CS 2750 Machine Learning
Lecture 9

Multi-way classification

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Administrative announcements

- **Homework 3 due today**
- **Homework 4 out**

Multi-way classification

- **Binary classification** $Y = \{0,1\}$
- **Multi-way classification**
  - **K classes** $Y = \{0,1,\ldots,K-1\}$
  - **Goal:** learn to classify correctly K classes
  - Or **learn** $f : X \rightarrow \{0,1,\ldots,K-1\}$
- **Errors:**
  - **Zero-one (misclassification) error for an example:**

$$Error_1(\mathbf{x}_i, y_i) = \begin{cases} 1 & f(\mathbf{x}_i, \mathbf{w}) \neq y_i \\ 0 & f(\mathbf{x}_i, \mathbf{w}) = y_i \end{cases}$$

  - **Mean misclassification error (for a dataset):**

$$\frac{1}{n}\sum_{i=1}^{n} Error_1(\mathbf{x}_i, y_i)$$

Multi-way classification

Approaches:

- **Discriminative approach**
  - Parametric discriminant functions
  - Learns discriminant functions **directly**
    - Much like the logistic regression model.
- **Generative model approach**
  - Generative model of the distribution p(**x**,y)
  - Learns the parameters of the model through density estimation techniques
  - Discriminant functions defined on the top of the model
    - "Indirect" learning of a classifier

Generative model approach

Indirect:

1. **Represent and learn the distribution** $p(\mathbf{x}, y)$
2. **Define and use probabilistic discriminant functions**

$$g_i(\mathbf{x}) = \log\ p(y = i \mid \mathbf{x})$$

Model $p(\mathbf{x}, y) = p(\mathbf{x} \mid y)p(y)$

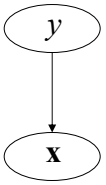
- $p(\mathbf{x} \mid y)$ = **Class-conditional distributions (densities)**

  k class conditional distributions

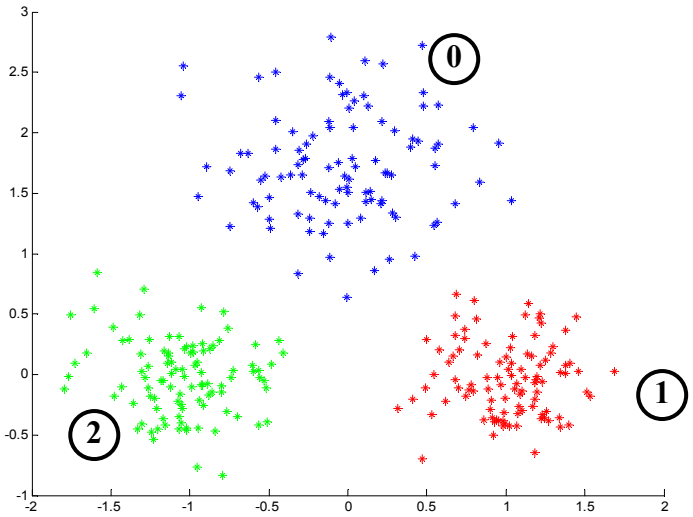
  $$p(\mathbf{x} \mid y = i) \qquad \forall i \quad 0 \le i \le K - 1$$

