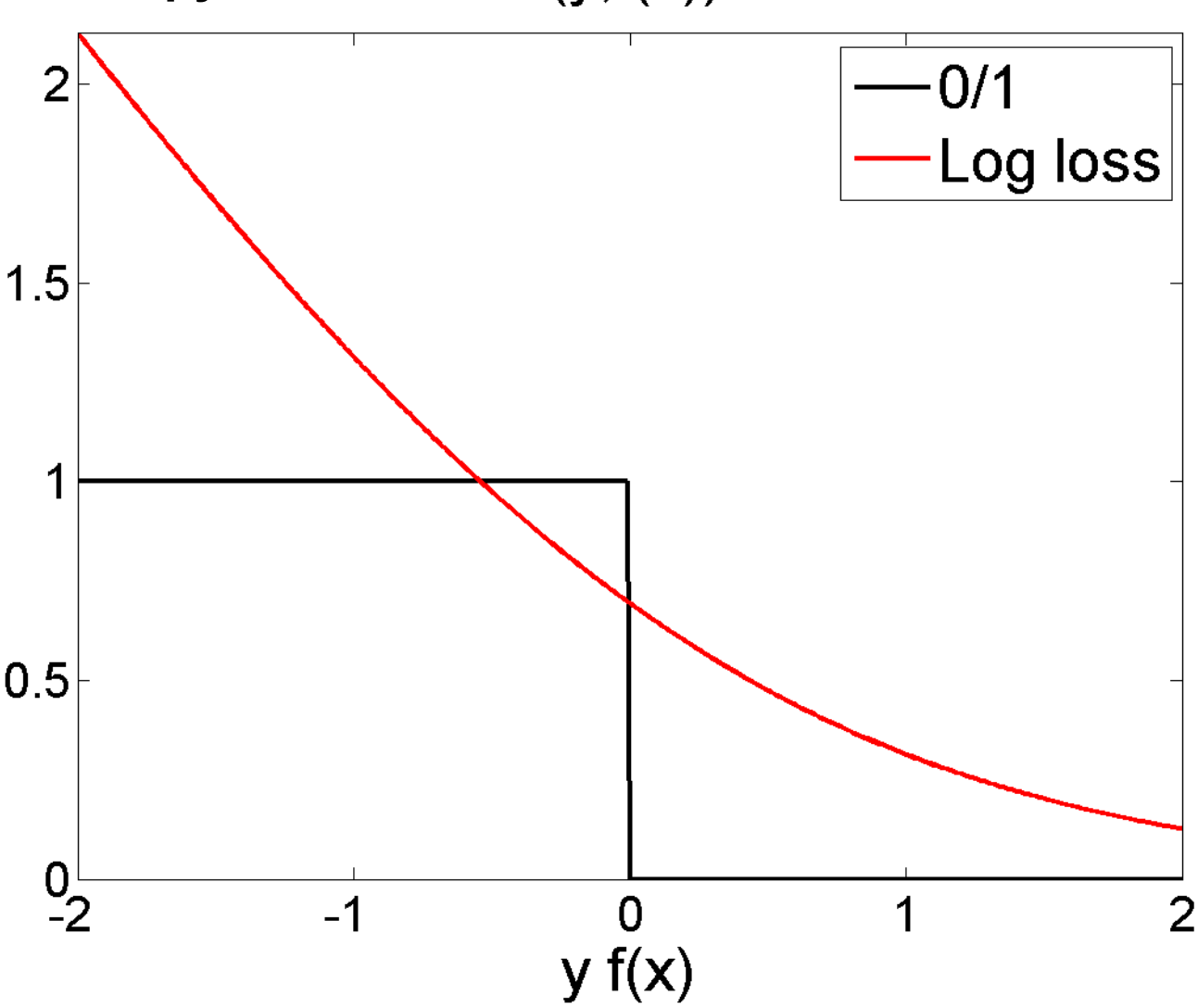
$$P(Y = y|X = \mathbf{x}; \mathbf{w}) = \frac{1}{1 + \exp(-yh_{\mathbf{w}}(\mathbf{x}))}$$

$$L(S, \mathbf{w}) = \sum_{i=1}^{N} -\log P(Y = y^i|X = \mathbf{x}^i; \mathbf{w})$$

