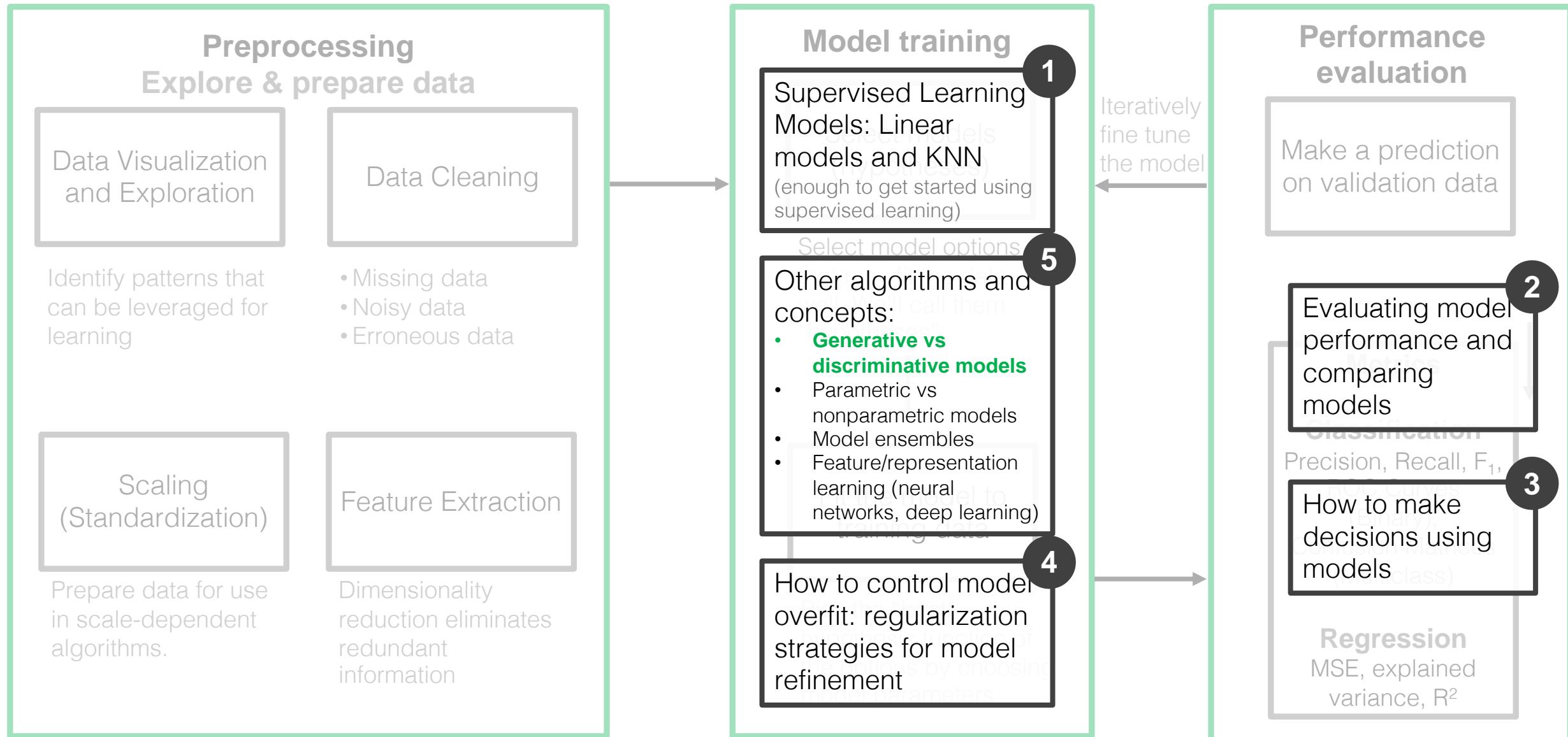


Generative Models for Classification

Supervised learning in practice



Classifiers

Covered so far

K-Nearest Neighbors

Perceptron

Logistic Regression

Linear Discriminant Analysis

Quadratic Discriminant Analysis

Naïve Bayes

Along the way...

Revisiting Bayes' Rule

Projections from higher dimensions

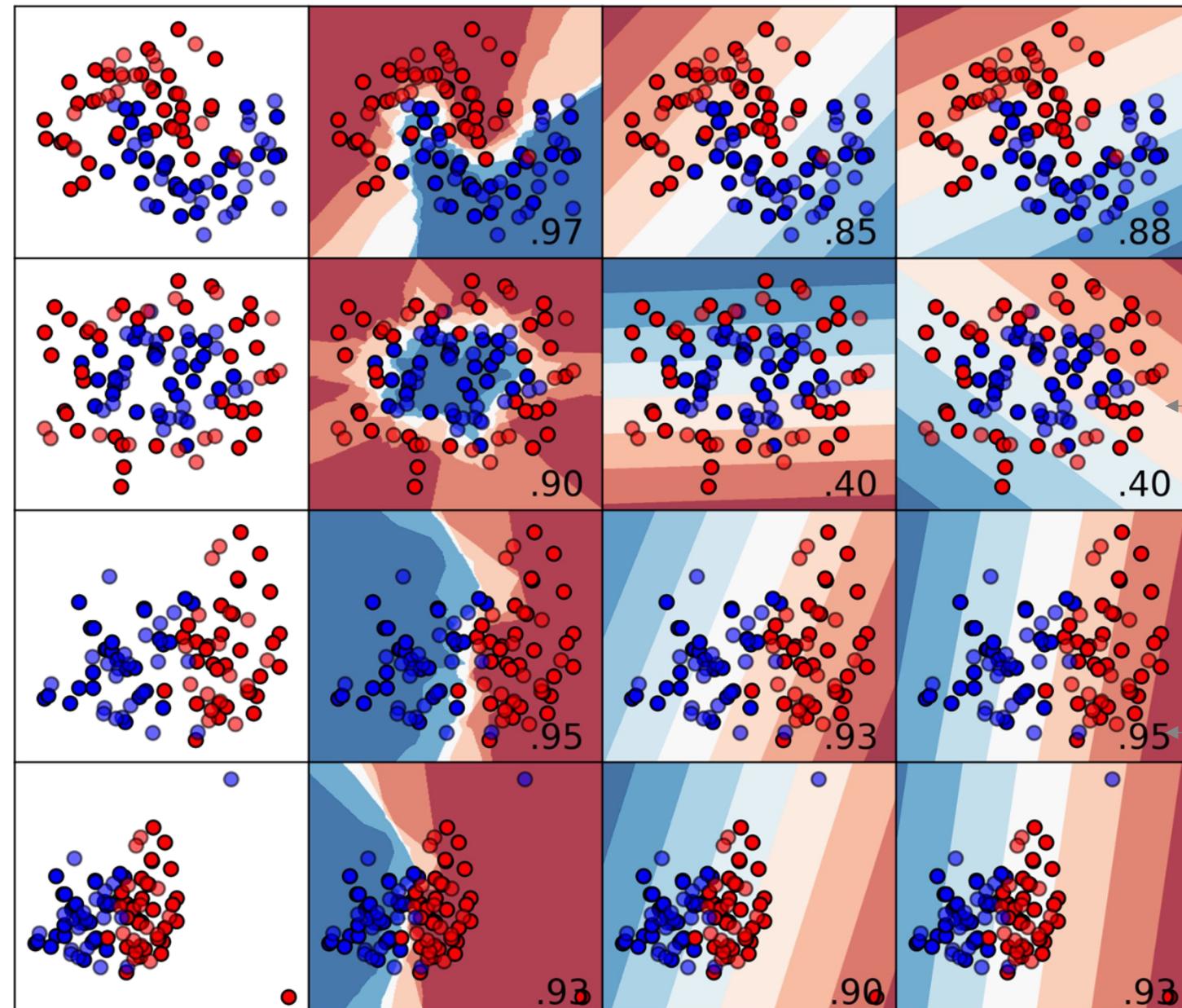
Multivariate normal distributions

Input data

KNN (k=5)

Perceptron

Logistic Reg.