- $p(y)$ = **Priors on classes**
- - probability of class y

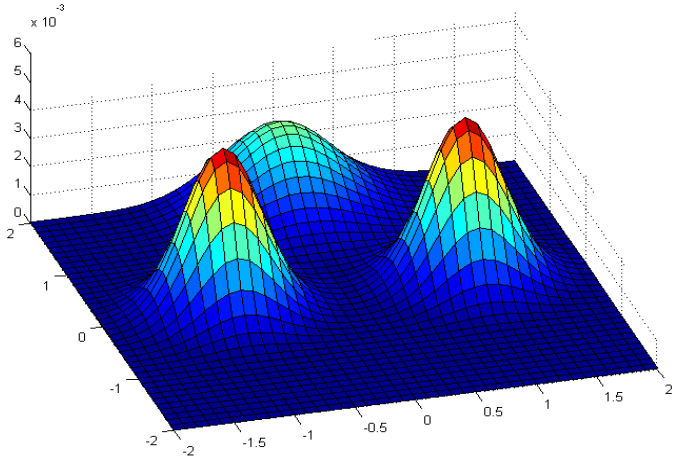
$$\sum_{i=1}^{K-1} p(y = i) = 1$$



Multi-way classification. Example



Multi-way classification



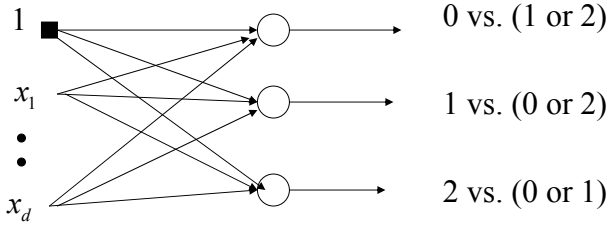
Discriminative approach.

- **Parameteric model** of discriminant functions
- Learns the discriminant functions directly

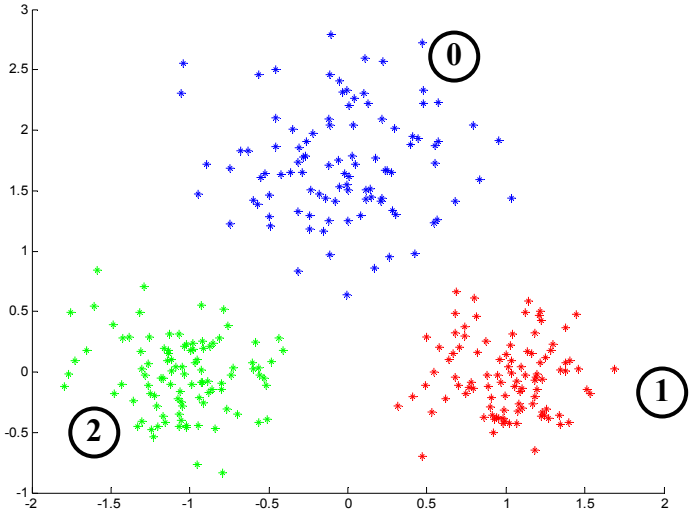
How to learn to classify multiple classes, say 0,1,2?

Approach 1:

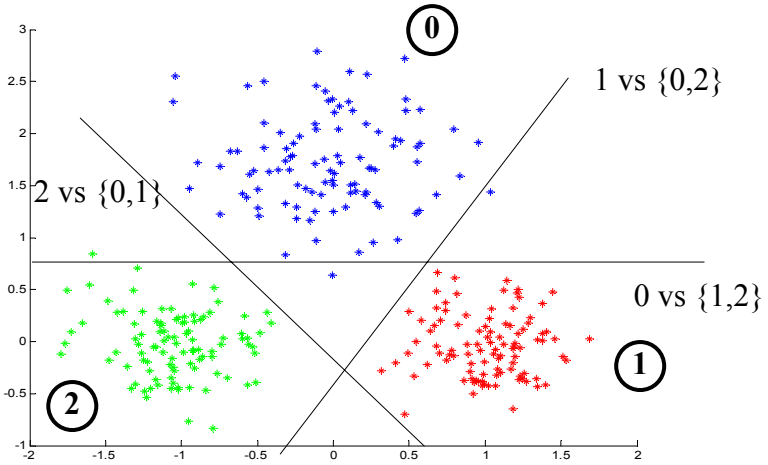
– Logistic regression on every class versus the rest