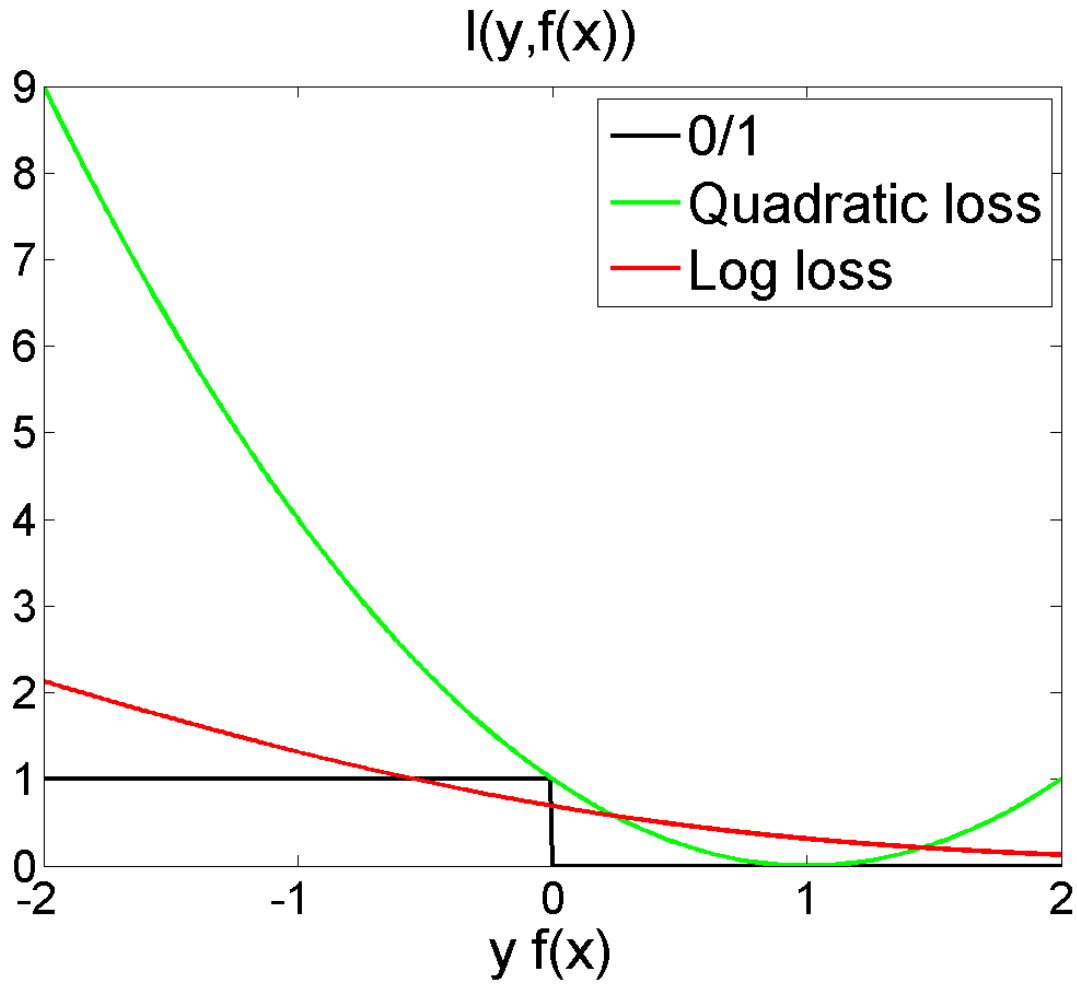
$$= \sum_{i=1}^{N} \log(1 + \exp(-y^i h_{\mathbf{w}}(\mathbf{x}^i)))$$