Comparison of classifiers
we have seen so far

The color gradient shows
the confidence scores

Test data accuracy

Examples of deep generative models...

Face Synthesis

These images are all synthetic

Image Synthesis ([link](#))

Karras et al. 2018, NVIDIA: Progressive growing of GANS for improved quality, stability, and variation



Synthetic Generation

Karras, T., Laine, S. and Aila, T., 2019. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4401-4410).



Synthetic Images

From source B:

Pose, general hair style, face shape, eyeglasses

Hair style, eyes open/closed

Color scheme and microstructure

Style Mixing:



Bayes' rule in the context of classification



Class 1: Light Image

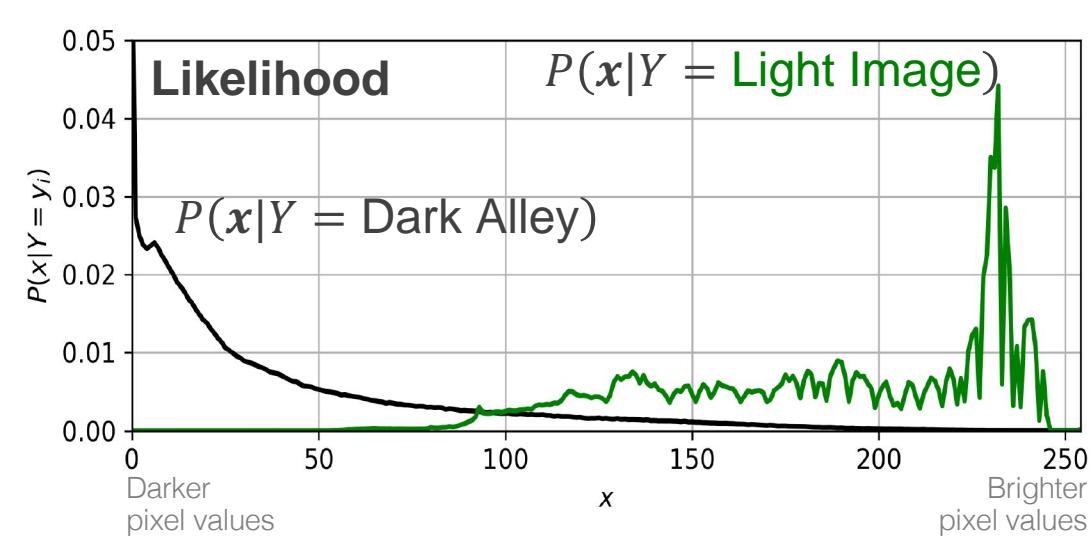
Randomly draw a pixel from either of the images: $x_i = 149$

Darker pixel values are lower numbers (closer to 0), brighter pixels are higher numbers (closer to 255)



Class 0: Dark Image

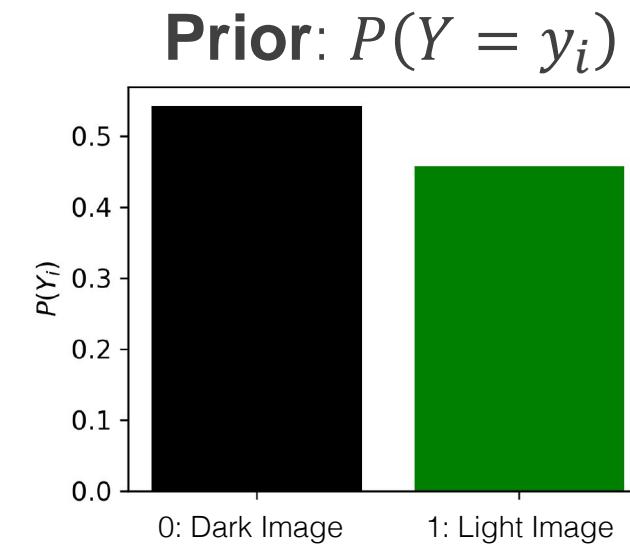
How do we determine which image the sample was most likely to have come from?



Class 1: Light Image y_1



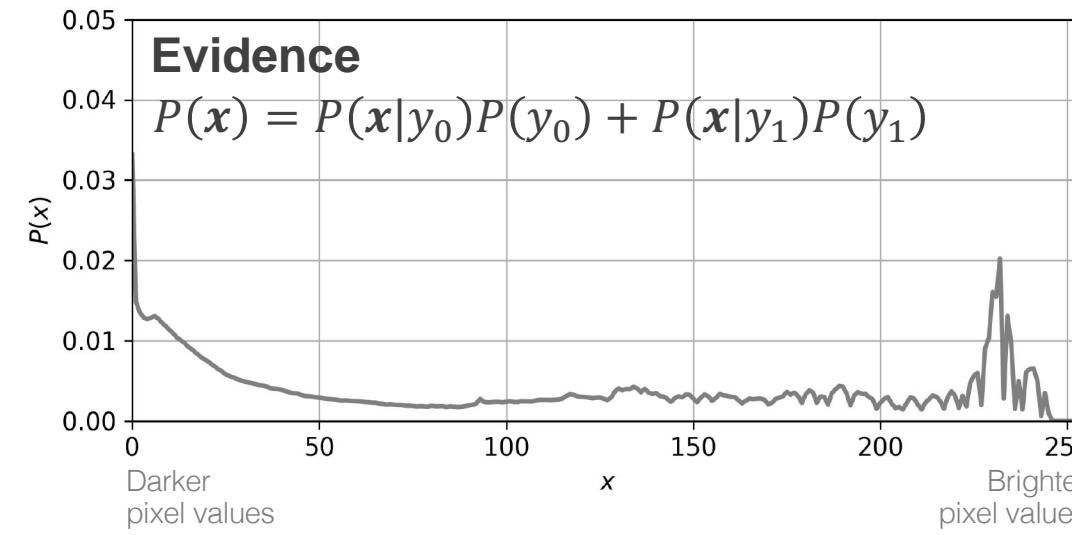
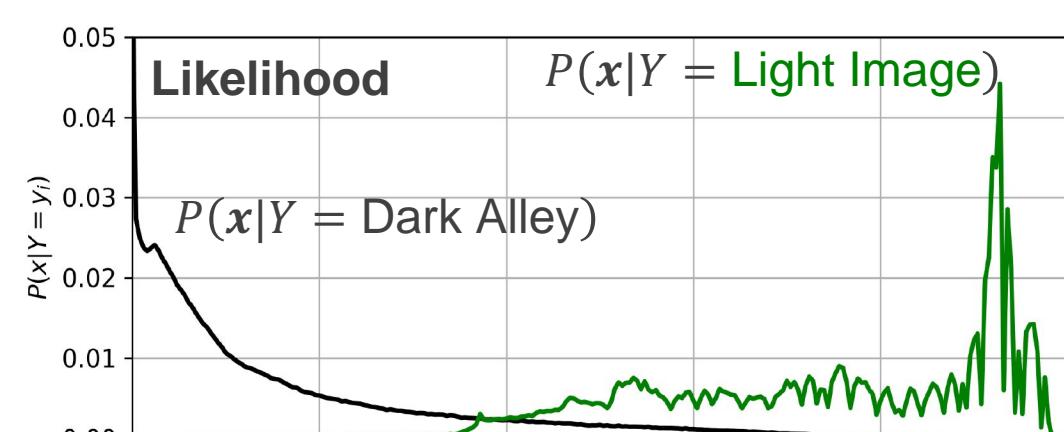
Class 0: Dark Image y_0