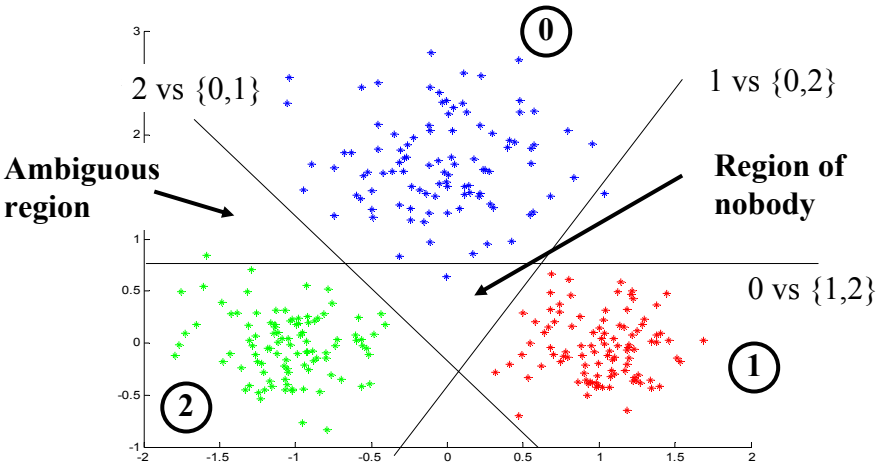
Multi-way classification. Example



Multi-way classification. Approach 1.



Multi-way classification. Approach 1.

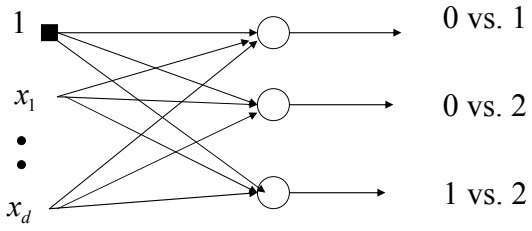


Discriminative approach.

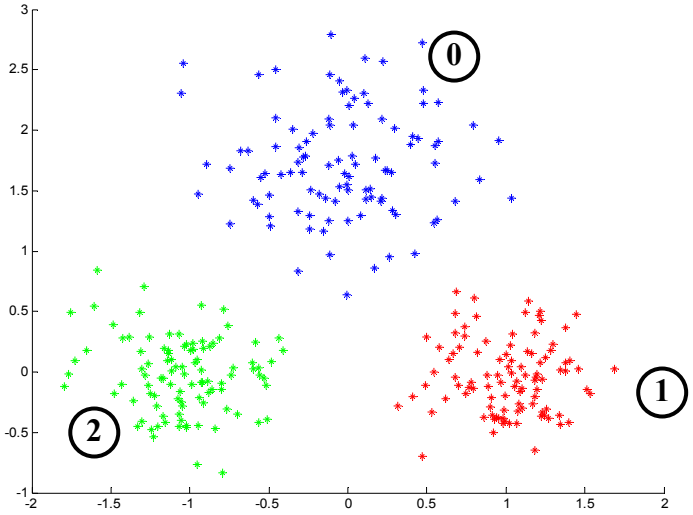
How to learn to classify multiple classes, say 0,1,2 ?

Approach 2:

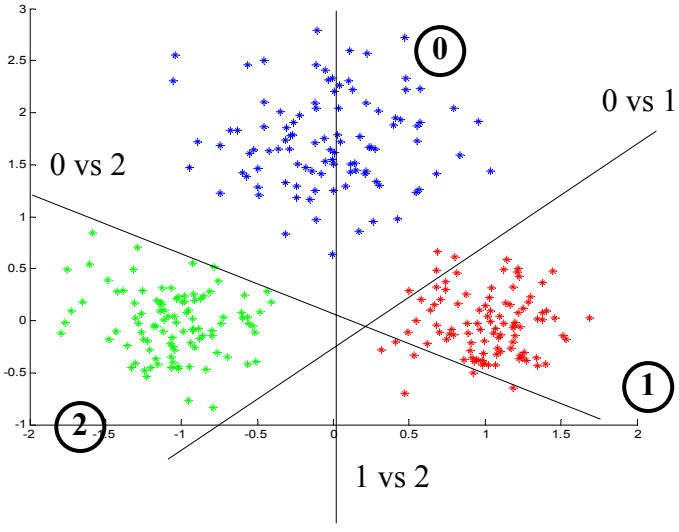
– Logistic regression on all pairs



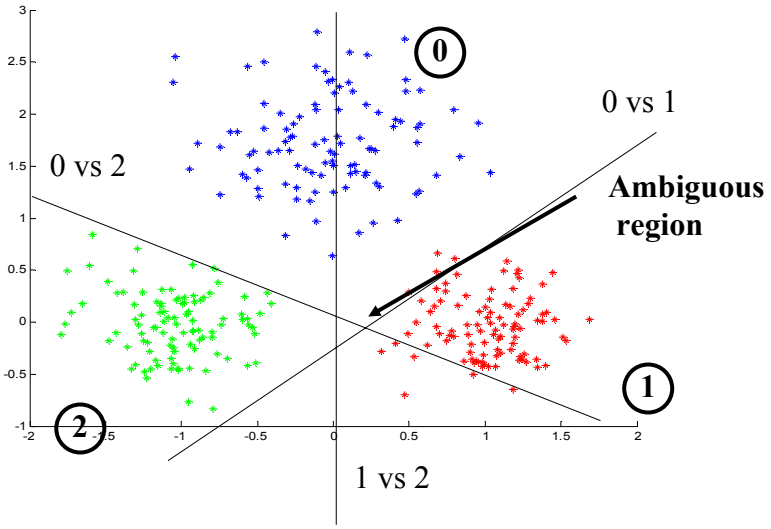
Multi-way classification. Example



Multi-way classification. Approach 2

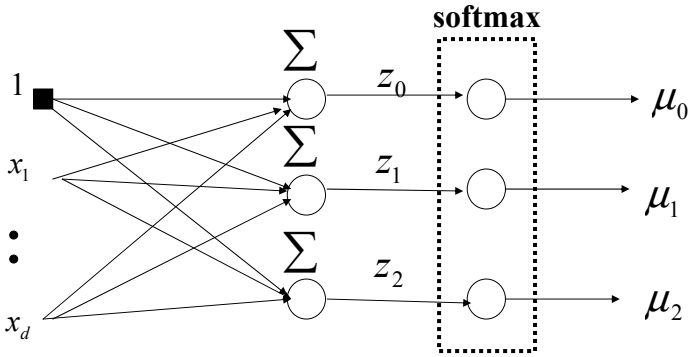


Multi-way classification. Approach 2



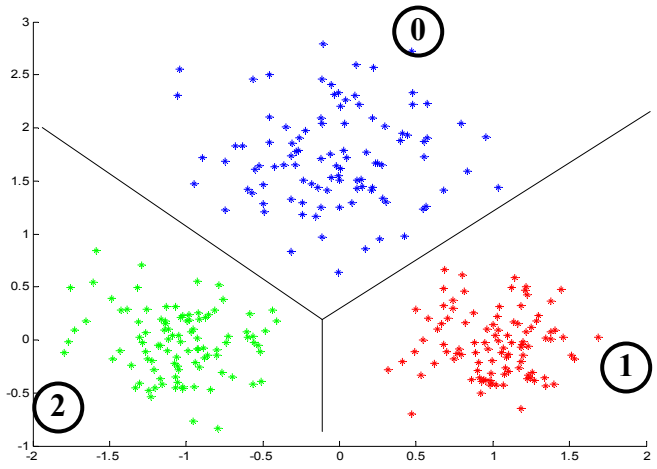
Multi-way classification with softmax

- A solution to the problem of having an ambiguous region



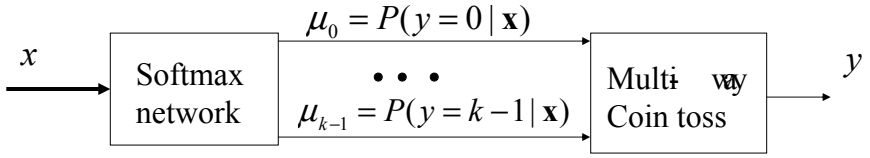
$$p(y=i \mid \mathbf{x}) = \mu_i = \frac{\exp(\mathbf{w}_i^T \mathbf{x})}{\sum_j \exp(\mathbf{w}_j^T \mathbf{x})} \qquad \sum_i \mu_i = 1$$

