CS 1675 Machine Learning
Lecture 12

Generative models for classification

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

Classification

- **Data:** $D = \{d_1, d_2, .., d_n\}$
  $d_i = <\mathbf{x}_i, y_i>$
  - y_i **represents a discrete class value**
- **Goal: learn** $f : X \rightarrow Y$
- **Binary classification**
  - **A special case when** $Y \in \{0,1\}$
- **First step:**
  - we need to devise a model of the function f

Discriminant functions

- A common way to represent a **classifier is by using**
  - **Discriminant functions**
- **Works for both the binary and multi-way classification**
- **Idea:**
  - For every class $i = 0,1, \ldots k$ define a function $g_i(\mathbf{x})$ mapping $X \to \Re$
  - When the decision on input **x** should be made choose the class with the highest value of $g_i(\mathbf{x})$

$$y^* = \arg\max_i \; g_i(\mathbf{x})$$

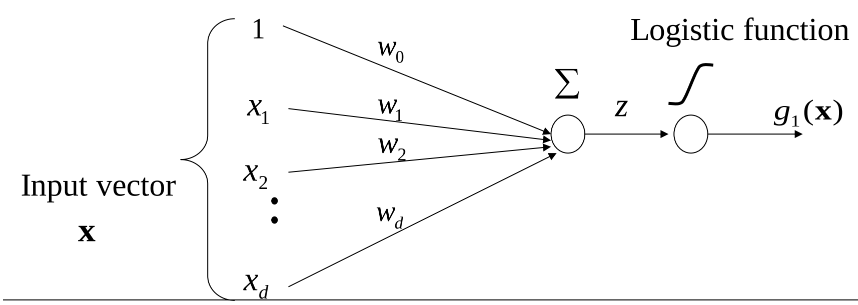
Logistic regression model

- **Discriminant functions:**

$$g_1(\mathbf{x}) = g(\mathbf{w}^T\mathbf{x}) \qquad g_0(\mathbf{x}) = 1 - g(\mathbf{w}^T\mathbf{x})$$

- **where** $g(z) = 1/(1+e^{-z})$ - is a logistic function

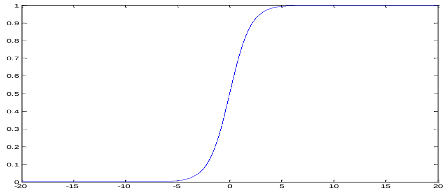
$$g_1(\mathbf{w}^T\mathbf{x}) = g(\mathbf{w}^T\mathbf{x}) = p(y=1 \mid \mathbf{x})$$



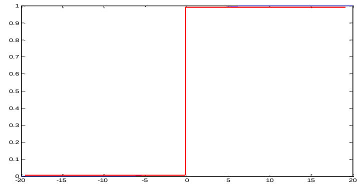
Logistic function

Function: $$g(z) = \frac{1}{(1+e^{-z})}$$

- Is also referred to as a **sigmoid function**
- takes a real number and outputs the number in the interval [0,1]
- Models a smooth switching function; replaces hard threshold function



Logistic (smooth) switching



Threshold (hard) switching

Generative approach to classification

Logistic regression:

- **Represents and learns a model of** $p(y \mid \mathbf{x})$
- **An example of a discriminative classification approach**
- **Model is <u>unable</u> to sample (generate) data instances** $(\mathbf{x}, y)$

Generative approach:

- **Represents and learns a joint distribution** $p(\mathbf{x}, y)$
- **Model is <u>able</u> to sample (generate) data instances** $(\mathbf{x}, y)$
- **The joint model defines probabilistic discriminant functions**

How?