Bayes' Rule

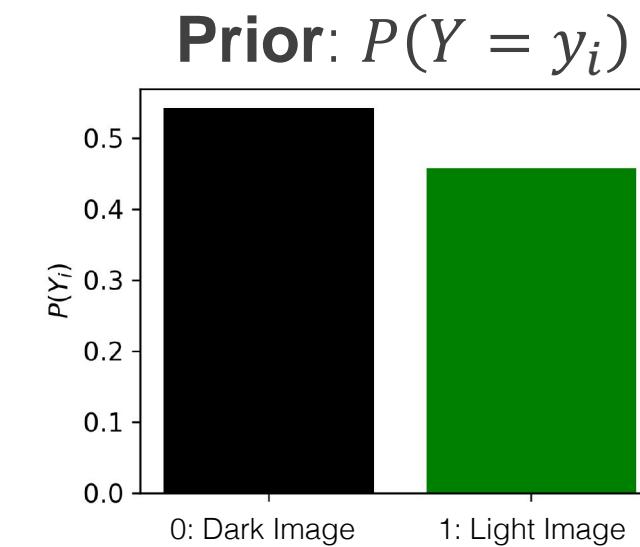
$$\text{Posterior } P(Y = y_i|x) = \frac{\text{Likelihood} \quad \text{Prior}}{\text{Evidence}}$$

$$P(Y = y_i|x) = \frac{P(x|Y = y_i)P(Y = y_i)}{P(x)}$$



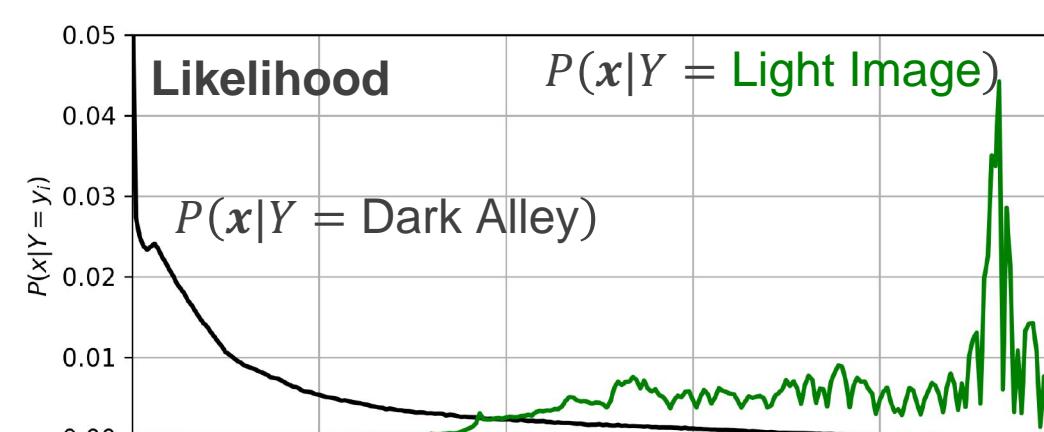
Class 1: Light Image y_1

Class 0: Dark Image y_0

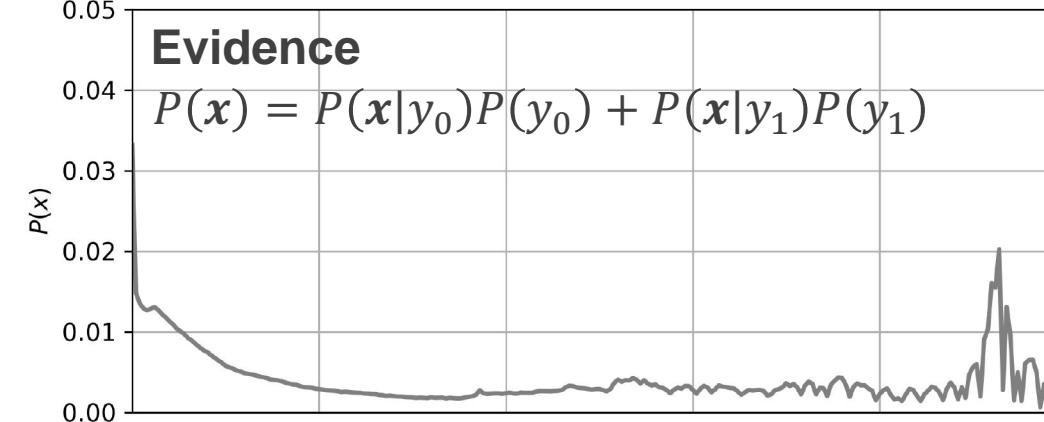


Bayes' Rule

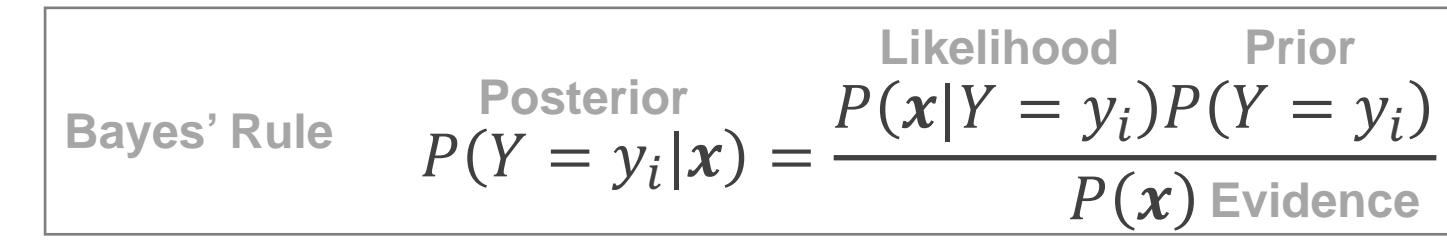
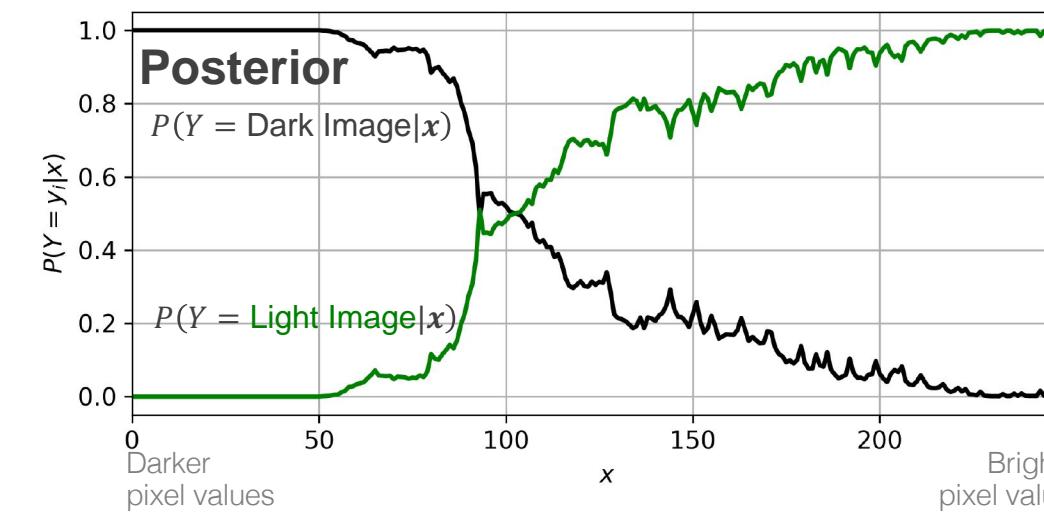
$$\text{Posterior } P(Y = y_i|x) = \frac{P(x|Y = y_i)P(Y = y_i)}{P(x) \text{ Evidence}}$$

