Module 3:

Linear Models for Classification

(Part B)

Lecture Objectives

Learn Probabilistic Generative Models

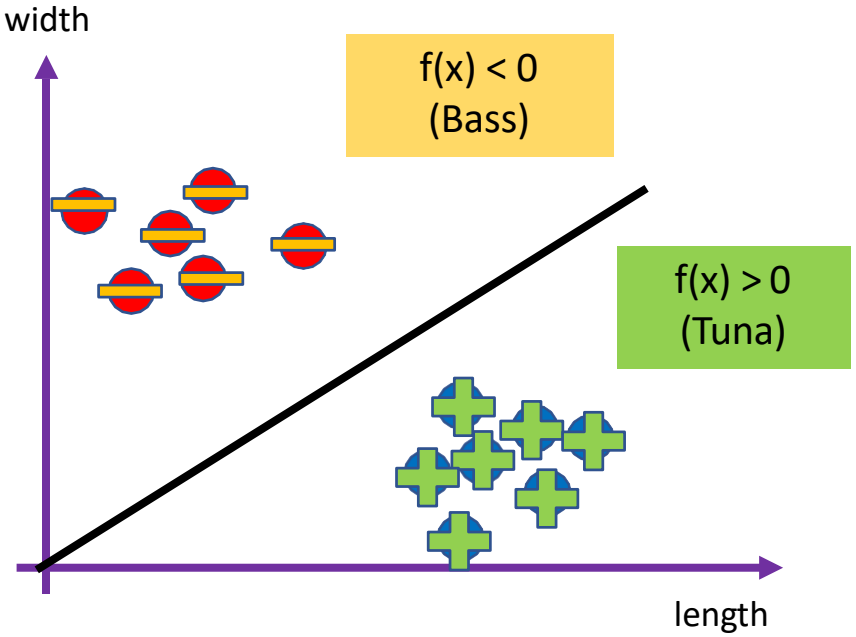
Learn Probabilistic Discriminative Models

Probabilistic Generative Models

Discriminative vs Generative Classifiers

Discriminative classifiers

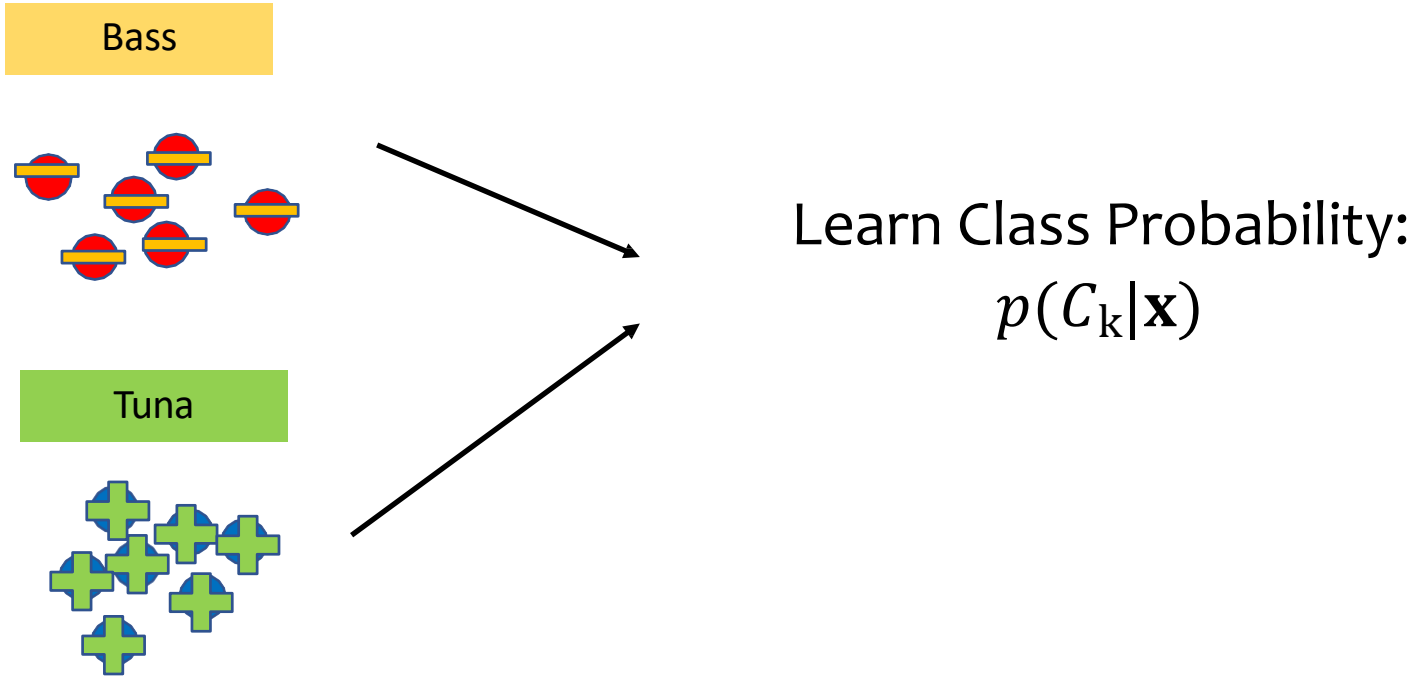
- **Approach 1:** Search for a decision boundary that separates class labels.



