CS 1675 Intro to Machine Learning
Lecture 11

Generative classification models

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 $X \rightarrow \Re$
  - When the decision on input **x** should be made choose the class with the highest value of $g_i(\mathbf{x})$

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

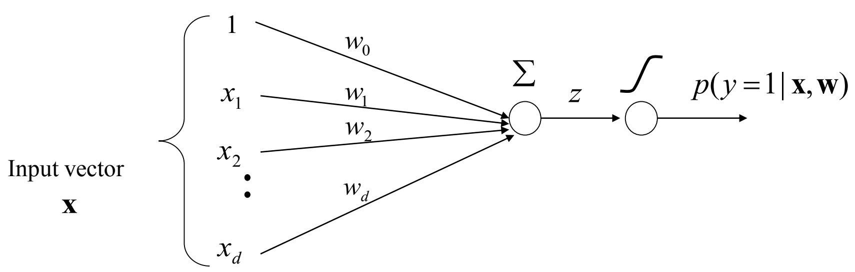
Logistic regression model

- **Discriminant functions:**

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

- **Values of discriminant functions vary in interval [0,1]**
  - **Probabilistic interpretation**

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



Logistic regression

- We learn **a probabilistic function**

$$f : X \rightarrow [0,1]$$

  - where f describes the probability of class 1 given $\mathbf{x}$

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

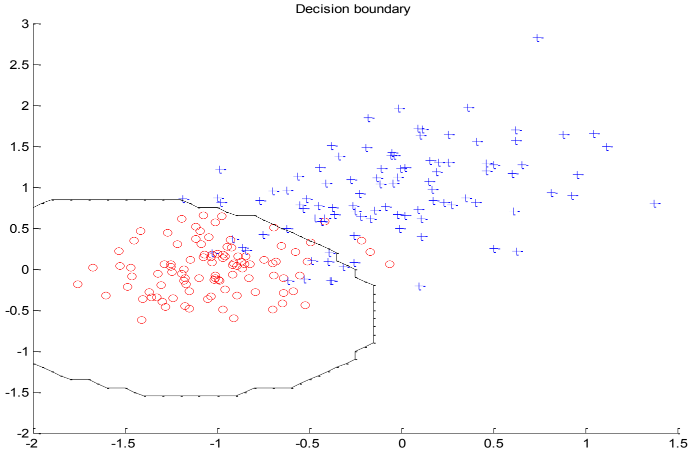
Note that: $$p(y = 0 \mid \mathbf{x}, \mathbf{w}) = 1 - p(y = 1 \mid \mathbf{x}, \mathbf{w})$$

- Making decisions with the logistic regression model:

If $p(y = 1 \mid \mathbf{x}) \geq 1/2$ then choose **1**
Else choose **0**

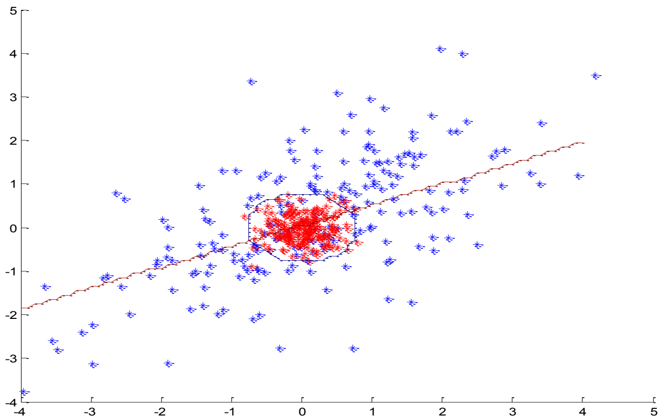
When does the logistic regression fail?

- Quadratic decision boundary is needed



When does the logistic regression fail?