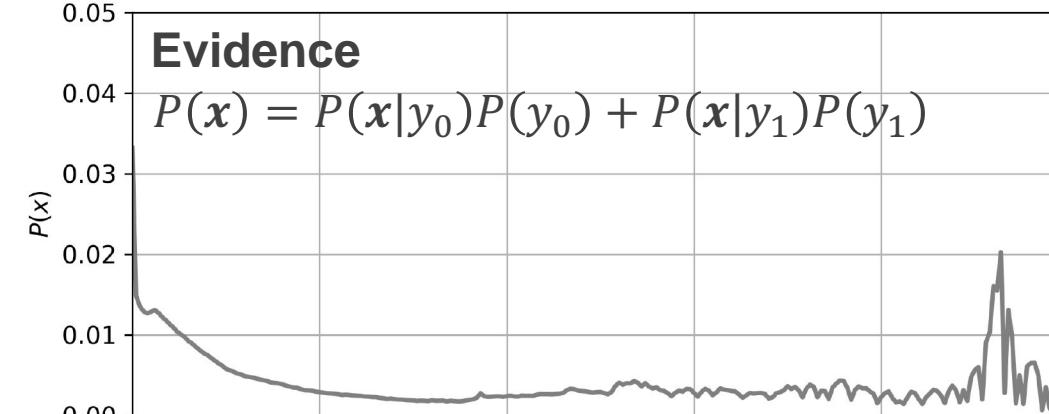
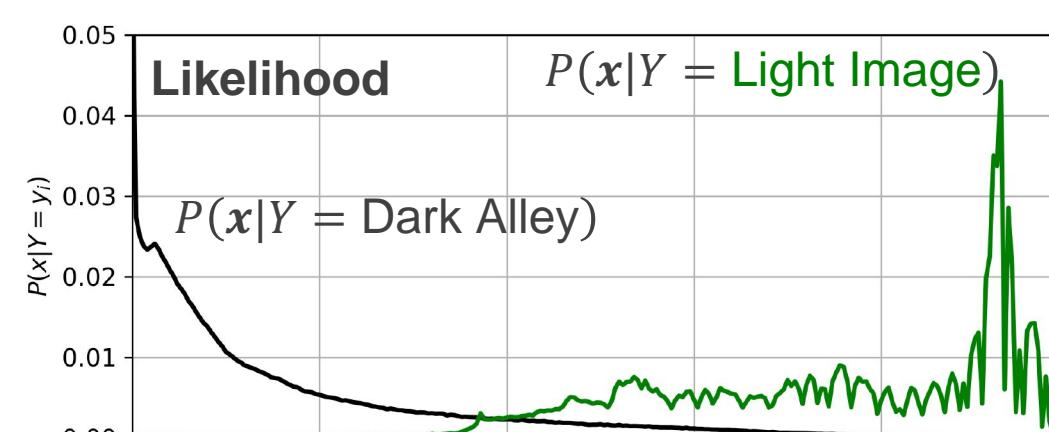


Class 1: Light Image y_1



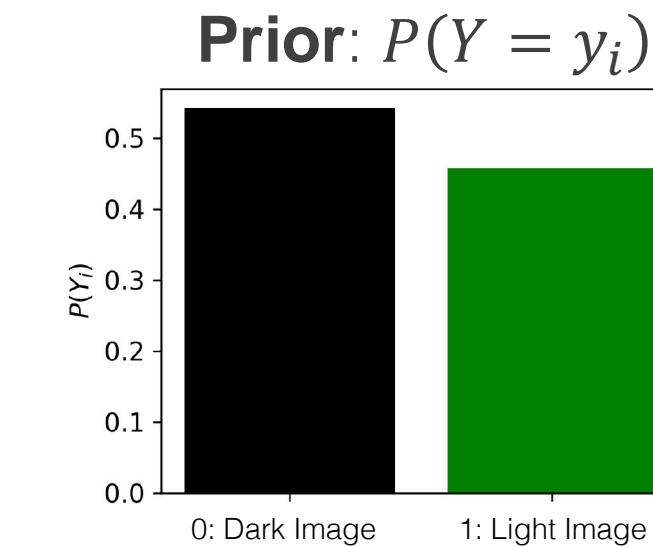
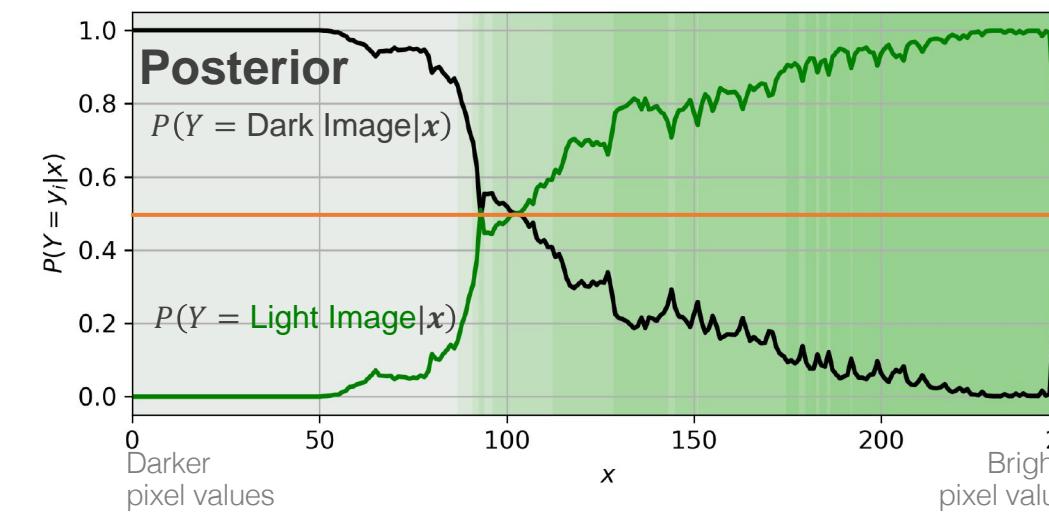
Class 0: Dark Image y_0