Discriminative vs Generative Classifiers

Discriminative classifiers

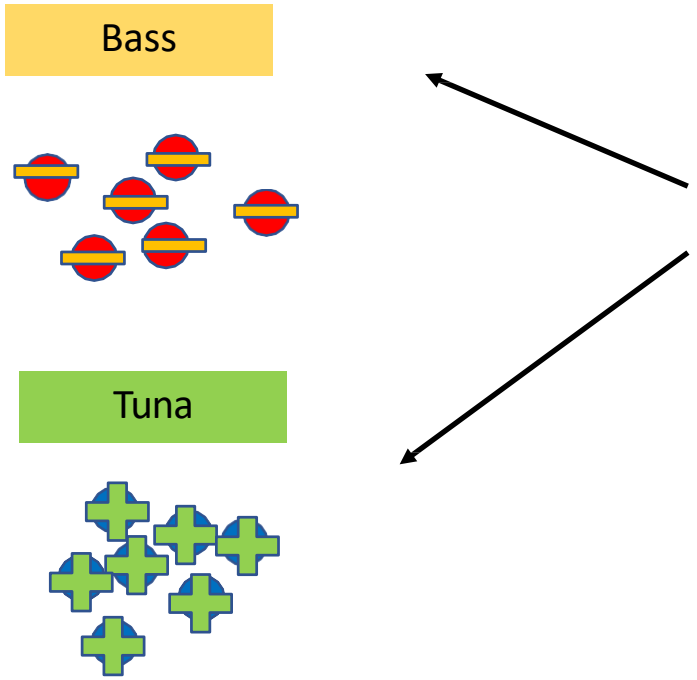
- **Approach 2:** Try to learn $p(C_k|\mathbf{x})$ directly



Discriminative vs Generative Classifiers

Generative classifiers

- Try to model $p(\mathbf{x}|C_{\mathrm{k}})$ (How does data look like for a class C_{k})
  – this can actually be used to **generate** the input data



Model:

- $p(\mathbf{x}|y = \mathrm{Tuna})$: model the distributions of features of Tuna
- $p(\mathbf{x}|y = \mathrm{Bass})$: model the distributions of features of Bass

Probabilistic Generative Models

- Classification via Bayes' rule (also called **Bayes classifier**)

  Idea: To get $p(C_k|\mathbf{x})$, use $p(\mathbf{x}|C_k)$ & $p(C_k)$ via Bayes' theorem:

$$p(C_k|\mathbf{x}) = \frac{p(\mathbf{x}|C_k)p(C_k)}{p(\mathbf{x})}$$

  - **Prediction Rule**: If we calculate $p(C_k|\mathbf{x})$ in order to make a prediction, we don't have to actually need to calculate denominator, since

$$\begin{aligned}\text{argmax}_{C_k} p(C_k|\mathbf{x}) &= \text{argmax}_{C_k} \frac{p(\mathbf{x}|C_k)p(C_k)}{p(\mathbf{x})} \\ &= \text{argmax}_{C_k} p(\mathbf{x}|C_k)p(C_k)\end{aligned}$$

Class-Conditional Probability Density Function (PDF)

Class-Conditional Probability Density Function (PDF)

- Let x be a continuous random variable (e.g. length in fish)

- $p(x|C_{\mathrm{k}})$ – the **class-conditional probability density function (PDF)** - the probability of x given that the state of nature of C_{k}

- Example: $p(length|\mathrm{Tuna})$ and $p(length|\mathrm{Bass})$ describe the differences in **length** between populations of Tuna and Bass

Gaussian Discriminative Analysis

Gaussian Discriminant Analysis

- A generative learning model assuming that class-conditional PDF $p(\mathbf{x}|C_k)$ is distributed according to a **normal (Gaussian) distribution.**
- This classifier is also called **Gaussian Bayes Classifier**

A simple case is when inputs are just 1-dimensional:

$$p(\mathbf{x}|C_k) = \frac{1}{\sqrt{2\pi}\sigma_{C_k}} \exp\left(-\frac{(\mathbf{x}-\mu_{C_k})^2}{2\sigma_{C_k}^2}\right)$$