$$g_1(\mathbf{x}) = p(y=1 \mid \mathbf{x}) = \frac{p(\mathbf{x}, y=1)}{p(\mathbf{x})} = \frac{p(\mathbf{x} \mid y=1)\, p(y=1)}{p(\mathbf{x})}$$

$$g_o(\mathbf{x}) = p(y=0 \mid \mathbf{x}) = \frac{p(\mathbf{x}, y=0)}{p(\mathbf{x})} = \frac{p(\mathbf{x} \mid y=0)\, p(y=0)}{p(\mathbf{x})}$$

$$p(y=0 \mid \mathbf{x}) + p(y=1 \mid \mathbf{x}) = 1$$

Generative approach to classification

Typical joint model $p(\mathbf{x}, y) = p(\mathbf{x} \mid y)\, p(y)$

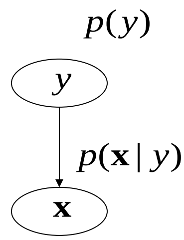
- $p(\mathbf{x} \mid y)$ = **Class-conditional distributions (densities)**

  binary classification: two class-conditional distributions

  $p(\mathbf{x} \mid y = 0) \qquad p(\mathbf{x} \mid y = 1)$

- $p(y)$ = **Priors on classes**
  - probability of class y
  - for binary classification: Bernoulli distribution

  $p(y = 0) + p(y = 1) = 1$



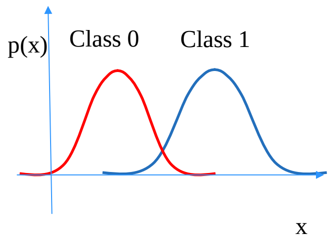
Quadratic discriminant analysis (QDA)

Model:

- **Class-conditional distributions are**
  - **multivariate normal distributions**

  $\mathbf{x} \sim N(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0) \quad \text{for} \quad y = 0$

  $\mathbf{x} \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) \quad \text{for} \quad y = 1$



Multivariate normal $\mathbf{x} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

$$p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right]$$

- **Priors on classes (class 0,1)** $y \sim Bernoulli$
  - **Bernoulli distribution**

  $p(y, \theta) = \theta^y (1 - \theta)^{1-y} \qquad y \in \{0,1\}$

Learning of parameters of the QDA model

Density estimation in statistics

- We see examples – we do not know the parameters of Gaussians (class-conditional densities)

$$p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right]$$

- **ML estimate of parameters** of a multivariate normal $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for a set of n examples of $\mathbf{x}$

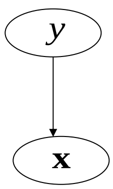
  Optimize log-likelihood: $l(D, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^{n} p(\mathbf{x}_i \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$

$$\hat{\boldsymbol{\mu}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \qquad \hat{\boldsymbol{\Sigma}} = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^T$$

- How about **class priors**?

Learning Quadratic discriminant analysis (QDA)

- **Learning Class-conditional distributions**
  - **Learn parameters of 2 multivariate normal distributions**



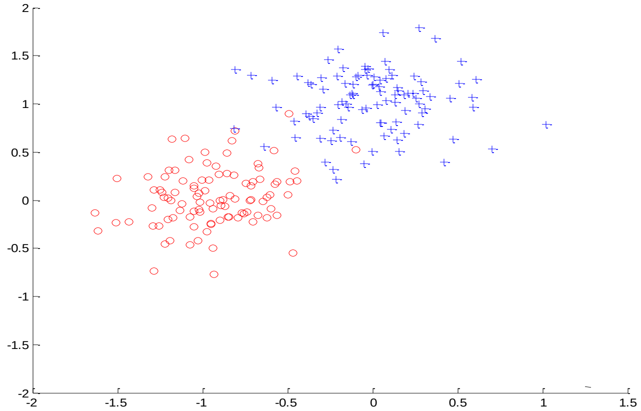
$$\mathbf{x} \sim N(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0) \quad \text{for} \quad y = 0$$
$$\mathbf{x} \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) \quad \text{for} \quad y = 1$$