- Another example of a non-linear decision boundary



Non-linear extension of logistic regression

- use **feature (basis) functions** to model **nonlinearities**
  - the same trick as used for the linear regression

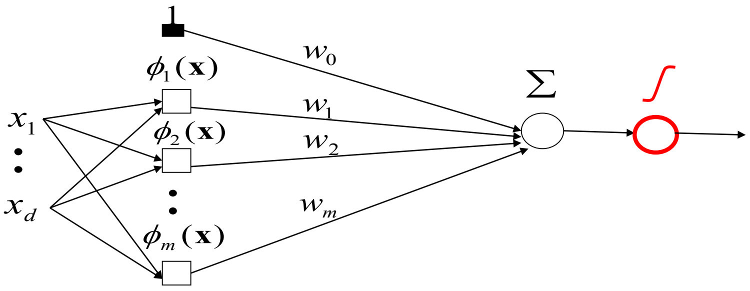
Linear regression

$$f(\mathbf{x}) = w_0 + \sum_{j=1}^{m} w_j \phi_j(\mathbf{x})$$

Logistic regression

$$f(\mathbf{x}) = g(w_0 + \sum_{j=1}^{m} w_j \phi_j(\mathbf{x}))$$

$\phi_j(\mathbf{x})$ - an arbitrary function of $\mathbf{x}$



Generative approach to classification

Logistic regression:

- **Represents and learns a model of** $p(y \mid \mathbf{x})$
- **An example of a discriminative approach**

Generative approach:

1. **Represents and learns the joint distribution** $p(\mathbf{x}, y)$
2. **Uses it to define probabilistic discriminant functions**

   E.g. $g_o(\mathbf{x}) = p(y = 0 \mid \mathbf{x}) \quad g_1(\mathbf{x}) = p(y = 1 \mid \mathbf{x})$

How? **Typically the joint is** $p(\mathbf{x}, y) = p(\mathbf{x} \mid y) p(y)$

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

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

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

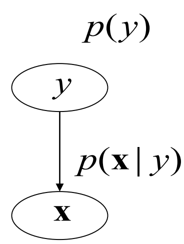
Generative approach to classification

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

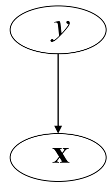
  $$p(y = 0) + p(y = 1) = 1$$

Quadratic discriminant analysis (QDA)

Model:

- **Class-conditional distributions**
  - **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 $\mathbf{x}$

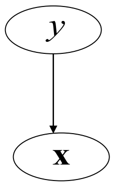
  Optimize log-likelihood: $l(D, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^{n} p(\mathbf{x}_i \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$

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

- How about **class priors**?

Learning Quadratic discriminant analysis (QDA)

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



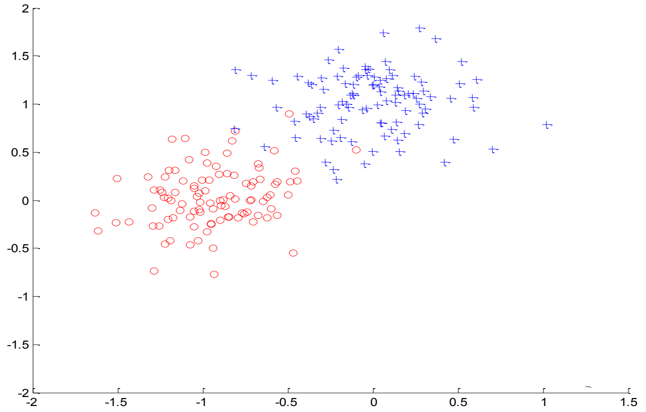
$$\mathbf{x} \sim N(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0) \quad \text{for} \quad y = 0$$
$$\mathbf{x} \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) \quad \text{for} \quad y = 1$$