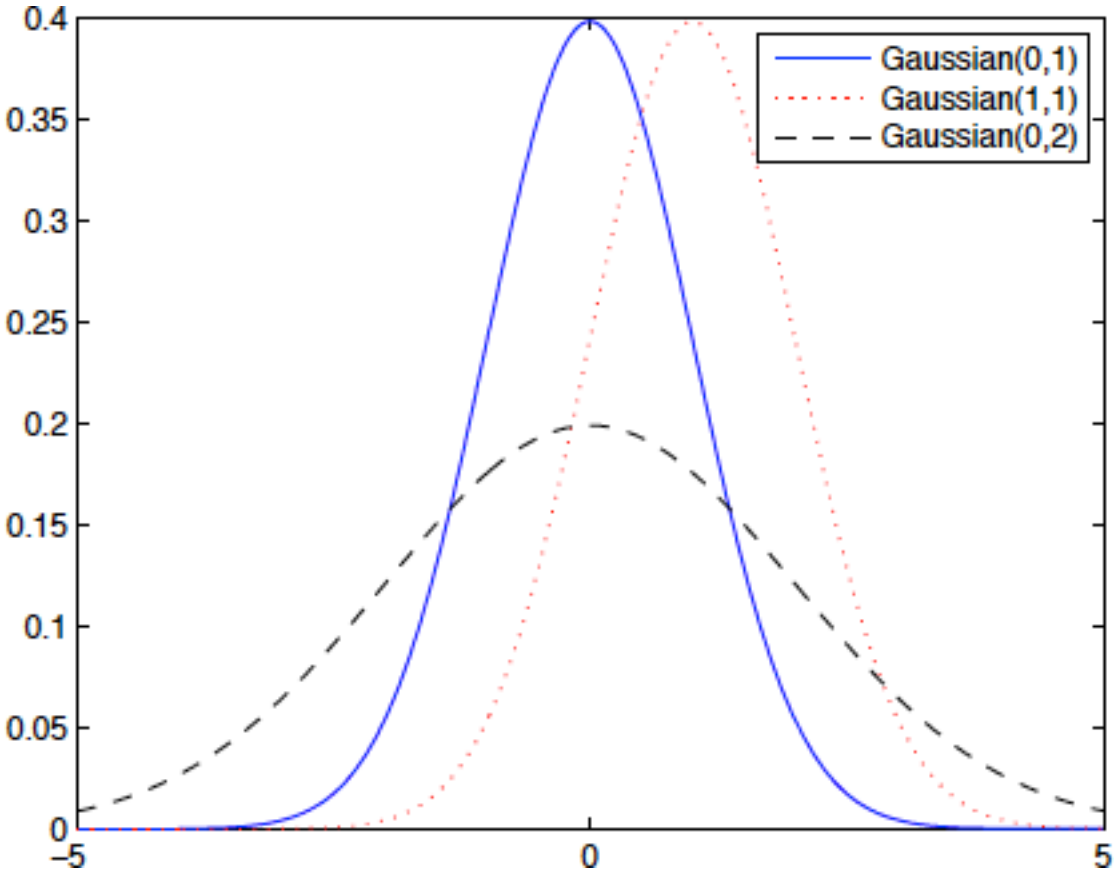
with **parameters**:

- mean μ_{C_k}
- variance $\sigma_{C_k}^2$

- Note that we have different parameters for different classes

Univariate Gaussian Distribution

Probability



Random Variable

Fitting a Gaussian Distribution to Data

Assume that the class-conditional PDF is a Gaussian:

$$p(\mathbf{x}|C_k) = \frac{1}{\sqrt{2\pi}\sigma_{C_k}} \exp\left(-\frac{(\mathbf{x}-\mu_{C_k})^2}{2\sigma_{C_k}^2}\right)$$

How to fit a Gaussian distribution to the training data?

- Given training examples $\{x_n, t_n\}_{n=1,\ldots N}$ with $t_n \in \{C_1, C_2\}$, we need to estimate the **model parameters** $\{(\mu_{C_1}, \sigma_{C_1}^2), (\mu_{C_2}, \sigma_{C_2}^2)\}$
- **Divide** the training set $\mathcal{D}$ into two classes: $\mathcal{D}_1$ and $\mathcal{D}_2$
- For each class C_k, we need to fit a **Gaussian** to model $p(x|C_k)$ **on** $\boldsymbol{\mathcal{D}_k}$

Finding parameters using MLE in Gaussians

How to find parameters that fit a Gaussian distribution to my training data?

- Try **Maximum Likelihood Estimation (MLE)** for a Gaussian

$$p(\mathbf{x_1}, \ldots, \mathbf{x_n}|C_k) = \prod_{n=1}^{N} p(\mathbf{x_n}|C_k) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mathbf{x_n} - \mu)^2}{2\sigma^2}\right)$$

Note: for simplicity of notation, we drop subscript C_k

What's the next step?

- Maximise the likelihood, or minimise its negative:

$$-\ln p(\mathbf{x_1}, \ldots, \mathbf{x_n}|C_k) = -\ln\left(\prod_{n=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mathbf{x_n} - \mu)^2}{2\sigma^2}\right)\right) =$$

$$\sum_{n=1}^{N} \ln(\sqrt{2\pi}\sigma) + \sum_{n=1}^{N} \frac{(\mathbf{x_n} - \mu)^2}{2\sigma^2}$$

Finding parameters using MLE in Gaussians

To find the parameters μ and σ^2, let's use the derivatives with respect to μ and σ^2 and set them to zero:

$$\frac{\partial(-\ln p(\mathbf{x}_1, ..., \mathbf{x}_n|C_k))}{\partial\mu} = 0 \quad \Rightarrow \quad \mu = \frac{1}{N}\sum_{n=1}^{N} x_n$$

$$\frac{\partial(-\ln p(\mathbf{x}_1, ..., \mathbf{x}_n|C_k))}{\partial\sigma^2} = 0 \quad \Rightarrow \quad \sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2$$

Finding parameters using MLE in Gaussians

In summary, we can compute the parameters of a Gaussian distribution for each class C_k by using the training data points $\mathcal{D}_k$ associated to classes

MLE estimates of parameters for a Gaussian distribution:

$$\mu_{C_k} = \frac{1}{N}\sum_{n=1}^{N} x_n$$

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu_{C_k})^2$$

How to use parameters? Remember

$$p(\mathbf{x}|C_k) \approx p(\mathbf{x}|\mu_{C_k}, \sigma^2_{C_k})$$

Posterior Probability

Compute the posterior probability:

- Given a new sample **x**, we choose C_1**:**

$$p(C_1|\mathbf{x}) > p(C_2|\mathbf{x}) \iff p(\mathbf{x}|C_1)p(C_1) > p(\mathbf{x}|C_2)p(C_2)$$

How to calculate $p(C_k)$?

$$p(C) = \begin{cases} \phi, & \text{if } C = C_k \\ 1-\phi, & \text{otherwise} \end{cases}$$

Bernoulli Distribution:

$$p(C_k) = \phi^{C_k}(1-\phi)^{1-C_k}$$

C_k is 1 if it's the class with probability Φ
C_k is 0 otherwise

Summary (Our learning problem)

What is our learning objective?