  - **Use the density estimation methods**

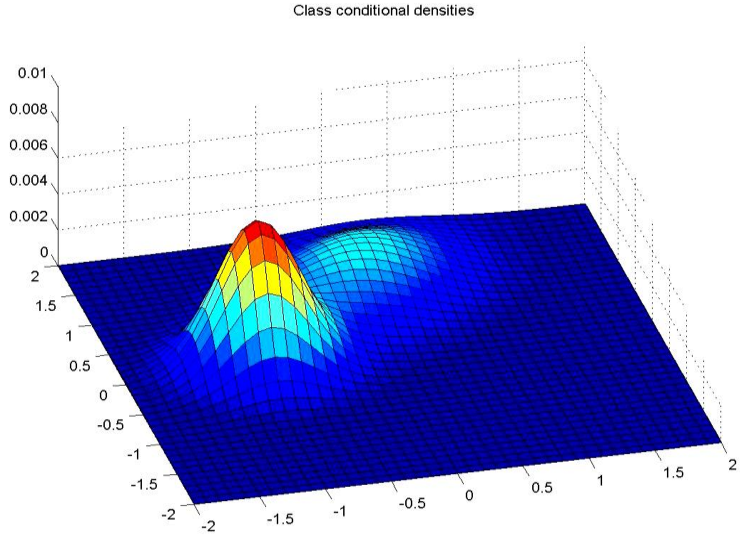
- **Learning Priors on classes (class 0,1)** $\quad y \sim Bernoulli$
  - **Learn the parameter of the Bernoulli distribution**
  - **Again use the density estimation methods**

$$p(y, \theta) = \theta^y (1 - \theta)^{1-y} \qquad y \in \{0,1\}$$

QDA



2 Gaussian class-conditional densities



QDA: Making class decision

Basically we need to design discriminant functions

- **Posterior of a class** – choose the class with better posterior probability

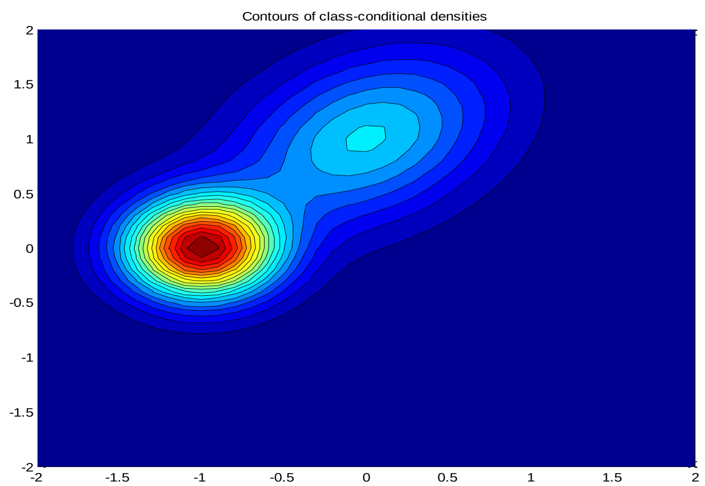
$$\underbrace{p(y=1\,|\,\mathbf{x})}_{g_1(\mathbf{x})} > \underbrace{p(y=0\,|\,\mathbf{x})}_{g_0(\mathbf{x})} \quad \Rightarrow \quad \begin{array}{l}\text{then } y=1 \\ \text{else } y=0\end{array}$$

$$p(y=1\,|\,\mathbf{x}) = \frac{p(\mathbf{x}\,|\,\mu_1,\mathbf{\Sigma}_1)\,p(y=1)}{p(\mathbf{x}\,|\,\mu_0,\mathbf{\Sigma}_0)\,p(y=0) + p(\mathbf{x}\,|\,\mu_1,\mathbf{\Sigma}_1)\,p(y=1)}$$