Multi-way classification with softmax



Learning of the softmax model

- Learning of parameters **w**: statistical view



Assume outputs y are transformed as follows

$$y \in \{0 \quad 1 \quad .. \quad k-1\} \quad \Rightarrow \quad y \in \left\{ \begin{pmatrix} 1 \\ 0 \\ .. \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 1 \\ .. \\ 0 \end{pmatrix} \quad .. \quad \begin{pmatrix} 0 \\ 0 \\ .. \\ 1 \end{pmatrix} \right\}$$

Learning of the softmax model

- Learning of the parameters **w**: statistical view
- **Likelihood of outputs**

$$L(D,\mathbf{w}) = p(\mathbf{Y} \mid \mathbf{X}, w) = \prod_{i=1,..n} p(y_i \mid \mathbf{x}_i, \mathbf{w})$$

- We want parameters **w** that maximize the likelihood
- **Log-likelihood trick**
  - Optimize log likelihood of outputs instead:

$$l(D,\mathbf{w}) = \log \prod_{i=1,..n} p(y_i \mid \mathbf{x}, \mathbf{w}) = \sum_{i=1,..n} \log p(y_i \mid \mathbf{x}, \mathbf{w})$$

$$= \sum_{i=1,..n} \sum_{q=0}^{k-1} \log \mu_i^{y_{i,q}} = \sum_{i=1,..n} \sum_{q=0}^{k-1} y_{i,q} \log \mu_{i,q}$$

- **Objective to optimize**

$$J(D_i, \mathbf{w}) = -\sum_{i=1}^{n} \sum_{q=0}^{k-1} y_{i,q} \log \mu_{i,q}$$

Learning of the softmax model

- **Error to optimize:**

$$J(D_i, \mathbf{w}) = -\sum_{i=1}^{n} \sum_{q=0}^{k-1} y_{i,q} \log \mu_{i,q}$$

- **Gradient**

$$\frac{\partial}{\partial w_{jk}} J(D_i, \mathbf{w}) = \sum_{i=1}^{n} -x_{i,j}(y_{i,j} - \mu_{i,j})$$

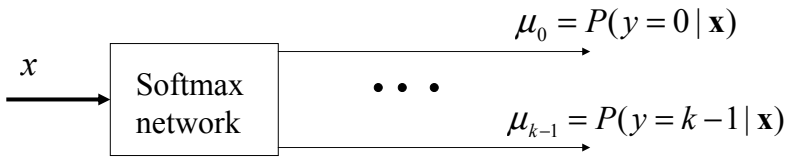
- The same very easy **gradient update** as used for the logistic regression

$$\mathbf{w}_j \leftarrow \mathbf{w}_j + \alpha \sum_{i=1}^{n} (y_{i,j} - \mu_{i,j}) \mathbf{x}_i$$

- But now we have to update weights of k networks

Multi-way classification

- When is the **softmax** the right model ?



- Assume:

$$p(\mathbf{x} \mid y=i) = h(\mathbf{x}, \boldsymbol{\varphi}) \exp\left\{ \frac{(\boldsymbol{\theta}_i^T \mathbf{x} - A(\boldsymbol{\theta}_i))}{a(\boldsymbol{\varphi})} \right\}$$

$\boldsymbol{\theta}_i$ - location parameter for class conditional i

$\boldsymbol{\varphi}$ - scaling parameter (the same for all classes)

Multi-way classification

- **Class conditional:**

$$p(\mathbf{x} \mid y=i) = h(\mathbf{x}, \boldsymbol{\varphi}) \exp\left\{ \frac{(\boldsymbol{\theta}_i^T \mathbf{x} - A(\boldsymbol{\theta}_i))}{a(\boldsymbol{\varphi})} \right\}$$