- Given the training set, learn the parameters to fully specify the joint distribution $p(\mathbf{x}, C_k)$

What are the model parameters?

$$\mathbf{w} = (\phi, \mu_{C_k}, \sigma^2_{C_k})$$

How to define the likelihood function of the joint distribution:

$$\prod_{n=1}^{N} p(\mathbf{x_n}, C_k; \mu_{C_k}, \sigma^2_{C_k}, \phi) = \prod_{n=1}^{N} p(\mathbf{x_n}|C_k; \mu_{C_k}, \sigma^2_{C_k}) p(C_k; \phi)$$

Summary (Our learning problem)

How to find such parameters?

- Apply log and calculate the partial derivative with respect to each of the parameters

$$\prod_{n=1}^{N} p(\mathbf{x_n}, C_k; \mu_{C_k}, \sigma^2_{C_k}, \phi) = \prod_{n=1}^{N} p(\mathbf{x_n}|C_k; \mu_{C_k}, \sigma^2_{C_k}) p(C_k; \phi)$$

- Do this separately since the left and right probabilities depend on different parameters.

$p(\mathbf{x_n}|C_k; \mu_{C_k}, \sigma^2_{C_k})$ ⇒ Previous Slides!

$p(C_k; \phi)$ ⇒ $\phi = \frac{1}{N} \sum_{n=1}^{N} 1\{t_n = C_k\}$

Probabilistic Generative Models via Multivariate Gaussian Distribution

Gaussian Discriminant Analysis for Multivariate Inputs

Gaussian Discriminant Analysis (Gaussian Bayes Classifier)

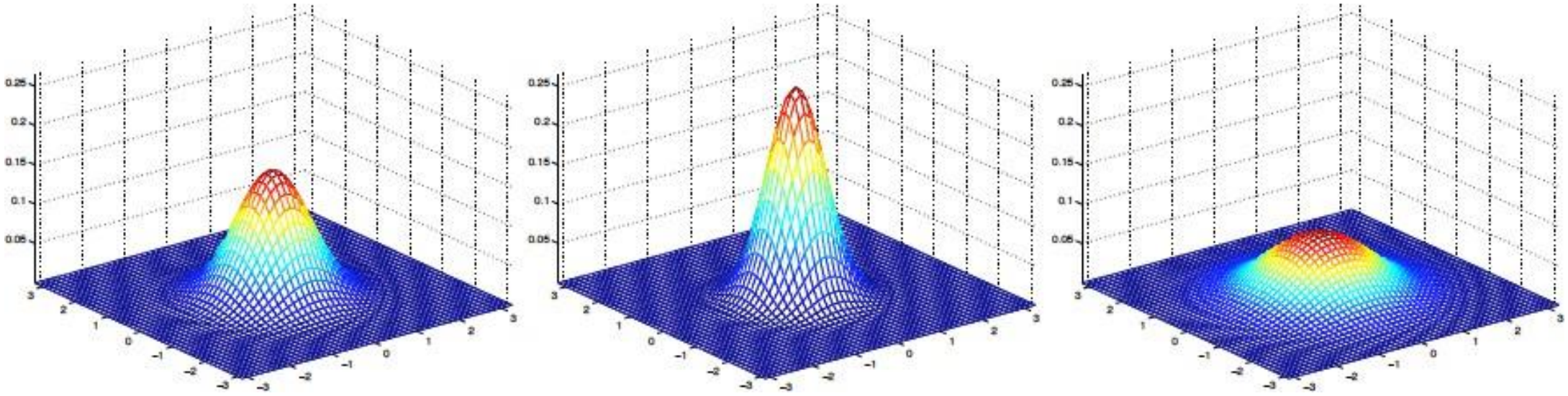
- Assume that the class-conditional PDF $p(\mathbf{x}|C_\mathrm{k})$ is distributed according to a **multivariate normal (Gaussian) distribution.**
- $\mathbf{x}$: a vector-valued random variable $\mathbf{x} = (x_1, \dots, x_\mathrm{D})$

$$p(\mathbf{x}|C_k) \approx p(\mathbf{x}|\boldsymbol{\mu}_{C_k}, \boldsymbol{\Sigma}) =$$

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

- Parameterized by k D-dimensional **mean vector $\boldsymbol{\mu}_{C_k}$** and a D x D **covariance matrix $\boldsymbol{\Sigma}$**, $|\boldsymbol{\Sigma}|$ is the determinant of $\boldsymbol{\Sigma}$
- Note: each class has a different $\mu_{\mathrm{C_k}}$, but all share the same $\boldsymbol{\Sigma}$!