Class 1: Light Image y_1

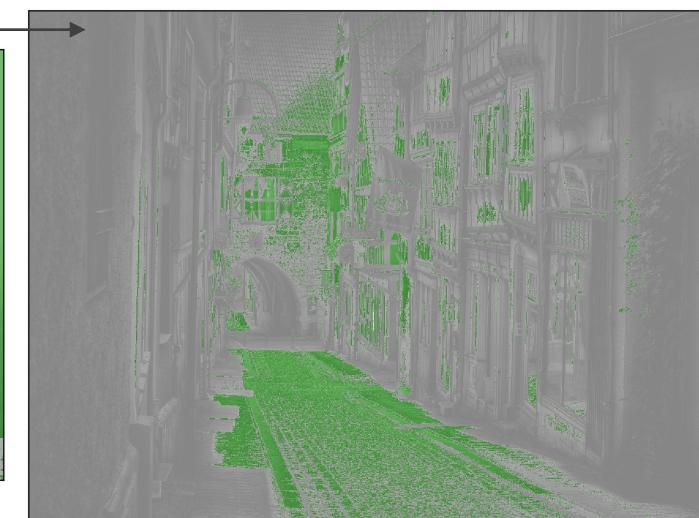
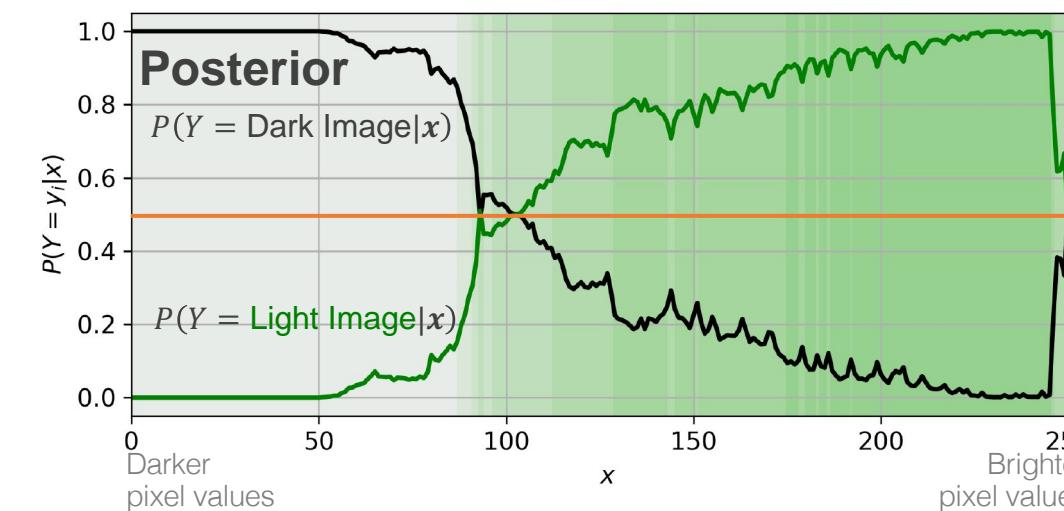
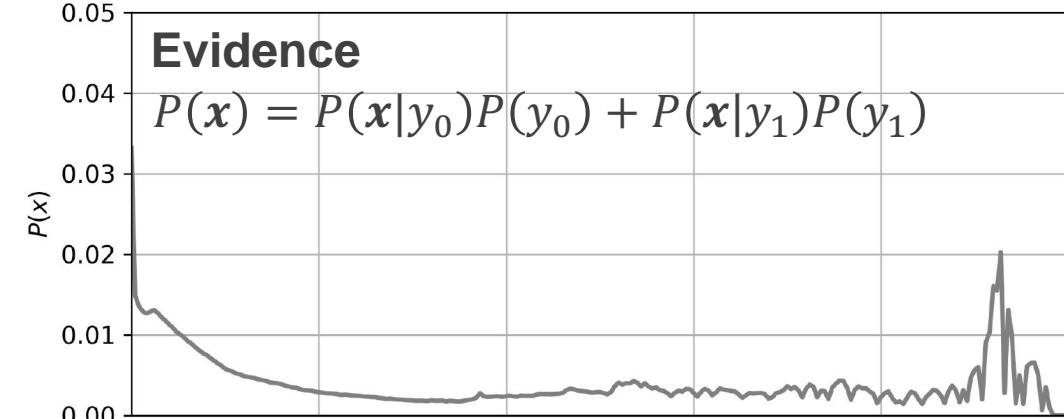
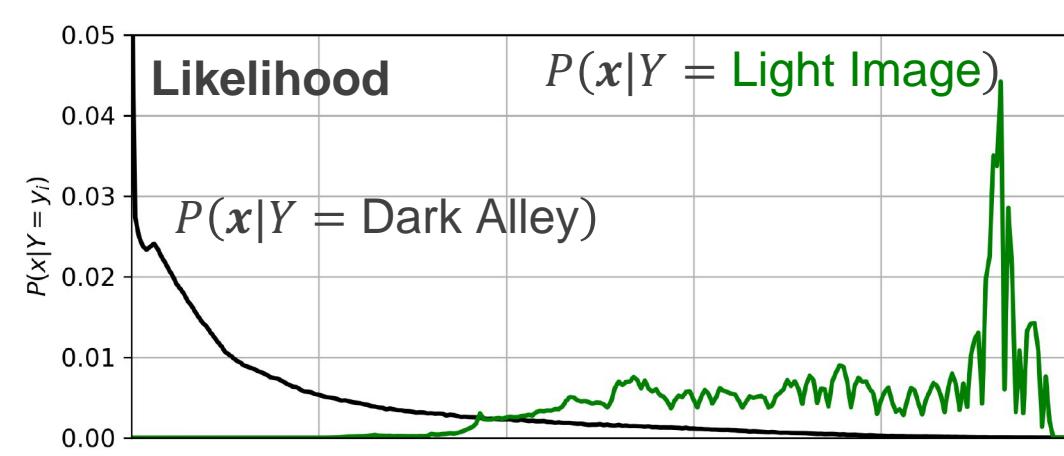
Class 0: Dark Image y_0



Decision rule:

If $P(Y = \text{Light Image}|x) > P(Y = \text{Dark Image}|x)$ then
else Dark Image