Log loss: $l(y, f(x)) = \log(1 + \exp(-yf(x)))$

a.k.a. "cross entropy" loss



Log loss vs. quadratic loss



Quadratic loss

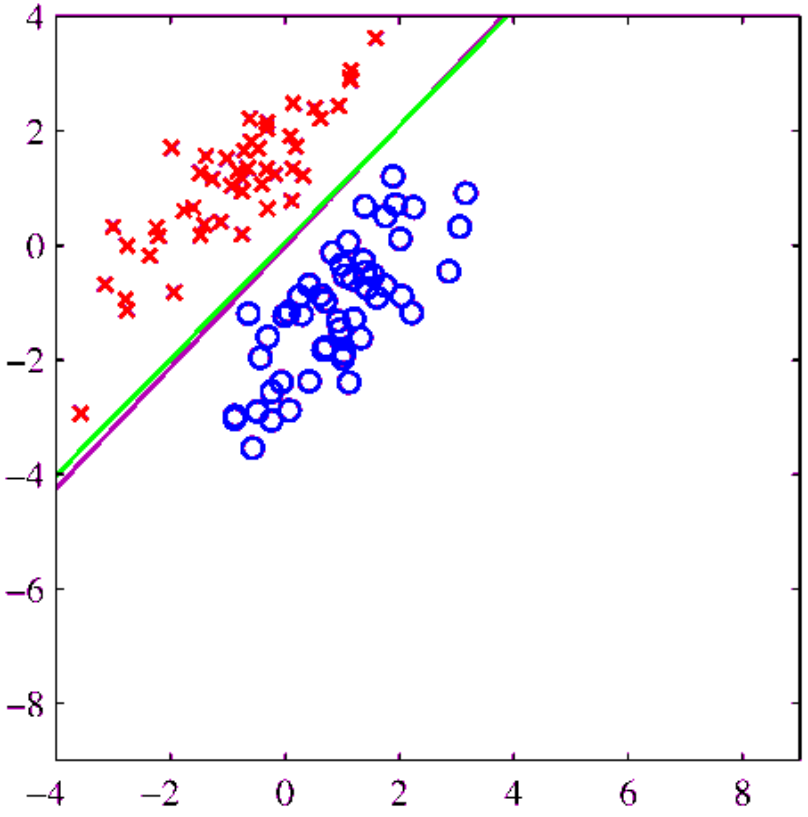
$$l(y, f(x)) = (1 - yf(x))^2$$

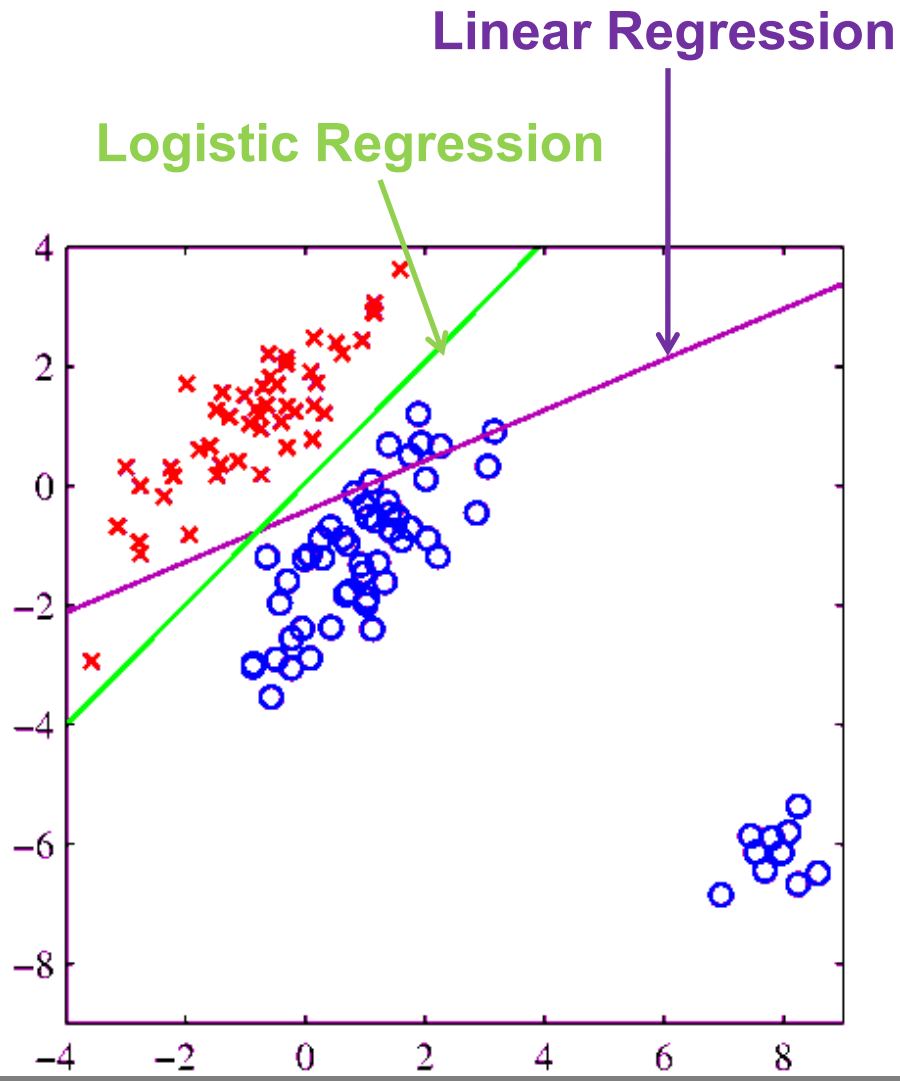
Log loss

$$l(y, f(x)) = \log(1 + \exp(-yf(x)))$$

Logistic vs Linear Regression

Logistic regression is more robust





From two to many

- So far: binary classification
- How about multi-class classification?

Multiple classes & linear regression

C classes: one-of-c coding (or one-hot encoding)