Multivariate Gaussian Distribution Densities



Posterior Probability

Interestingly, the posterior probability takes the following form:

- For the case with K=2,

$$p(C_1|\mathbf{x}) = \frac{1}{1+\exp^{-(\mathbf{w}\cdot\mathbf{x}+w_0)}}$$

$$= \sigma(\mathbf{w}\cdot\mathbf{x}+w_0), \text{with}$$

$$\mathbf{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_{C_1} - \boldsymbol{\mu}_{C_2})$$

$$w_0 = -\frac{1}{2}\boldsymbol{\mu}_{C_1}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_{C_1} + \frac{1}{2}\boldsymbol{\mu}_{C_2}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_{C_2} + \ln\frac{p(C_1)}{p(C_2)}$$

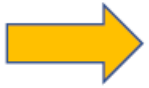
Finding parameters using MLE in Gaussians (Binary)

What is our objective?

- Given training examples $\{\mathbf{x}_n, t_n\}_{n=1,\ldots N}$ with $t_n \in \{C_0, C_1\}$, we need to **estimate the model parameters**

What are the model parameters in the model for multivariate features?

$$p(\mathbf{x}|C_k) \approx p(\mathbf{x}|\boldsymbol{\mu}_{\boldsymbol{C_k}}, \boldsymbol{\Sigma})$$
$$p(\mathbf{x}|C_k) = p(\mathbf{x}|C_k; \boldsymbol{\mu}_{\boldsymbol{C_k}}, \boldsymbol{\Sigma})$$



$$\mathbf{w} = (\boldsymbol{\mu}_{C_k}, \Sigma)$$

$$p(C) = \begin{cases} \phi, & \text{if } C = C_k \\ 1 - \phi, & \text{otherwise} \end{cases}$$

$$\mathbf{w} = (\phi, \boldsymbol{\mu}_{C_k}, \Sigma)$$

Finding parameters using MLE in Gaussians (Binary)

Find the parameters that satisfy the following:

$$\text{argmax}_{\mathbf{w}} \prod_{n=1}^{N} p(\mathbf{x}_n|C_k;\boldsymbol{\mu}_{C_k},\Sigma)p(C_k;\phi)$$

$$\varphi = \frac{N_1}{N_1+N_2}$$

$$\boldsymbol{\mu}_1 = \frac{1}{N_1}\sum_{n=1}^{N} t_n \boldsymbol{x}_n$$

$$\boldsymbol{\mu}_2 = \frac{1}{N_2}\sum_{n=1}^{N} (1-t_n)\boldsymbol{x}_n$$

$$\boldsymbol{\Sigma} = \frac{N_1}{N}\mathbf{S}_1 + \frac{N_2}{N}\mathbf{S}_2$$

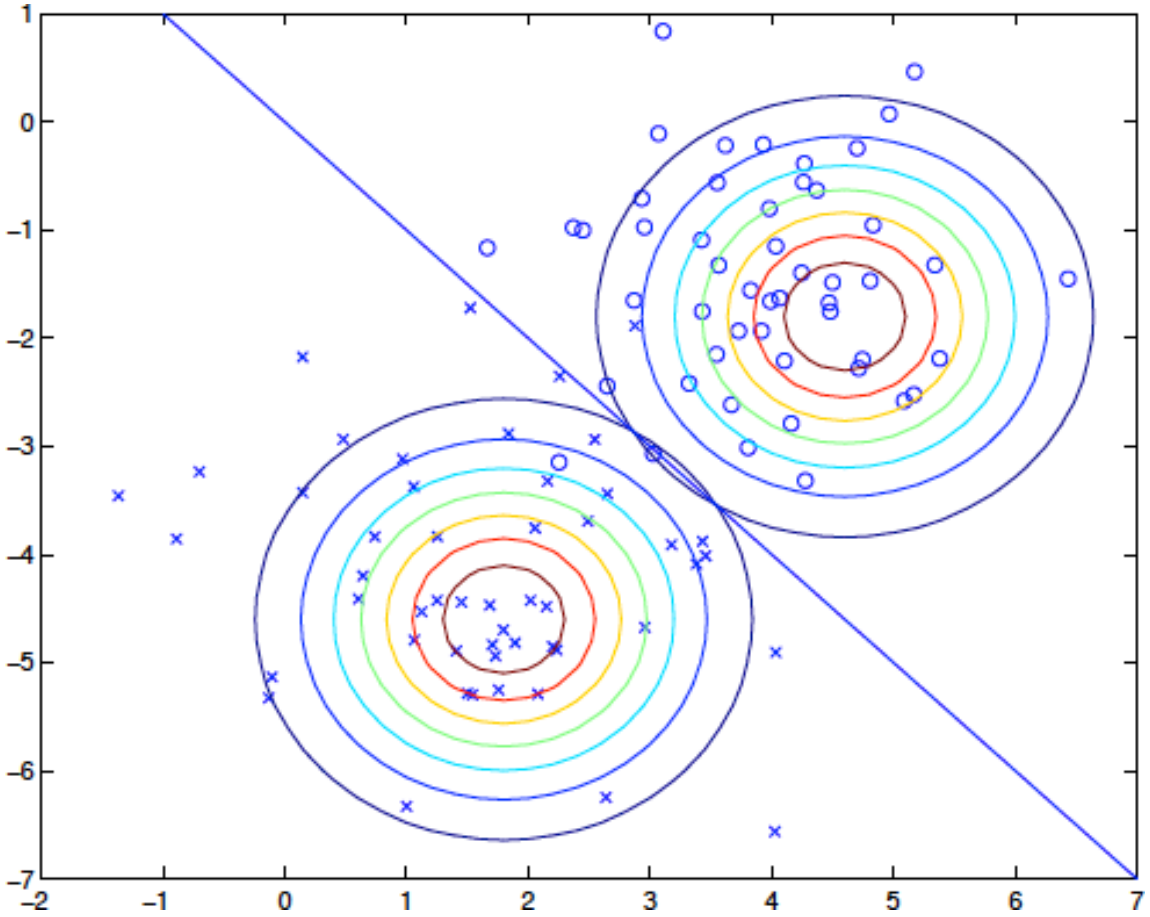
$$\mathbf{S}_1 = \frac{1}{N_1}\sum_{n\in\mathcal{C}_1}(\mathbf{x}_n-\mu_1)(\mathbf{x}_n-\mu_1)^T$$