  - **Use the density estimation methods**

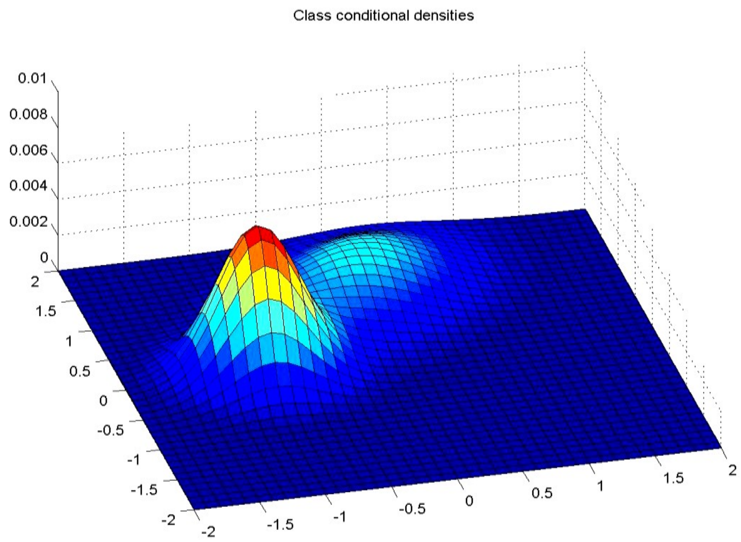
- **Learning Priors on classes (class 0,1)** $\quad y \sim Bernoulli$
  - **Learn the parameter of the Bernoulli distribution**
  - **Again use the density estimation methods**

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

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

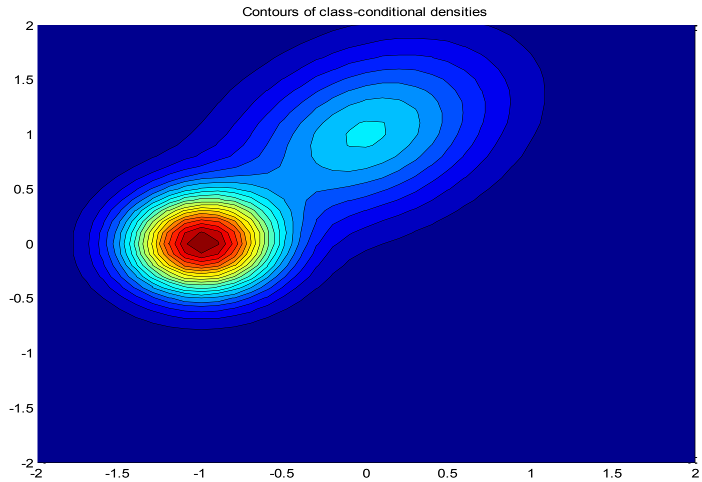
$$\underbrace{p(y=1\,|\,\mathbf{x})}_{g_1(\mathbf{x})} > \underbrace{p(y=0\,|\,\mathbf{x})}_{g_0(\mathbf{x})} \quad \Rightarrow \quad \text{then } y=1 \text{ else } y=0$$