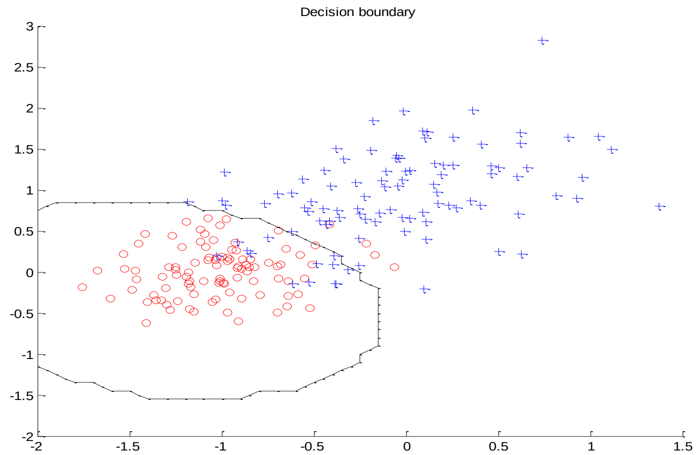
- **Notice it is sufficient to compare:**

$$p(\mathbf{x}\,|\,\mu_1,\mathbf{\Sigma}_1)\,p(y=1) > p(\mathbf{x}\,|\,\mu_0,\mathbf{\Sigma}_0)\,p(y=0)$$

QDA: Quadratic decision boundary



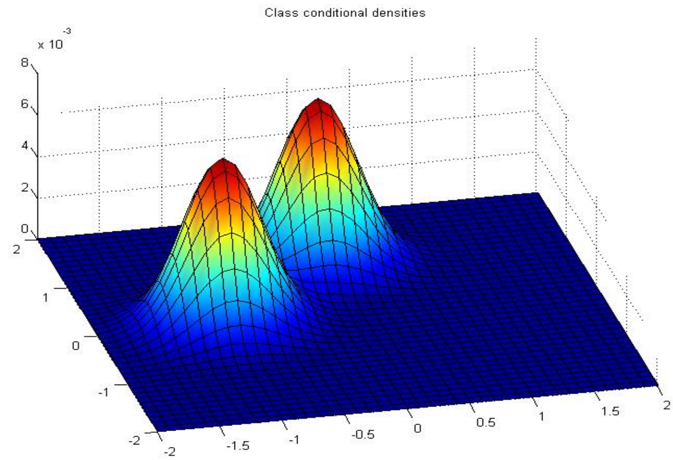
QDA: Quadratic decision boundary



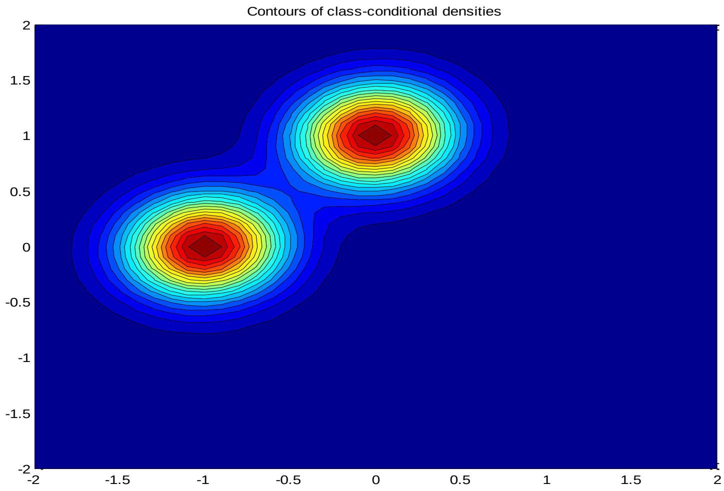
Linear discriminant analysis (LDA)

- **Assumes covariances** are the same $\mathbf{x} \sim N(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}),\ y = 0$