$$\mathbf{S}_2 = \frac{1}{N_2}\sum_{n\in\mathcal{C}_2}(\mathbf{x}_n-\mu_2)(\mathbf{x}_n-\mu_2)^T$$

$t_n = 1$, if $\boldsymbol{x}_n$ is in C_1

$t_n = 0$, if $\boldsymbol{x}_n$ is in C_2

Visualization



Prediction Rule

Simply:

$$p(C_1|\mathbf{x}) > p(C_2|\mathbf{x}) \Rightarrow p(\mathbf{x}|C_1)p(C_1) > p(\mathbf{x}|C_2)p(C_2)$$

Alternatively:

$$a = \ln \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_2)p(C_2)}$$

Predict C_1 if $a > 0$, and C_2 otherwise

Decision Boundary

Let

$$a = \ln \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_2)p(C_2)}$$

Plug the Gaussian class densities into the variable "a":

$$a = \ln \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_2)p(C_2)}$$

$$p_{\boldsymbol{\theta}}(x|\mathcal{C}_k) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)\right]$$

$$= \ln \frac{p(\mathbf{x}|C_1)}{p(\mathbf{x}|C_2)} + \ln \frac{p(C_1)}{p(C_2)}$$

$$= \ln \frac{\boldsymbol{exp}[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1)]}{\boldsymbol{exp}[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_2)^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2)]} + \ln \frac{p(C_1)}{p(C_2)}$$

$$= \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_2)^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) + \ln \frac{p(C_1)}{p(C_2)}$$

Decision Boundary

$$a = \ln \frac{\boldsymbol{exp}[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1)]}{\boldsymbol{exp}[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_2)^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2)]} + \ln \frac{p(C_1)}{p(C_2)}$$

$$= \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_2)^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) + \ln \frac{p(C_1)}{p(C_2)}$$

Where we note that the quadratic term $\boldsymbol{x}^T\Sigma^{-1}\boldsymbol{x}$ is cancelled.
Hence a takes a simple linear form

$$a = \boldsymbol{w}\boldsymbol{x} + w_0$$

$$\boldsymbol{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

$$w_0 = \frac{1}{2}\boldsymbol{\mu}_2^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 - \frac{1}{2}\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \ln \frac{p(C_1)}{p(C_2)}$$

Note: “a” takes a simple linear form. This means the induced decision boundary is linear.

Logistic Regression

Logistic Regression

Logistic Regression?

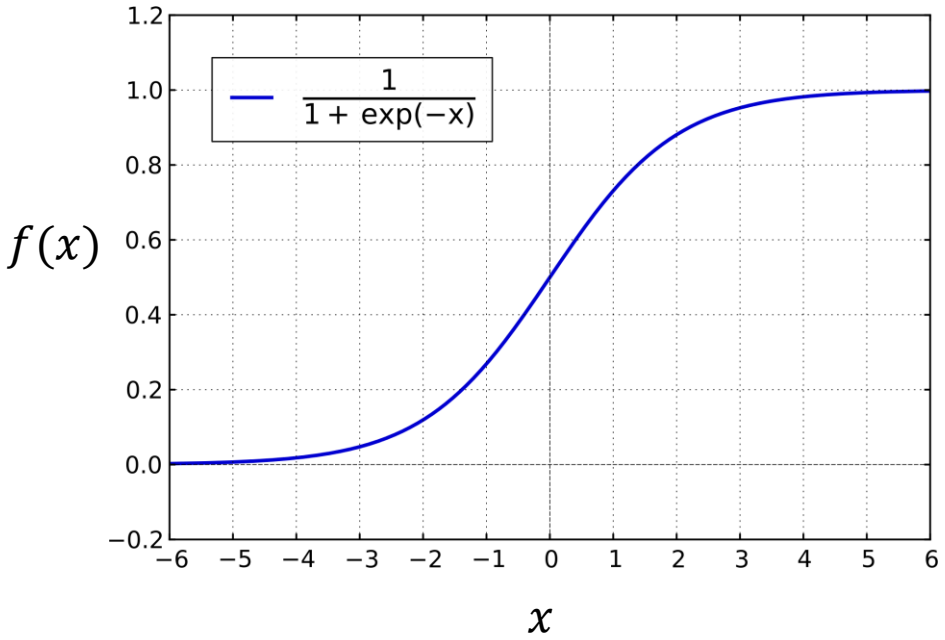
- **Directly** model the prediction of y (target) on x (input) as a **conditional probability:** $p(y|x)$
- Compared with **non-probabilistic** linear regression
  - Map input to a continuous target value
  - But the continuous target value is constrained to [0, 1]
  - Add one activation function – logistic function
- This approach is known as **logistic regression**