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

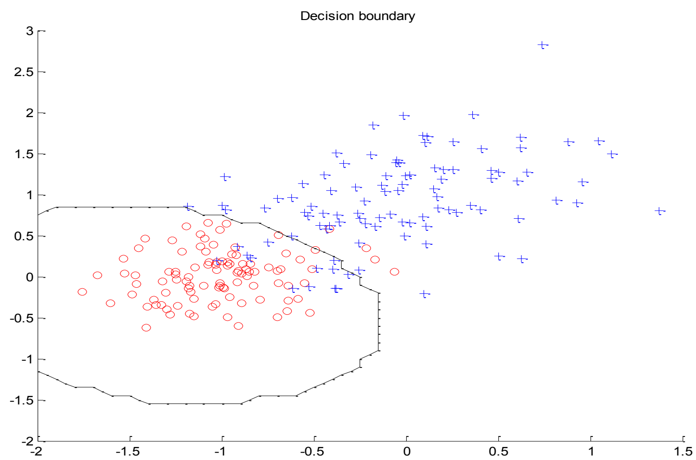
- **It is sufficient to compare:**

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

QDA: Quadratic decision boundary



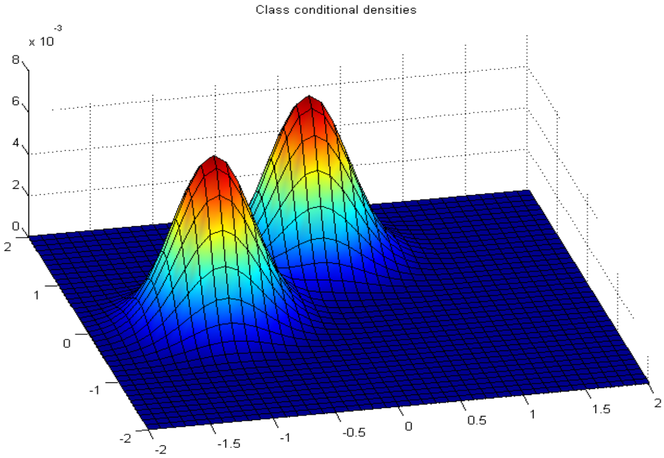
QDA: Quadratic decision boundary



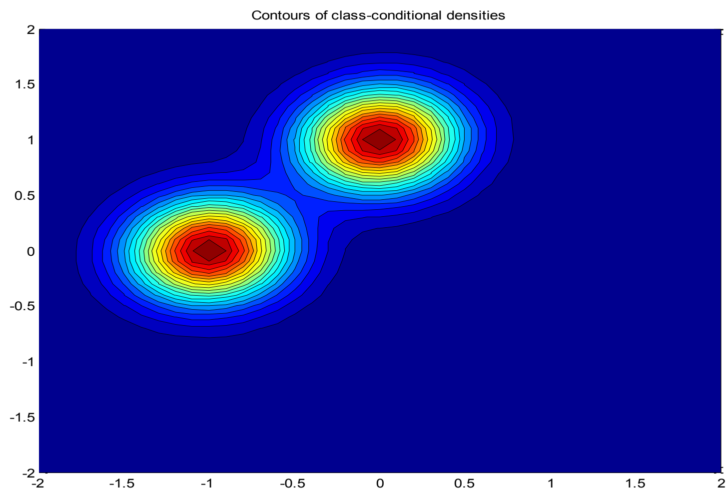
Linear discriminant analysis (LDA)

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

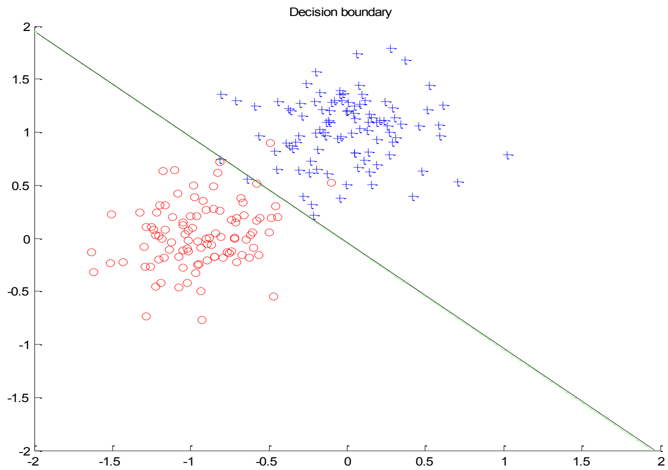
$$\mathbf{x} \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}),\ y = 1$$



LDA: Linear decision boundary



LDA: linear decision boundary



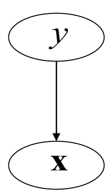
Generative classification models

Idea:

1. **Represent and learn the distribution** $p(\mathbf{x}, y)$
2. **Use it to define probabilistic discriminant functions**

   E.g. $g_o(\mathbf{x}) = p(y = 0 \mid \mathbf{x}) \quad g_1(\mathbf{x}) = p(y = 1 \mid \mathbf{x})$