4 classes, i-th sample is in 3rd class: $\mathbf{y}^i = (0, 0, 1, 0)$

Matrix notation:

$$\mathbf{Y} = \left[\begin{array}{c} \mathbf{y}^1 \\ \hline \vdots \\ \hline \mathbf{y}^N \end{array}\right] = \left[\begin{array}{c|c|c} \mathbf{y}_1 & \dots & \mathbf{y}_C \end{array}\right] \qquad \text{where} \quad \mathbf{y}_c = \begin{bmatrix} y_c^1 \\ \vdots \\ y_c^N \end{bmatrix}$$

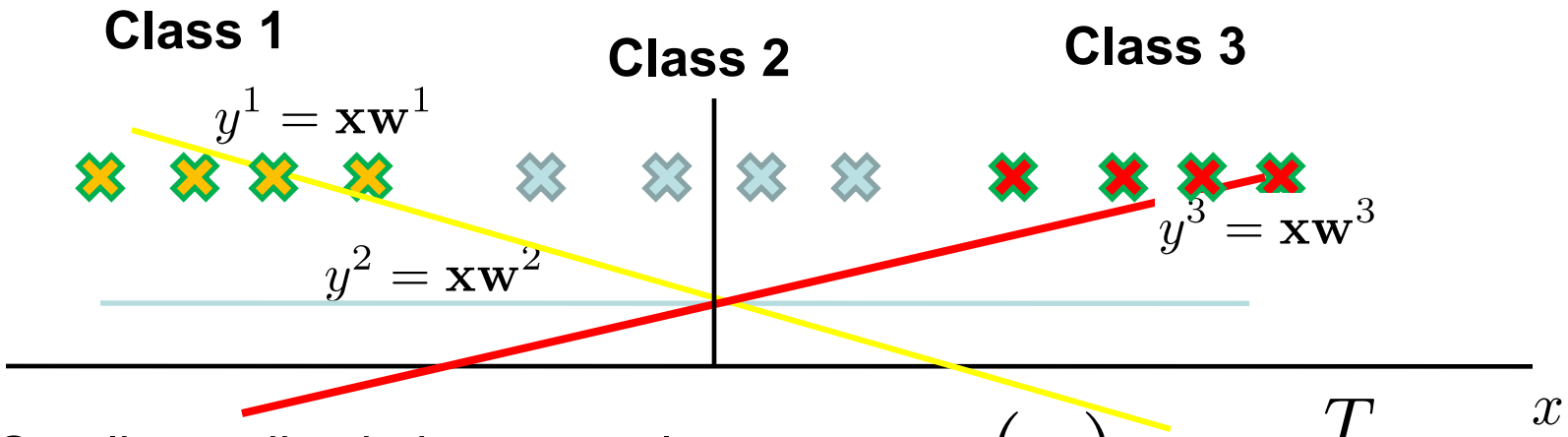
$$\mathbf{W} = \left[\begin{array}{c|c|c} \mathbf{w}_1 & \dots & \mathbf{w}_C \end{array}\right]$$