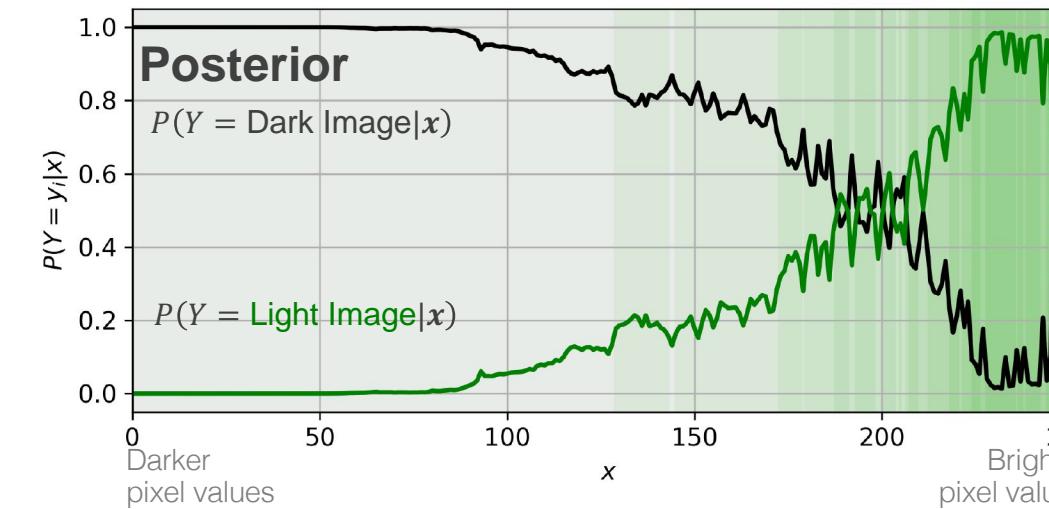
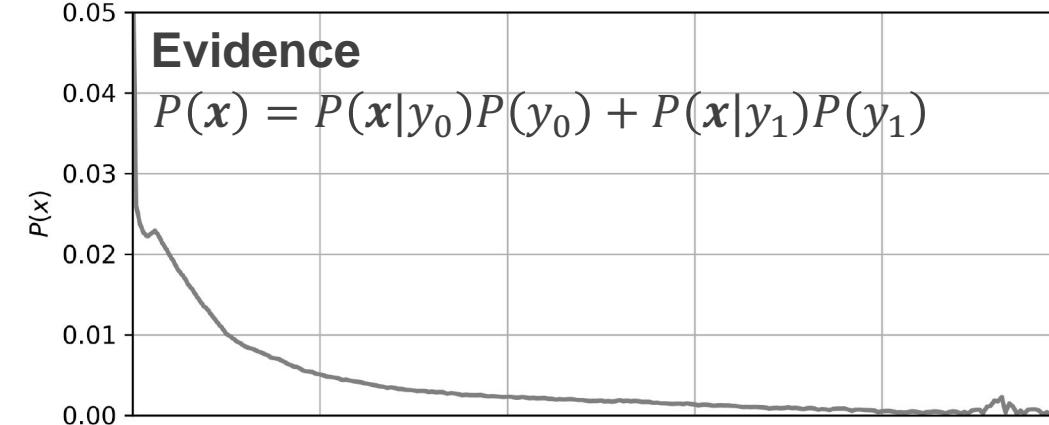
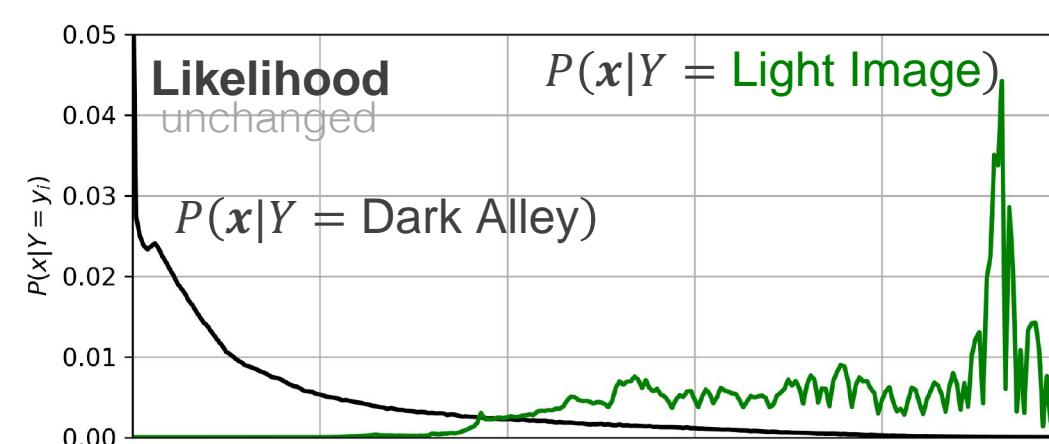
Light Image



Classifying each of the individual pixels as being either from **Light Image** or **Dark Image** results in classification above

Decision rule:

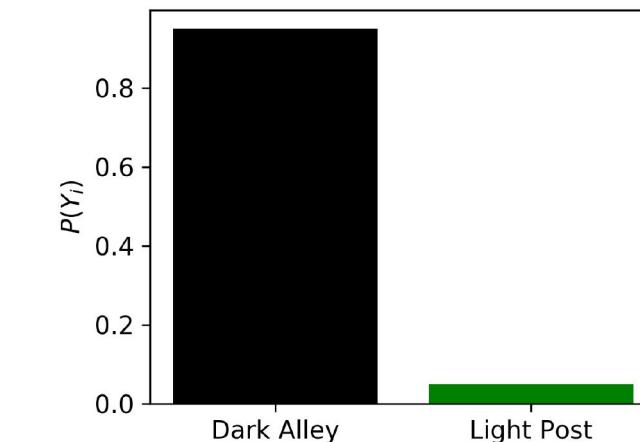
If $P(Y = \text{Light Image}|x) > P(Y = \text{Dark Image}|x)$ then **Light Image**
else **Dark Image**