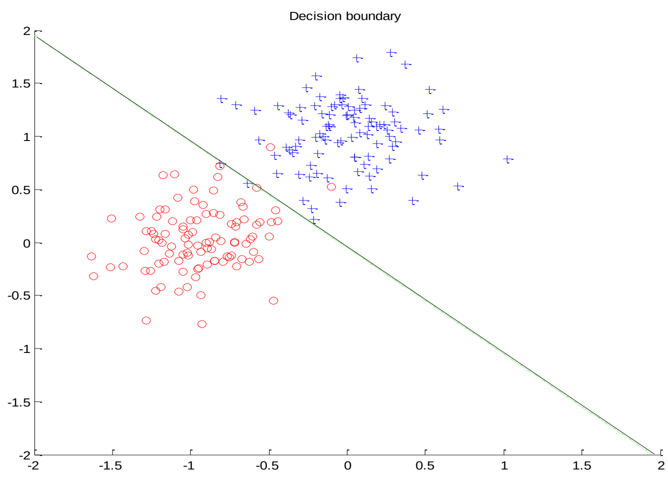
$$\mathbf{x} \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}),\ y = 1$$



LDA: Linear decision boundary



LDA: linear decision boundary



Generative classification models

Idea:

1. **Represent and learn the distribution** $p(\mathbf{x}, y)$
2. **Model is <u>able</u> to sample (generate) data instances** $(\mathbf{x}, y)$
3. **The model is used to get probabilistic discriminant functions** $g_o(\mathbf{x}) = p(y=0 \mid \mathbf{x})$ $\quad g_1(\mathbf{x}) = p(y=1 \mid \mathbf{x})$

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

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

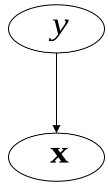
  binary classification: two class-conditional distributions

  $p(\mathbf{x} \mid y=0)$ $\qquad p(\mathbf{x} \mid y=1)$

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

  binary classification: Bernoulli distribution

  $p(y=0) + p(y=1) = 1$



Naïve Bayes classifier

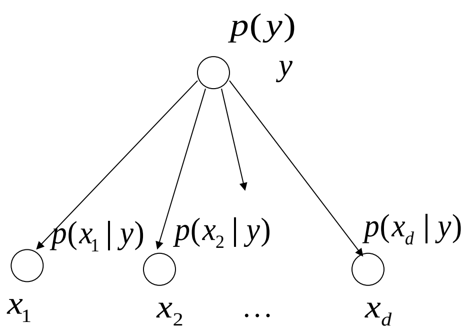
A generative classifier model with an additional simplifying assumption:

- **All input attributes are conditionally independent of each other given the class.**
- One of the basic ML classification models (often performs very well in practice)

  So we have:

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

$$p(\mathbf{x} \mid y) = \prod_{i=1}^{d} p(x_i \mid y)$$



Learning parameters of the model

Much simpler density estimation problems

- We need to learn:

  $p(\mathbf{x} \mid y=0)$ and $p(\mathbf{x} \mid y=1)$ and $p(y)$

- Because of the assumption of the conditional independence we need to learn:

  for every input variable i: $p(x_i \mid y=0)$ and $p(x_i \mid y=1)$

- **Much easier if the number of input attributes is large**
- **Also, the model gives us a flexibility to represent input attributes of different forms !!!**
- E.g. one attribute can be modeled using the Bernoulli, the other using Gaussian density, or a Poisson distribution

Making a class decision for the Naïve Bayes

Discriminant functions

- **Posterior of a class** – choose the class with better posterior probability

$$p(y=1 \mid \mathbf{x}) > p(y=0 \mid \mathbf{x}) \quad \text{then } y=1 \quad \text{else } y=0$$

$$p(y=1 \mid \mathbf{x}) = \frac{\left(\prod_{i=1}^{d} p(x_i \mid \Theta_{1,i})\right) p(y=1)}{\left(\prod_{i=1}^{d} p(x_i \mid \Theta_{1,i})\right) p(y=0) + \left(\prod_{i=1}^{d} p(x_i \mid \Theta_{2,i})\right) p(y=1)}$$

Evaluation of classifiers

Classification model learning

Learning:

- Many different ways and objective criteria used to learn the classification models. Examples:
  - Mean squared errors to learn the discriminant functions
  - Negative log likelihood (logistic regression)

Evaluation:

- One possibility: Use the same error criteria as used during the learning (apply to train & test data). Problems:
  - May work for discriminative models
  - Harder to interpret for humans.
- **Question:** how to more naturally evaluate the classifier performance?