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

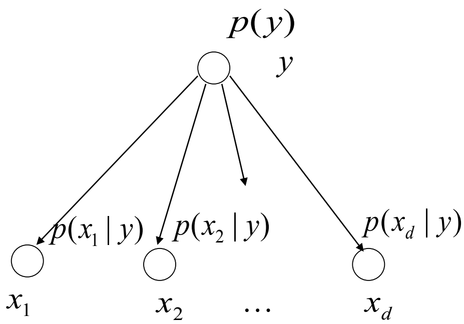
A generative classifier model with an additional simplifying assumption

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

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 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 as Gaussian density, or as a Poisson distribution

Making a class decision for the Naïve Bayes

Discriminant functions

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

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

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

Next: two interesting questions

(1) Two models with linear decision boundaries:

- **Logistic regression**
- **LDA model (2 Gaussians with the same covariance matrices**

$$x \sim N(\mu_0, \Sigma) \quad \text{for} \quad y=0$$
$$x \sim N(\mu_1, \Sigma) \quad \text{for} \quad y=1$$

- **Question: Is there any relation between the two models?**

(2) Two models with the same gradient:

- **Linear model for regression**
- **Logistic regression model for classification**

have the same gradient update

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))\mathbf{x}_i$$

- **Question: Why is the gradient the same?**

Logistic regression and generative models

- **Two models with linear decision boundaries:**
  - **Logistic regression**
  - **Generative model with 2 Gaussians with the same covariance matrices** $x \sim N(\mu_0, \Sigma)$ for $y = 0$

    $x \sim N(\mu_1, \Sigma)$ for $y = 1$

Question: Is there any relation between the two models?

Answer: Yes, the two models are related !!!

- When we have **2 Gaussians with the same covariance matrix** the probability of y given **x** has the form of a logistic regression model **!!!**

$$p(y = 1 \mid \mathbf{x}, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) = g(\mathbf{w}^T \mathbf{x})$$

Logistic regression and generative models

- **Members of the exponential family can be often more naturally described as**

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

$\boldsymbol{\theta}$ - A location parameter $\boldsymbol{\varphi}$ - A scale parameter

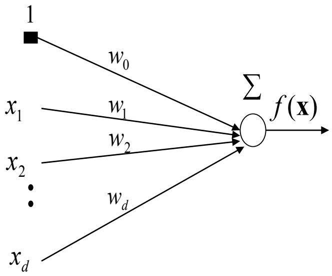
- **Claim:** A **logistic regression** is a correct model when class conditional densities are from the same distribution in the exponential family and have **the same scale factor** $\boldsymbol{\varphi}$
- **Very powerful result !!!!**
  - **We can represent posteriors of many distributions with the same small logistic regression model**

CS 2750 Machine Learning

The gradient puzzle …

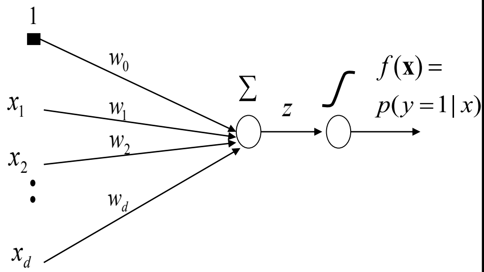
Linear regression

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



Logistic regression

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



Gradient update:

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))\mathbf{x}_i$$

Online: $\mathbf{w} \leftarrow \mathbf{w} + \alpha (y - f(\mathbf{x}))\mathbf{x}$

The same

Gradient update:

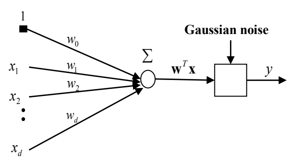
$$\mathbf{w} \leftarrow \mathbf{w} + \alpha \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))\mathbf{x}_i$$

Online: $\mathbf{w} \leftarrow \mathbf{w} + \alpha (y - f(\mathbf{x}))\mathbf{x}$

The gradient puzzle …

- The **same simple gradient update rule** derived for both the linear and logistic regression models
- Where the magic comes from?
- Under the **log-likelihood** measure the function models and the models for the output selection fit together:
  - **Linear model + Gaussian noise**