- **Class posterior:**

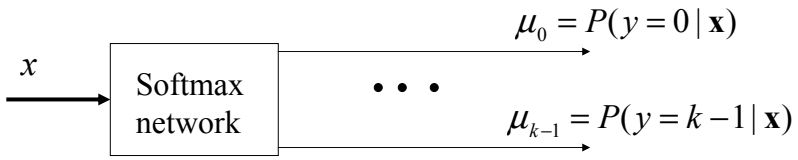
$$p(y=i \mid x) = \frac{p(\mathbf{x} \mid y=i)p(y=i)}{\sum_j p(\mathbf{x} \mid y=j)p(y=j)}$$

$$= \frac{h(\mathbf{x}, \boldsymbol{\varphi}) \exp\left\{ \frac{(\boldsymbol{\theta}_i^T \mathbf{x} - A(\boldsymbol{\theta}_i))}{a(\boldsymbol{\varphi})} \right\} p(y=i)}{\sum_j h(\mathbf{x}, \boldsymbol{\varphi}) \exp\left\{ \frac{(\boldsymbol{\theta}_j^T \mathbf{x} - A(\boldsymbol{\theta}_j))}{a(\boldsymbol{\varphi})} \right\} p(y=j)} = \frac{\exp(\mathbf{w}_i^T \mathbf{x} + b_i)}{\sum_j \exp(\mathbf{w}_j^T \mathbf{x} + b_j)}$$

$$\mathbf{w}_i = \frac{\boldsymbol{\theta}_i}{a(\boldsymbol{\varphi})} \qquad b_i = \frac{A(\boldsymbol{\theta}_i)}{a(\boldsymbol{\varphi})} + \ln p(y=i)$$

Multi-way classification

- **Softmax model is an accurate model** when class conditional densities are represented with densities from the exponential family and the same scaling parameter



$$p(\mathbf{x} \mid y=i) = \exp\left\{\frac{(\boldsymbol{\theta}_i^T \mathbf{x} - b(\boldsymbol{\theta}_i))}{a(\boldsymbol{\varphi})} + c(\mathbf{x}, \boldsymbol{\varphi})\right\}$$

$\boldsymbol{\theta}_i$ - location parameter for class conditional i

$\boldsymbol{\varphi}$ - scaling parameter (the same for all classes)

CS 2750 Machine Learning
Lecture 9b

Bayesian decision theory.

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Confusion matrix

- **Confusion matrix:**
  - Counts of examples with:
  - class label ω_j that are classified with a label α_i

	$\alpha = 0$	$\alpha = 1$	$\alpha = 2$
$\omega = 0$	140	20	22
$\omega = 1$	17	54	8
$\omega = 2$	12	4	76

agreement

Zero-one loss function

- **Misclassification error**
  - Based on the zero one loss function
    - Any misclassified example counts as 1
    - Correctly classified example counts as 0

	$\alpha = 0$	$\alpha = 1$	$\alpha = 2$
$\omega = 0$	140	20	22
$\omega = 1$	17	54	8
$\omega = 2$	12	4	76

Loss of 1

Loss of 1

Loss 0

General loss function

- **Error function based on a more general loss function**
  - Different misclassifications have different weight (loss)
  - α_i our choice
  - ω_j true label
  - $\lambda(\alpha_i \mid \omega_j)$ loss for classification

Example:

$\lambda(\alpha_i \mid \omega_j)$

	$\alpha = 0$	$\alpha = 1$	$\alpha = 2$
$\omega = 0$	0	1	5
$\omega = 1$	3	0	2
$\omega = 2$	3	1	0

Bayesian decision theory

- **More general loss function**
  - Different misclassifications have different weight (loss)

    $\lambda(\alpha_i \mid \omega_j)$
- **Expected loss for choice** (action) α_i

$$R(\alpha_i \mid \mathbf{x}) = \sum_j \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid \mathbf{x})$$

  - Also called conditional risk
- **Decision rule:** $\alpha(\mathbf{x})$
  - Chooses label (action) according to the input
- **Overall expected loss for the decision rule**

$$R(\alpha) = \int R(\alpha(\mathbf{x}), \mathbf{x}) P(\mathbf{x}) d\mathbf{x}$$

Bayesian decision theory

- **The optimal decision rule**

$$\alpha^*(\mathbf{x}) = \arg\min_{\alpha_i} \sum_j \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid \mathbf{x})$$

How to modify classifiers to handle different loss functions?