Evaluation of classification models

For any data set we use to test the classification model on we can build a **confusion matrix:**

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

target

		$\omega = 1$	$\omega = 0$
predict	$\alpha = 1$	140	17
	$\alpha = 0$	20	54

Evaluation of classification models

Confusion matrix entries are often normalized with respect to the number of examples N to get proportions of the different agreements and disagreements among predicted and target values

target

		$\omega = 1$	$\omega = 0$
predict	$\alpha = 1$	$140/231$	$17/231$
	$\alpha = 0$	$20/231$	$54/231$

Basic evaluation statistics

Basic statistics calculated from the confusion matrix:

target

		$\omega = 1$	$\omega = 0$
predict	$\alpha = 1$	140	17
	$\alpha = 0$	20	54

Classification Accuracy = 194/231

Basic evaluation statistics

Basic statistics calculated from the confusion matrix:

target

		$\omega = 1$	$\omega = 0$
predict	$\alpha = 1$	140	17
	$\alpha = 0$	20	54

Classification Accuracy = 194/231

Misclassificion Error = 37/231 = 1 - **Accuracy**

Evaluation for binary classification

Entries in the confusion matrix for binary classification have names:

target

predict		$\omega=1$	$\omega=0$
	$\alpha=1$	TP	FP
	$\alpha=0$	FN	TN

TP: True positive (hit)
FP: False positive (false alarm)
TN: True negative (correct rejection)
FN: False negative (a miss)

Additional statistics

- **Sensitivity (recall)**

$$SENS = \frac{TP}{TP+FN}$$

- **Specificity**

$$SPEC = \frac{TN}{TN+FP}$$

- **Positive predictive value (precision)**

$$PPT = \frac{TP}{TP+FP}$$

- **Negative predictive value**

$$NPV = \frac{TN}{TN+FN}$$

Binary classification: additional statistics

Confusion matrix:

		target: 1	0	
predict	1	140	10	$PPV = 140/150$
	0	20	180	$NPV = 180/200$
		$SENS = 140/160$	$SPEC = 180/190$	

Row and column quantities:

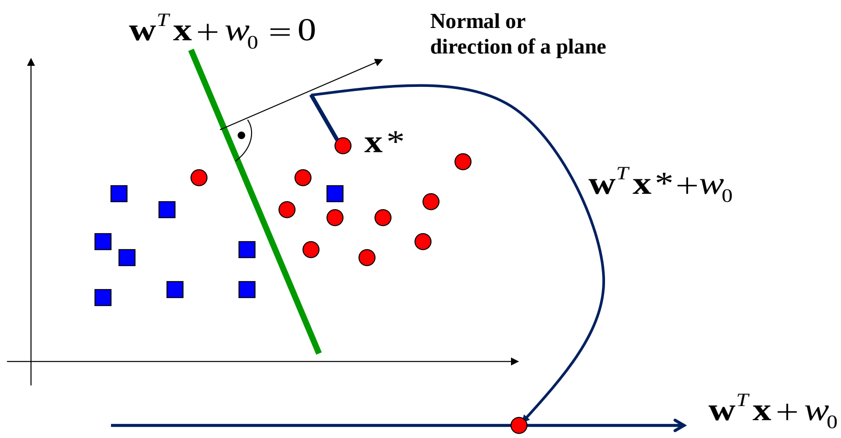
- Sensitivity (SENS)
- Specificity (SPEC)
- Positive predictive value (PPV)
- Negative predictive value (NPV)