Loss function: $$L(\mathbf{W}) = \sum_{c=1}^{C} (\mathbf{y}_c - \mathbf{X}\mathbf{w}_c)^T (\mathbf{y}_c - \mathbf{X}\mathbf{w}_c)$$

Least squares fit (decouples per class):

$$\mathbf{w}_c^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{y}_c$$

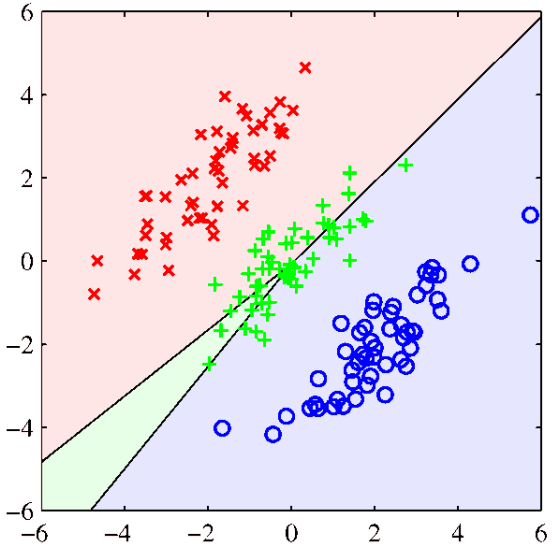
Linear regression: masking problem



One linear discriminant per class: $s_c(\mathbf{x}) = \mathbf{w}_c^T \mathbf{x}$

Nothing ever gets assigned to class 2!

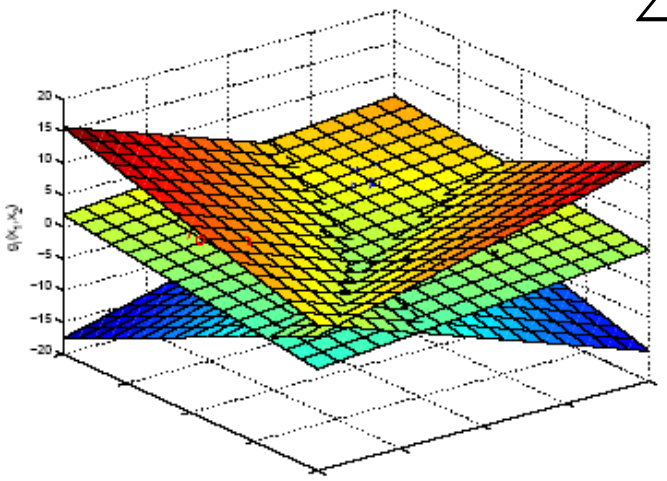
2D version:



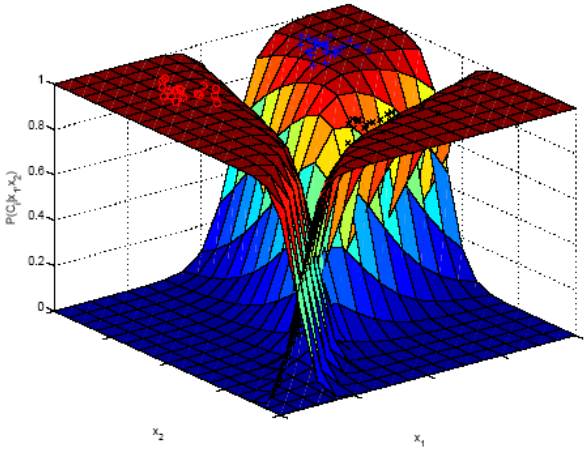
Multiple classes & logistic regression

Soft maximum (softmax) of competing classes:

$$P(y = c|\mathbf{x}; \mathbf{W}) = \frac{\exp(\mathbf{w}_c^T \mathbf{x})}{\sum_{c'=1}^{C} \exp(\mathbf{w}_{c'}^T \mathbf{x})} \doteq g_c(\mathbf{x}, \mathbf{W})$$