Logistic Regression

Logistic Regression Function:

- Assume that a particular function form: sigmoid applied to a linear function of the data:

$$y(\mathbf{x}) = f(\mathbf{w}^{\mathrm{T}}\mathbf{x}), f(z) = \frac{1}{1+\exp^{(-z)}}$$



Logistic Regression

How to model class probability via logistic function (binary classification 0 & 1)

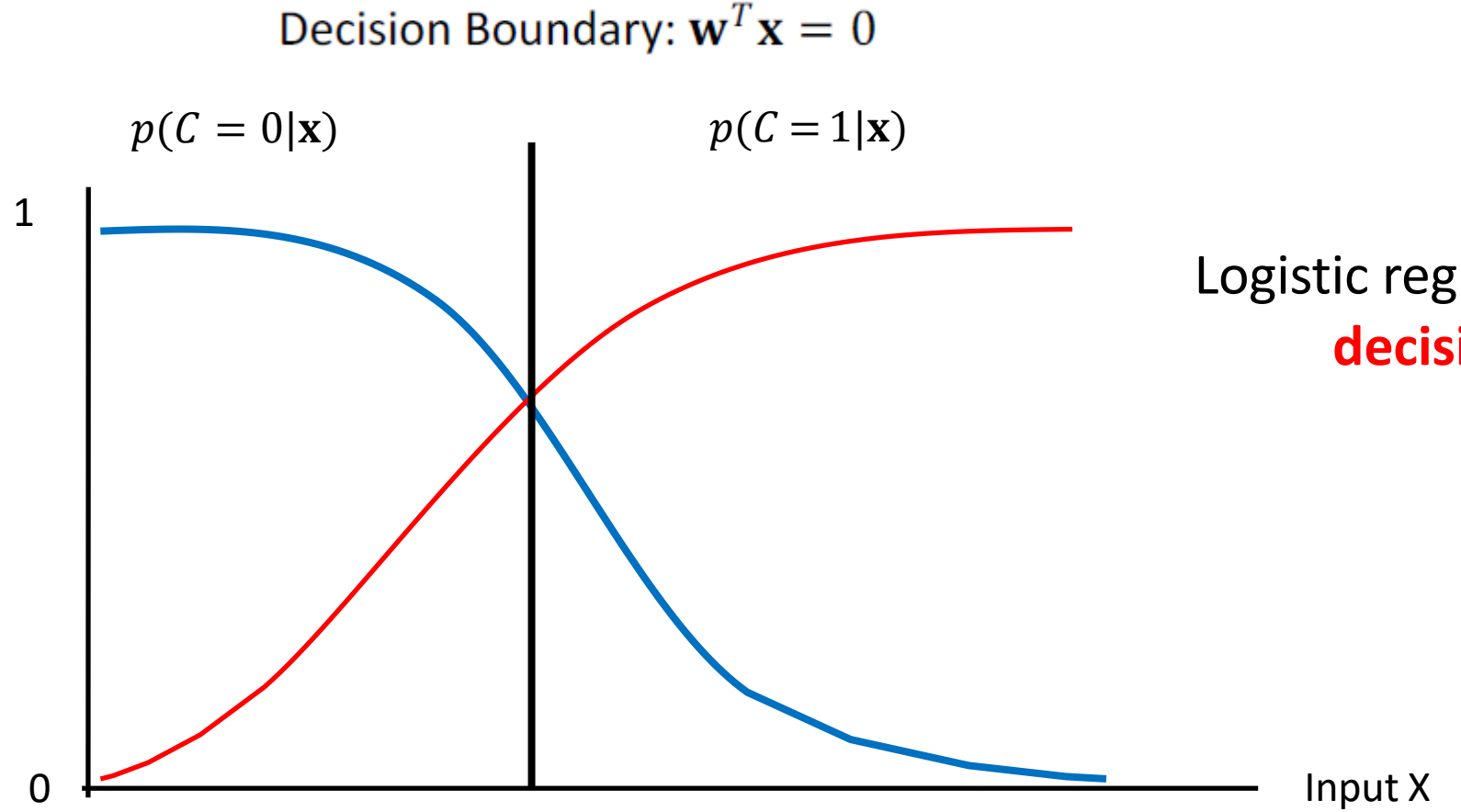
- $p(C = 1|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x})$ with $\sigma(z) = \frac{1}{1+\exp^{(-z)}}$
- If we substitute

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

$$p(C = 0|\mathbf{x}) = 1 - \frac{1}{1 + \exp^{(-\mathbf{w}^T\mathbf{x})}} = \frac{\exp^{(-\mathbf{w}^T\mathbf{x})}}{1 + \exp^{(-\mathbf{w}^T\mathbf{x})}}$$

Learning Model Parameters

How can we visualise the decision boundary for logistic regression?