- **Discriminative models:**
  - Directly optimize the parameters according to the new loss function – estimate the loss based on examples
- **Generative models:**
  - Learn probabilities as before
  - Decisions about classes are biased to minimize the empirical loss (as seen above)

$$\alpha^*(\mathbf{x}) = \arg\min_{\alpha_i} \sum_j \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid \mathbf{x})$$

Estimating the loss

- Counts in the **confusion matrix can be used to estimate the expected loss function:**

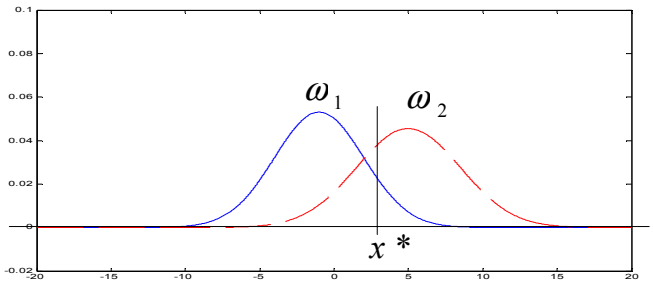
$N(\alpha_i \mid \omega_j)$	$\alpha = 0$	$\alpha = 1$	$\alpha = 2$
$\omega = 0$	140	20	22
$\omega = 1$	17	54	8
$\omega = 2$	12	4	76

agreement

- **Loss**

$$\frac{1}{N} \sum_i \sum_j \lambda(\alpha_i \mid \omega_j) N(\alpha_i \mid \omega_j)$$

Binary decisions: accuracy.



- **Probabilities:**
  - *True positive (hit)* $p(x > x^* \mid \mathbf{x} \in \omega_2)$
  - *False positive (false alarm)* $p(x > x^* \mid \mathbf{x} \in \omega_1)$
  - *True negative (correct rejection)* $p(x < x^* \mid \mathbf{x} \in \omega_1)$
  - *False negative (a miss)* $p(x < x^* \mid \mathbf{x} \in \omega_2)$

Binary decision. Accuracy

- **Confusion matrix**

	$x \in \omega_2(+)$	$x \in \omega_1(-)$	
$x > x^*(+)$	140	10	$PPT = 140/150$
$x < x^*(-)$	20	180	$NPV = 180/200$
	$SENS = 140/160$	$SPEC = 180/190$	

Row and column quantities:

- Sensitivity (SENS)
- Specificity (SPEC)
- Positive predictive value (PPV)
- Negative predictive value (NPV)

Classification accuracy

- **Sensitivity**

$$\frac{p(x > x^* \mid \mathbf{x} \in \omega_2)}{p(x > x^* \mid \mathbf{x} \in \omega_2) + p(x < x^* \mid \mathbf{x} \in \omega_2)} = \frac{TP}{TP + FN}$$

- **Specificity**

$$\frac{p(x < x^* \mid \mathbf{x} \in \omega_1)}{p(x < x^* \mid \mathbf{x} \in \omega_1) + p(x > x^* \mid \mathbf{x} \in \omega_1)} = \frac{TN}{TN + FP}$$

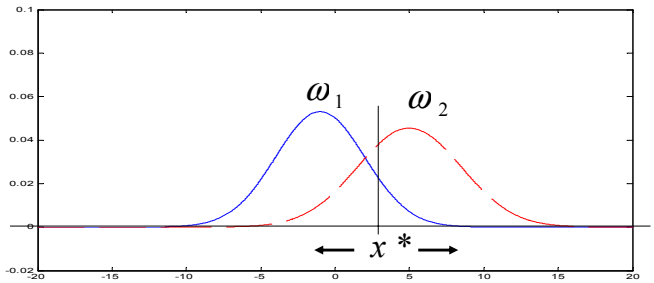
- **Positive predictive value**

$$\frac{p(x > x^* \mid \mathbf{x} \in \omega_2)}{p(x > x^* \mid \mathbf{x} \in \omega_2) + p(x > x^* \mid \mathbf{x} \in \omega_1)} = \frac{TP}{TP + FP}$$