Discriminants (inputs)



Softmax (outputs)

Loss function for single-class classification

Training: given $S = \{(\mathbf{x}^i, y^i)\}, i = 1, \ldots, N$, estimate optimal $\mathbf{w}$

Loss function: quantify appropriateness of $\mathbf{w}$

$$L(S, \mathbf{w}) = -\log P(\mathbf{y}|\mathbf{X}; \mathbf{w})$$

$$= -\sum_{i=1}^{N} y^i \log g(\mathbf{w}^T \mathbf{x}^i) + (1 - y^i) \log(1 - g(\mathbf{w}^T \mathbf{x}^i))$$

Bernoulli model for posterior distribution:

$$P(Y = y|X = \mathbf{x}; \mathbf{w}) = g(\mathbf{w}^T \mathbf{x})^y (1 - g(\mathbf{w}^T \mathbf{x}))^{1-y}$$

Bernoulli & Categorical distribution

Binary random variable $Y \in \{0, 1\}$

Bernoulli Distribution:

$$P(Y = c) = \begin{cases} p & c = 1 \\ 1 - p & c = 0 \end{cases}$$

$$= p^c (1 - p)^{1-c}$$

Discrete random variable $Y \in \{1, \ldots, K\}$

$$P(Y = c) = \begin{cases} p_1, & c = 1 \\ \vdots & \\ p_K, & c = K \end{cases}$$

$$= \prod_{k=1}^{K} p_k^{[c=k]}$$

(where []: Iverson bracket)

Parameter estimation, multi-class case

One-hot label encoding: $\mathbf{y}^i = (0, 0, 1, 0)$

Likelihood of training sample: $(\mathbf{y}^i, \mathbf{x}^i)$

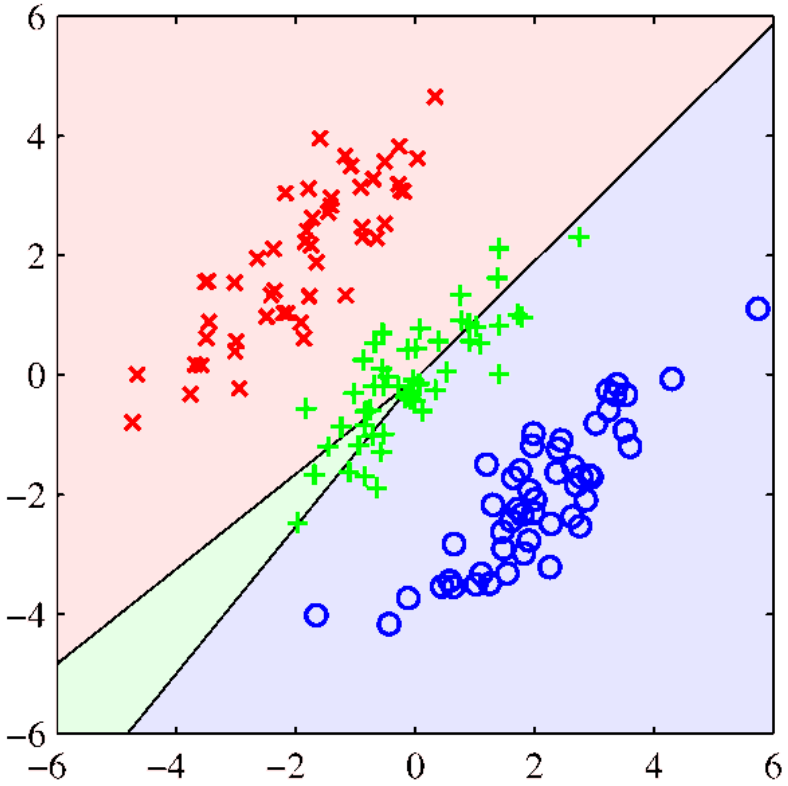
$$P(\mathbf{y}^i | \mathbf{x}^i; \mathbf{w}) = \prod_{c=1}^{C} \left(g_c(\mathbf{x}, \mathbf{W})\right)^{\mathbf{y}_c^i}$$

Optimization criterion:

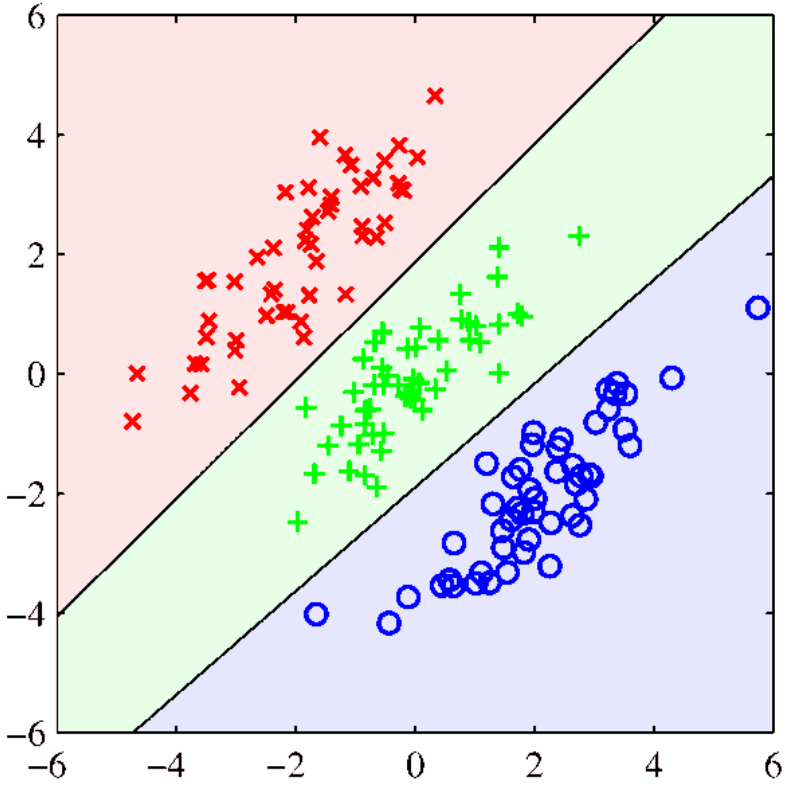
$$L(\mathbf{W}) = -\sum_{i=1}^{N} \sum_{c=1}^{C} \mathbf{y}_c^i \log \left(g_c(\mathbf{x}, \mathbf{W})\right)$$

Logistic vs Linear Regression, n>2 classes

Linear regression



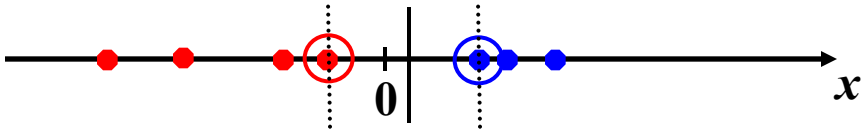
Logistic regression



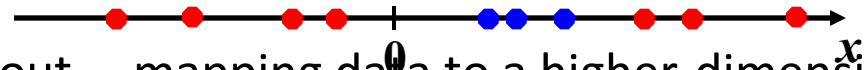
Logistic regression does not exhibit the masking problem

Non-linear detection boundaries

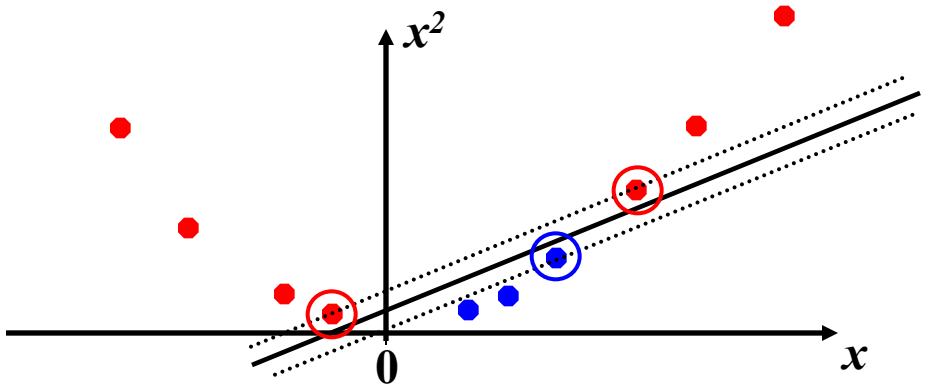
- Datasets that are linearly separable (with some noise) work out great:



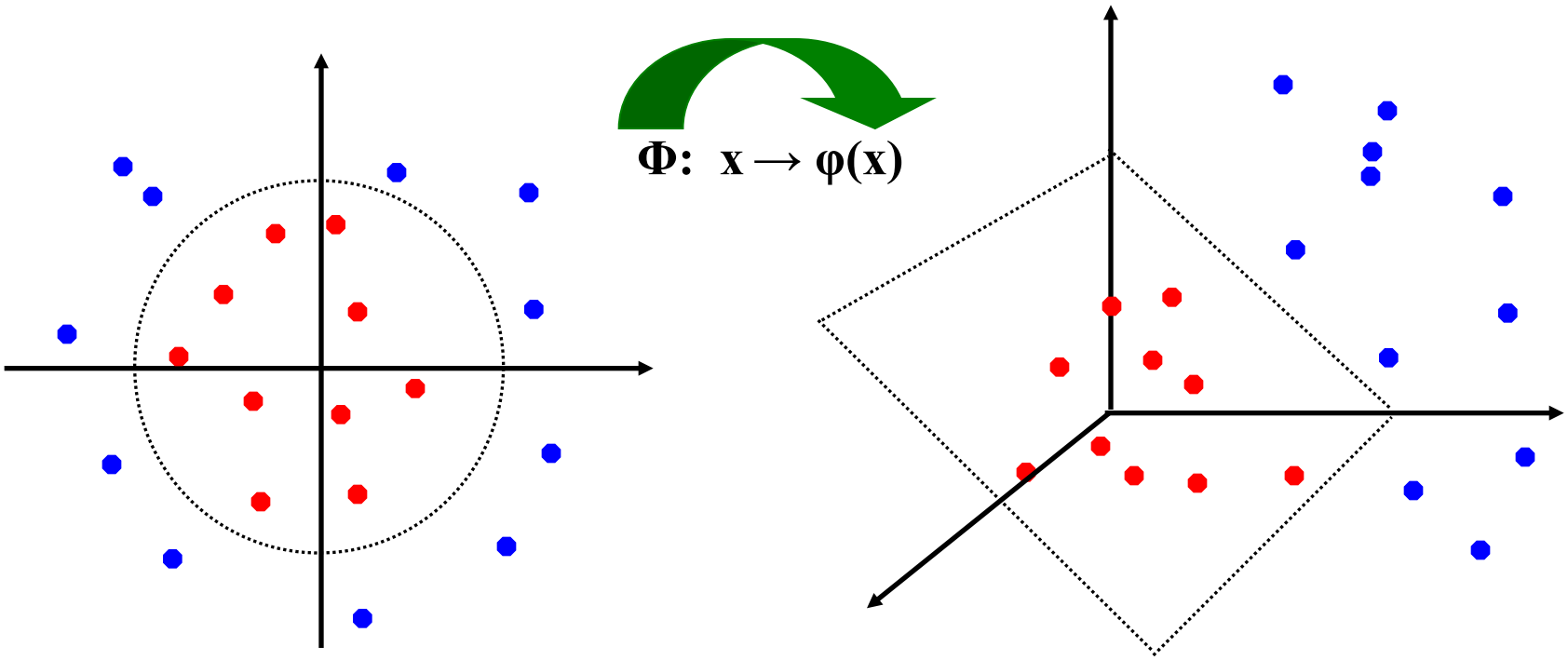
- But what are we going to do if the dataset is just too hard?



- How about … mapping data to a higher-dimensional space:



Non-linear decision boundaries



Parameter estimation, non-linear case

Linear case:

$$L(\mathbf{W}) = -\sum_{i=1}^{N}\sum_{c=1}^{C} \mathbf{y}_c^i \log\left(g_c(\mathbf{x}, \mathbf{W})\right)$$

Nonlinear case:

$$L(\mathbf{W}') = -\sum_{i=1}^{N}\sum_{c=1}^{C} \mathbf{y}_c^i \log\left(g_c(\phi(\mathbf{x}), \mathbf{W}')\right)$$