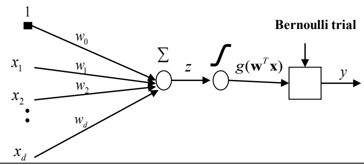
    $$y = \mathbf{w}^T \mathbf{x} + \varepsilon \qquad \varepsilon \sim N(0, \sigma^2)$$



  - **Logistic + Bernoulli**

    $$y = \text{Bernoulli}(\theta)$$

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



Generalized linear models (GLIMs)

Assumptions:

- The conditional mean (expectation) is:

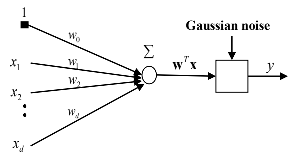
$$\mu = f(\mathbf{w}^T \mathbf{x})$$

  - Where $f(.)$ is a **response function**
- Output y is characterized by an exponential family distribution with a conditional mean μ

Examples:

- **Linear model + Gaussian noise**

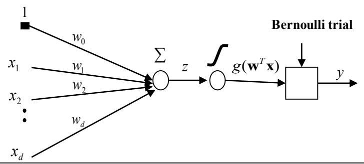
$$y = \mathbf{w}^T \mathbf{x} + \varepsilon \qquad \varepsilon \sim N(0, \sigma^2)$$



- **Logistic + Bernoulli**

$$y \approx \text{Bernoulli}(\theta)$$

$$\theta = g(\mathbf{w}^T \mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^T \mathbf{x}}}$$



Generalized linear models (GLIMs)

- **A canonical response functions $f(.)$:**
  - **encoded in the sampling distribution**

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

- **Leads to a simple gradient form**
- **Example: Bernoulli distribution**

$$p(x \mid \mu) = \mu^x (1-\mu)^{1-x} = \exp\left\{ \log\left(\frac{\mu}{1-\mu} \right) x + \log(1-\mu) \right\}$$

$$\theta = \log\left(\frac{\mu}{1-\mu} \right) \qquad \mu = \frac{1}{1 + e^{-\theta}}$$

  - **Logistic function matches the Bernoulli**

Evaluation of classifiers
ROC

Evaluation

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

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

Evaluation

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

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

Evaluation

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

target

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

Accuracy = 194/231

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

Evaluation for binary classification

Entries in the confusion matrix for binary classification have names:

target

		$\omega = 1$	$\omega = 0$
predict	$\alpha = 1$	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**

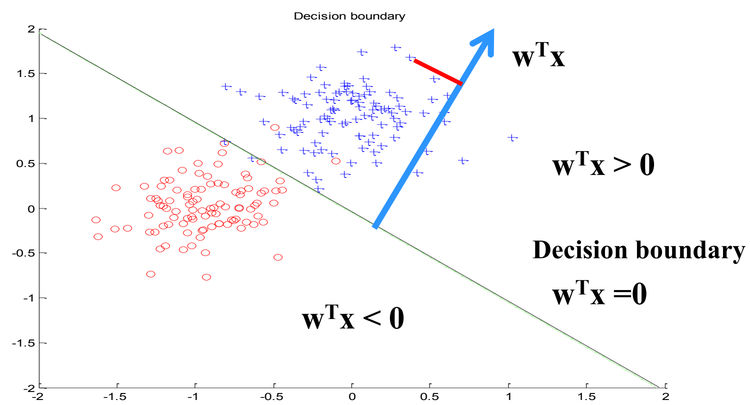
		target		
		1	0	
predict	1	140	10	$PPV = 140/150$
	0	20	180	$NPV = 180/200$
		$SENS = 140/160$	$SPEC = 180/190$	

Row and column quantities:
- Sensitivity (SENS)
- Specificity (SPEC)
- Positive predictive value (PPV)
- Negative predictive value (NPV)