- **Negative predictive value**

$$\frac{p(x < x^* \mid \mathbf{x} \in \omega_1)}{p(x < x^* \mid \mathbf{x} \in \omega_1) + p(x < x^* \mid \mathbf{x} \in \omega_2)} = \frac{TN}{TN + FN}$$

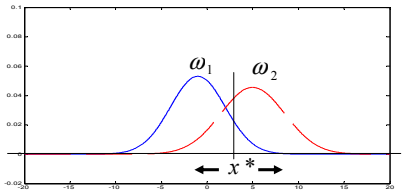
Decision threshold

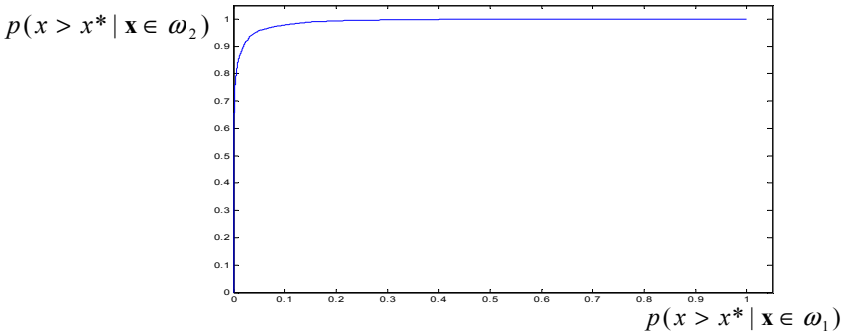


- **Movement of x* changes the probabilities:**
  - *True positive (hit)* $p(x > x^* \mid \mathbf{x} \in \omega_2)$
  - *False positive (false alarm)* $p(x > x^* \mid \mathbf{x} \in \omega_1)$
  - *True negative (correct rejection)* $p(x < x^* \mid \mathbf{x} \in \omega_1)$
  - *False negative (a miss)* $p(x < x^* \mid \mathbf{x} \in \omega_2)$

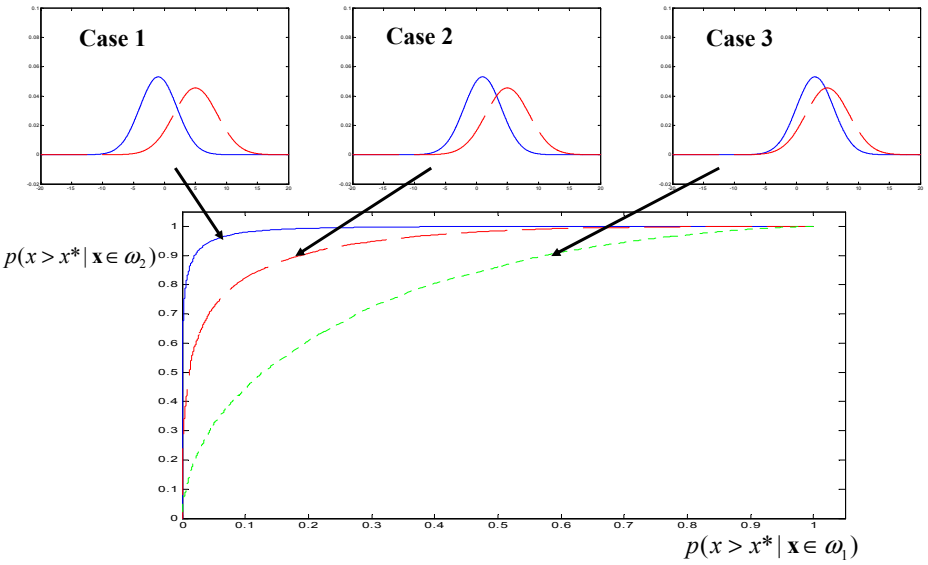
Receiver Operating Characteristic (ROC)

- **ROC curve plots :**
  $p(x > x^* \mid \mathbf{x} \in \omega_1)$
  vs $p(x > x^* \mid \mathbf{x} \in \omega_2)$
  for different x*





ROC curve



Receiver operating characteristic

- **ROC**
  - shows the discriminability between the two classes under different decision biases
- **Decision bias**
  - can be changed using different loss function