Logistic regression has a **linear decision boundary**

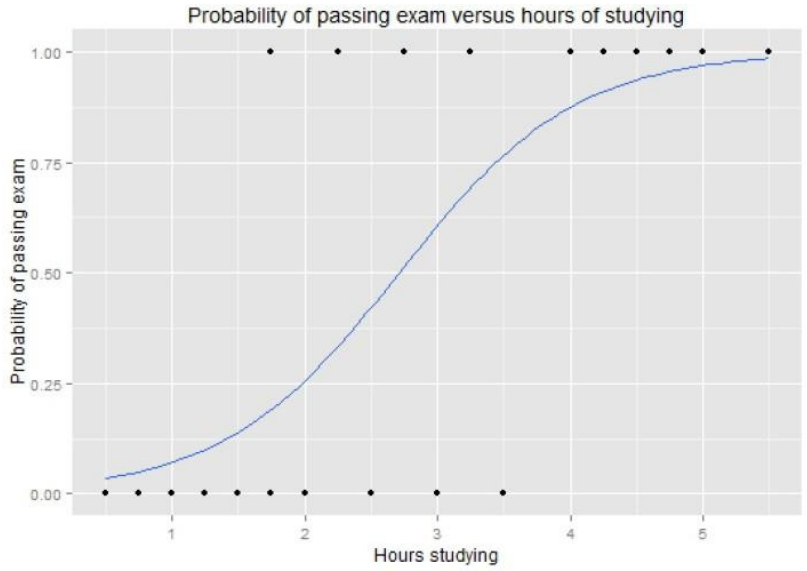
Example

Probability of passing an exam versus hours of study

- Given hours a student studies, estimate the probability that the student will pass the exam?
- Training data: A group of 20 students spend between 0 and 6 hours studying for an exam

Hours	0.50	0.75	1.00	1.25	1.50	1.75	1.75	2.00	2.25	2.50	2.75	3.00	3.25	3.50	4.00	4.25	4.50	4.75	5.00	5.50
Pass	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	1	1	1	1	1

- Learned model: $y(hours) = \sigma(\mathbf{w}^{\mathrm{i}}\mathbf{x}) = \sigma(-4.078 + 1.5x)$ Our focus: Learn $\mathbf{w}$ for our model



Hours of study	Probability of passing exam
1	0.07
2	0.26
3	0.61
4	0.87
5	0.97

Logistic Regression

How can we learn the model parameters **w**

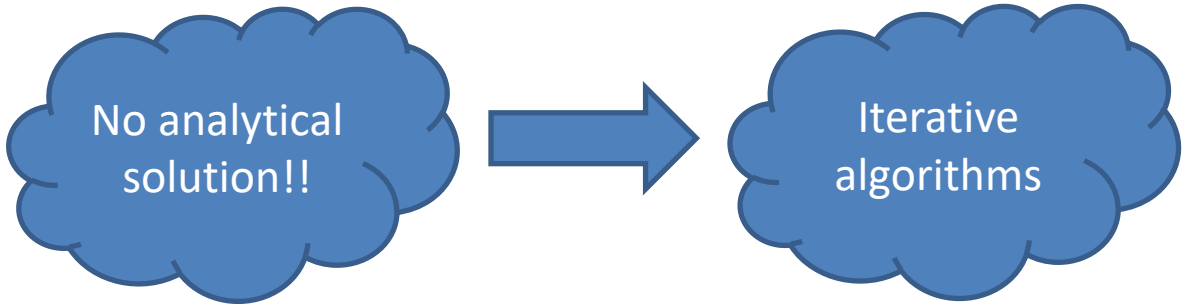
- Training data: $\{(\boldsymbol{x}_1, t_1), \ldots, (\boldsymbol{x}_n, t_n)\}$, t_n = 1 (class 1) and 0 (class 2) (just for explanation)

The maximum likelihood function:

$$\mathcal{L}(\boldsymbol{w}) := \log \prod_{n=1}^{N} y_n^{t_n} (1 - y_n)^{1-t_n}$$

$$y(\boldsymbol{x}) := p(\mathcal{C}_1 | \boldsymbol{x}) = \sigma(\boldsymbol{w} \cdot \boldsymbol{x})$$

$$\frac{\partial}{\partial \boldsymbol{w}} \mathcal{L}(\boldsymbol{w}) = 0 \Rightarrow \sum_{n=1}^{N} \left(t_n - \sigma(\boldsymbol{w} \cdot \boldsymbol{x})\right) \boldsymbol{x} = \boldsymbol{0}$$



Gradient Descent

Putting all together (plugging the update into gradient descent):

$$\mathcal{L}(\boldsymbol{w}) := \log \prod_{n=1}^{N} y_n^{t_n}(1-y_n)^{1-t_n}$$

Stochastic gradient descent for logistic regression:

- Initialise the parameters to zero
- Do the following until $|\mathcal{L}(\boldsymbol{w}^{(\tau+1)}) - \mathcal{L}(\boldsymbol{w}^{(\tau)})| < \epsilon$
  - For each training data point $(\mathbf{x}_n, t_n)$, update $\mathbf{w}$:

$$\boldsymbol{w}^{(\tau+1)} := \boldsymbol{w}^{(\tau)} - \eta^{(\tau)}(y_n - t_n)\boldsymbol{x}_n$$

where $(y_n - t_n)\boldsymbol{x}_n$ is the gradient of the error function

Logistic Regression Wrap-up (compared with probabilistic generative models)

- Quick to train
- Fast at classification
- Good accuracy for many simple data sets
- Resistant to overfitting
- Model parameters can be thought as indicators of feature importance

Tutorial (Week 6)

- Bayesian classifier
- Logistic regression


THANK YOU
for your attention!