F1 score:
harmonic mean of SENS and PPV

$$F1 = 2 * \frac{SENS * PPV}{SENS + PPV}$$

Binary classification models

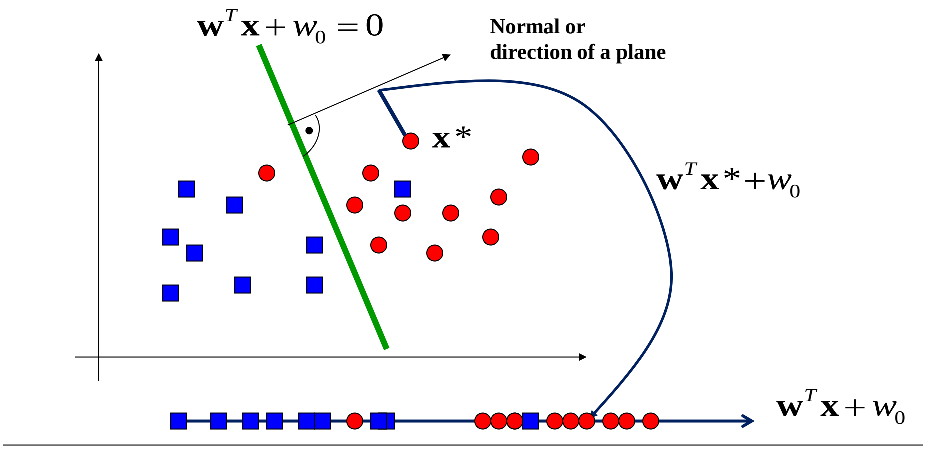
Often project data points to one dimensional space:
Defined for example by: $\mathbf{w}^T\mathbf{x}+w_0$ or $p(y=1|\mathbf{x},\mathbf{w})$



Binary classification models

Often project data points to one dimensional space:

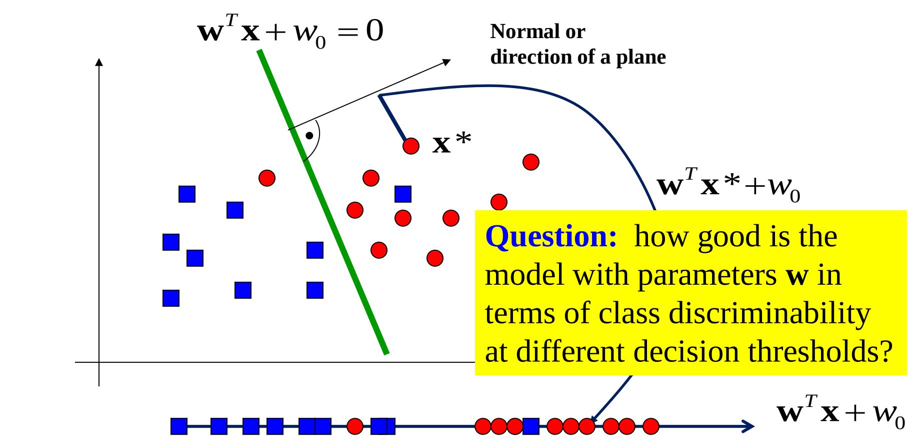
Defined for example by: $\mathbf{w}^T\mathbf{x}+w_0$ or p(y=1|**x**,**w**)



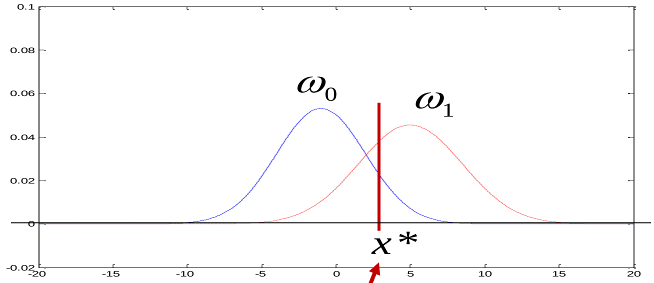
Binary classification models

Often project data points to one dimensional space:

Defined for example by: $\mathbf{w}^T\mathbf{x}+w_0$ or p(y=1|**x**,**w**)



Receiver Operating Characteristic (ROC)



- **Probabilities:**
  - *SENS* $p(x > x^* \mid \mathbf{x} \in \omega_1)$
  - *SPEC* $p(x < x^* \mid \mathbf{x} \in \omega_0)$

threshold

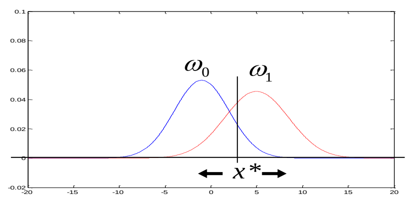
Receiver Operating Characteristic (ROC)

- **ROC curve plots :**

SN= $p(x > x^* \mid \mathbf{x} \in \omega_1)$

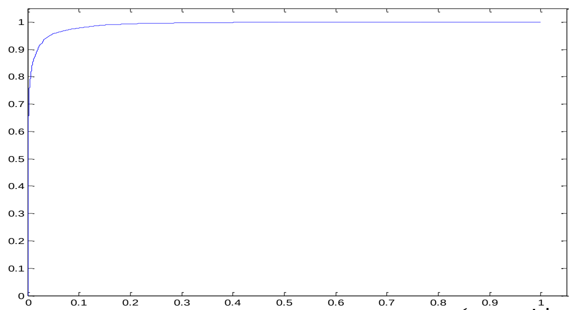
1-SP= $p(x > x^* \mid \mathbf{x} \in \omega_0)$

for different x*



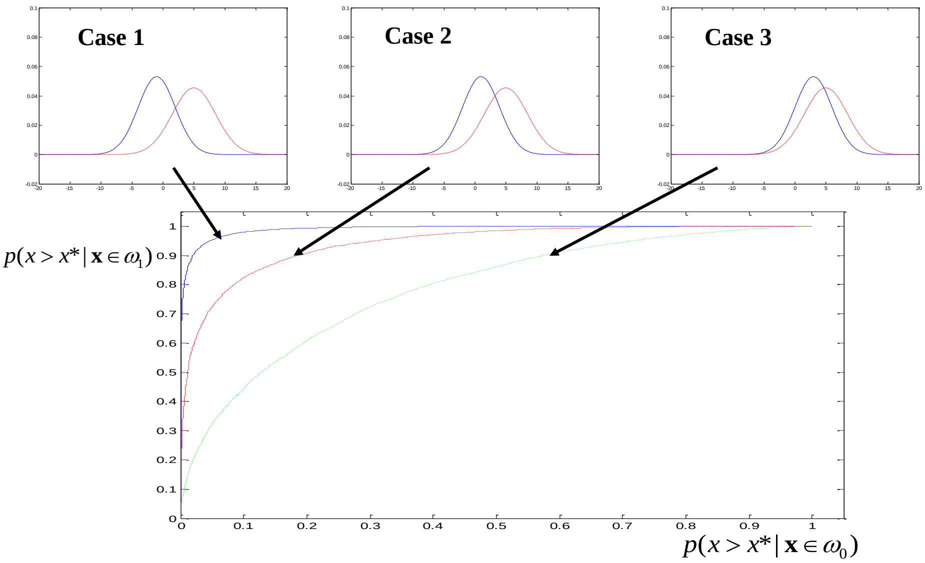
SENS

$p(x > x^* \mid \mathbf{x} \in \omega_1)$



1-SPEC $p(x > x^* \mid \mathbf{x} \in \omega_0)$

ROC curve



Receiver operating characteristic

- **ROC**
  - shows the discriminability between the two classes under different thresholds representing different decision biases
- **Decision bias**
  - can be changed using the different loss function
- **Quality of a classification model:**
  - Area under the ROC
  - Best value 1, worst (no discriminability): 0.5