Class 1: Light Image y_1

Class 0: Dark Image y_0

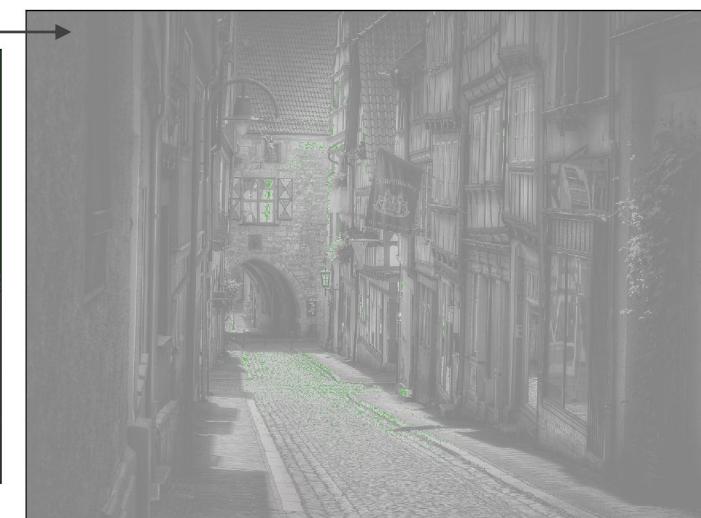
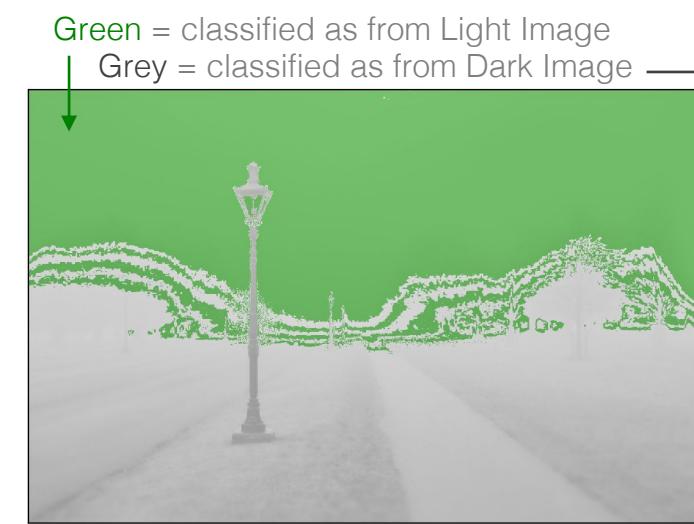
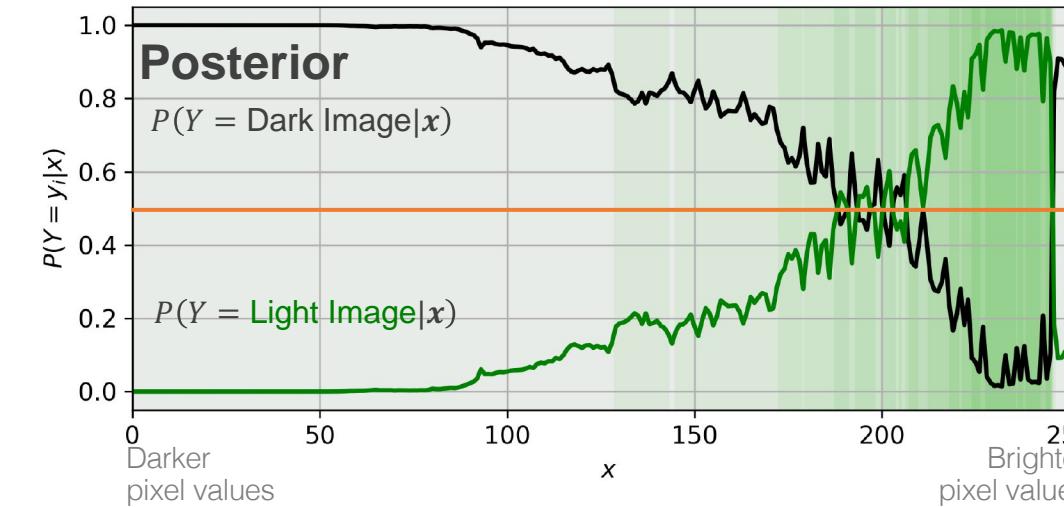
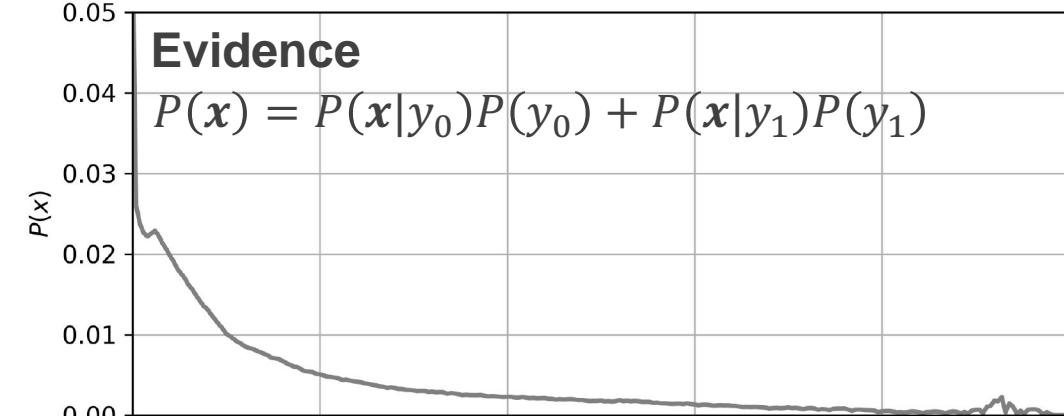
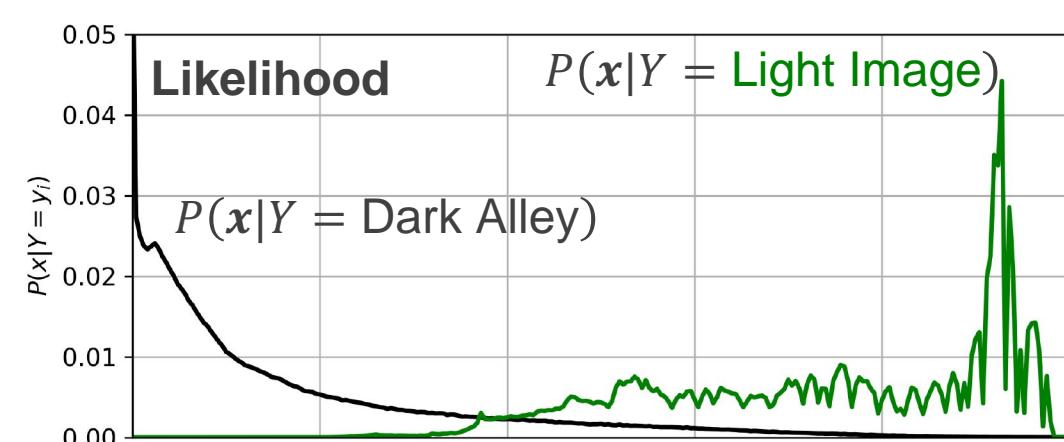
Prior: $P(Y = y_i)$



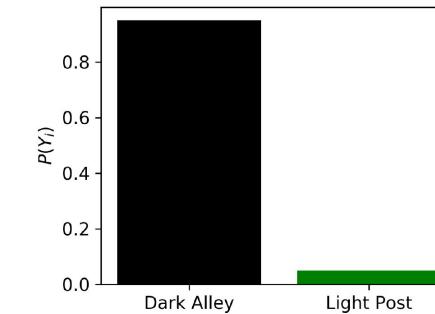
Let's assume the sampling of pixels occurred more from the **Dark Image**

Bayes' Rule

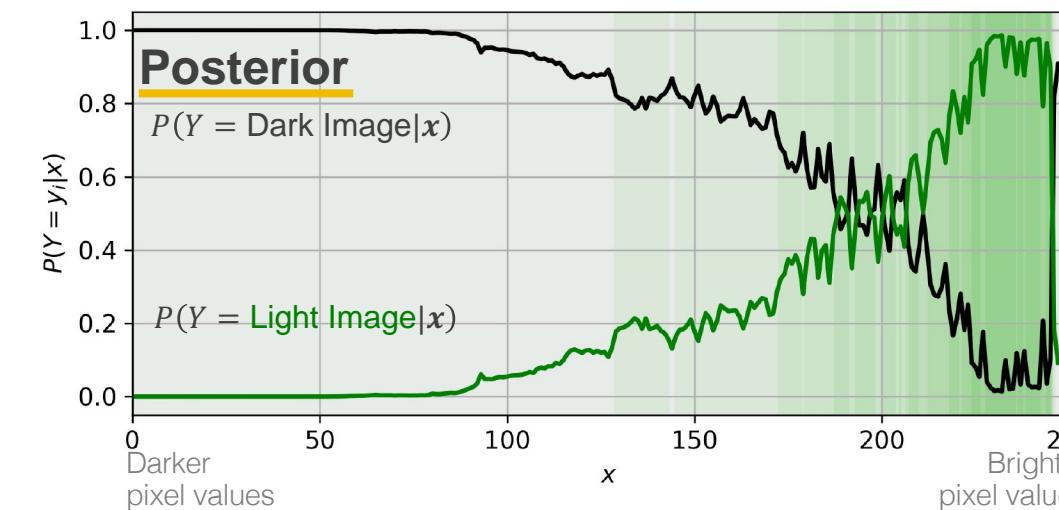
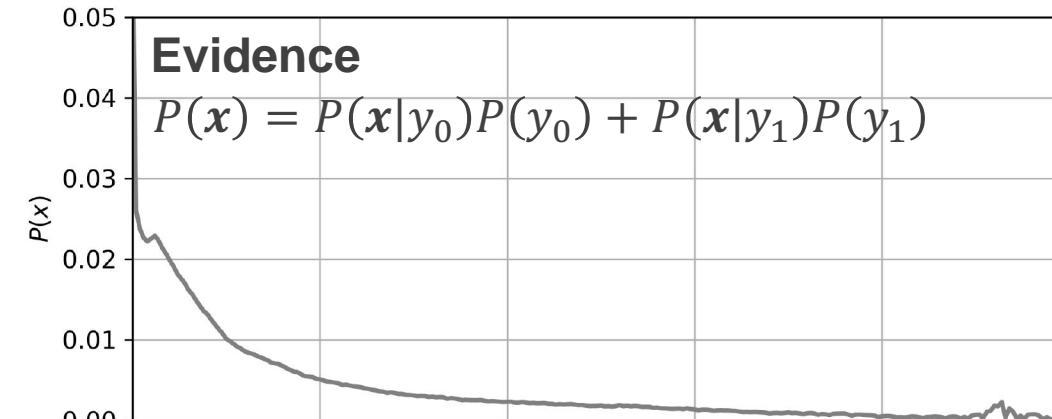
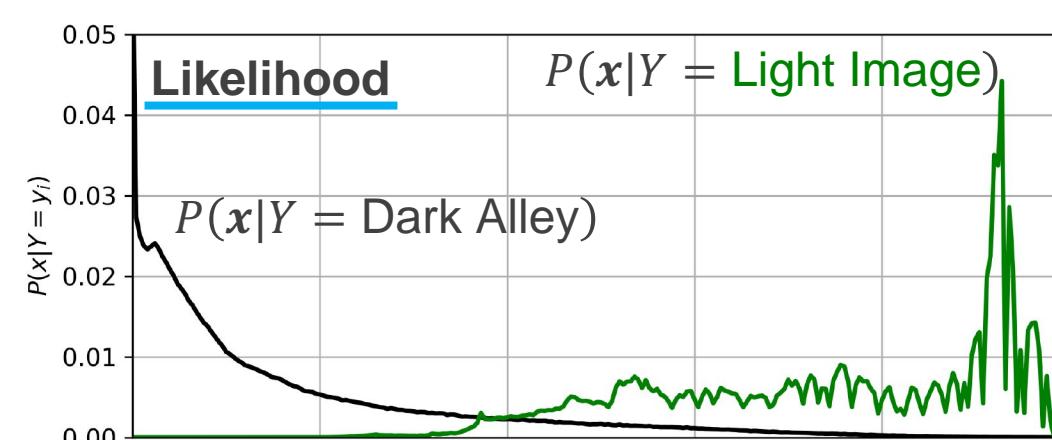
$$\text{Posterior} \quad P(Y = y_i|x) = \frac{\text{Likelihood} \quad P(x|Y = y_i)P(Y = y_i)}{\text{Evidence} \quad P(x)}$$