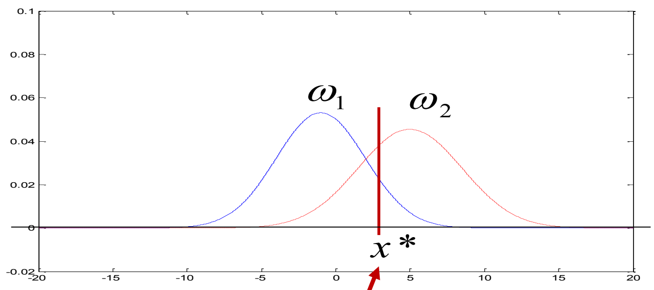
Classifiers

Project datapoints to one dimensional space:

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



Binary decisions: Receiver Operating Curves



- **Probabilities:**
  - *SENS* $p(x > x^* \mid \mathbf{x} \in \omega_2)$
  - *SPEC* $p(x < x^* \mid \mathbf{x} \in \omega_1)$

threshold

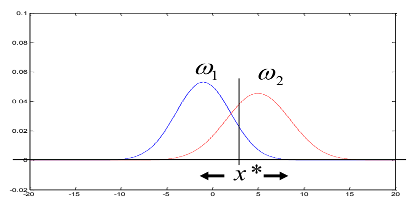
Receiver Operating Characteristic (ROC)

- **ROC curve plots :**

SN= $p(x > x^* \mid \mathbf{x} \in \omega_2)$

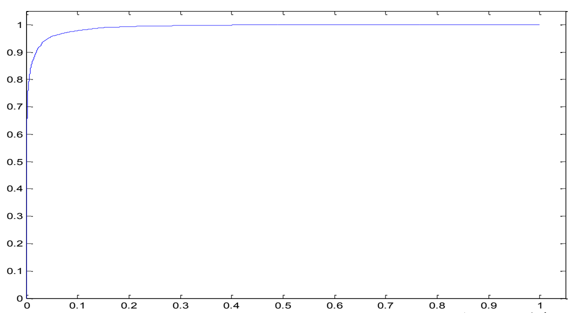
1-SP= $p(x > x^* \mid \mathbf{x} \in \omega_1)$

for different x*



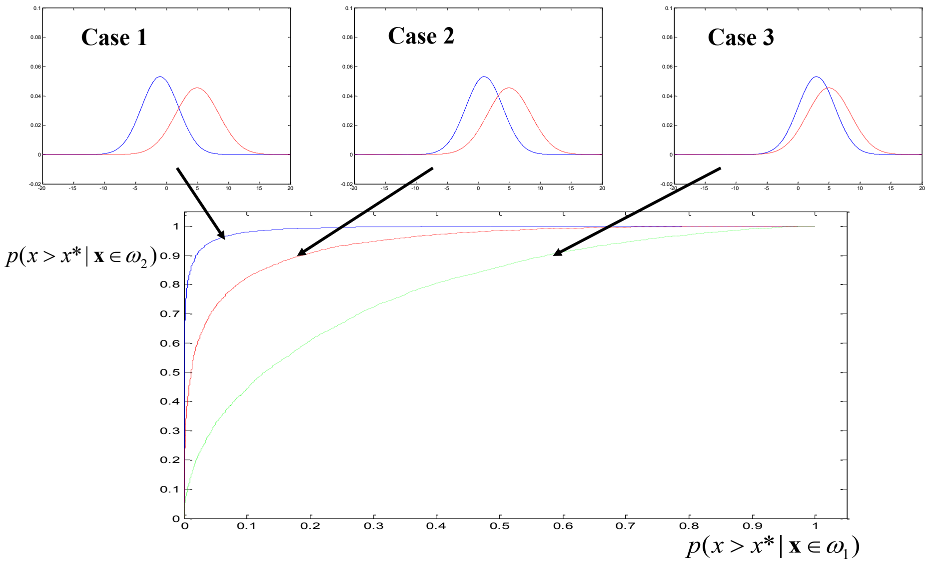
SENS

$p(x > x^* \mid \mathbf{x} \in \omega_2)$



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

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
- **Quality of a classification model:**
  - Area under the ROC
  - Best value 1, worst (no discriminability): 0.5