Prior: $P(Y = y_i)$



Assuming we the sampling of pixels occurred more from the
Dark Image



Generative models model the likelihood

- These can also be used to generate synthetic data
- Often good performance when sample size is small

Examples: linear discriminant analysis, naïve Bayes, hidden Markov models, Gaussian mixture models, Generative Adversarial Networks

$$P(Y = y_i|x) = \frac{\text{Likelihood} \quad \text{Prior}}{\text{Evidence}}$$

Likelihood

Posterior

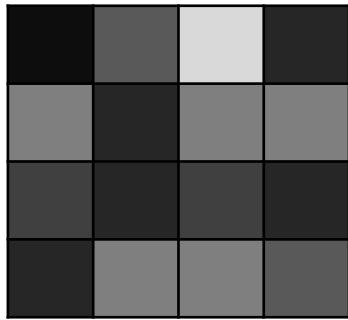
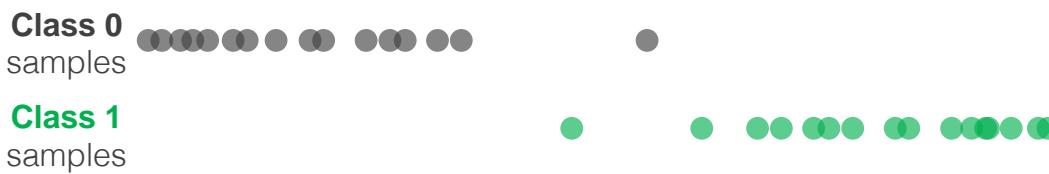
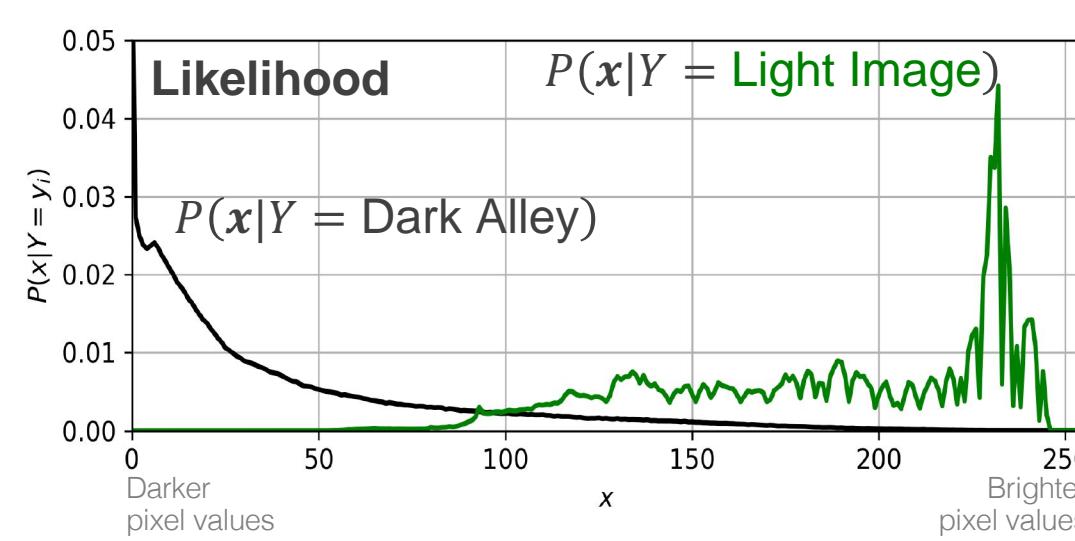
Prior

Evidence

Discriminative models model the posterior

- Or they just directly estimate labels without a probabilistic interpretation, $f(\mathbf{x}) \rightarrow \mathbf{y}$
- Often better performance for large sample sizes

Examples: logistic regression, support vector machines, neural networks, k nearest neighbors



Synthetic “Image”
sampled from
Class 0



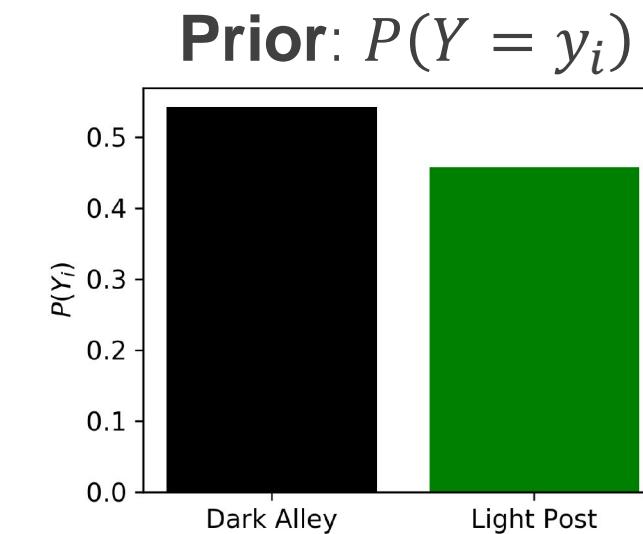
Synthetic “Image”
sampled from
Class 1

How to “Generate” Data?



Class 1: Light Image y_1

Class 0: Dark Image y_0



Bayes’ Rule

Posterior

$$P(Y = y_i|x) = \frac{\text{Likelihood} \quad \text{Prior}}{\text{Evidence}}$$

$$P(x|Y = y_i)P(Y = y_i)$$

$$P(x)$$

Generative modeling for classification

Assume we have c different classes

For a new sample, classify it as the class with the **largest posterior** $P(y = i|x)$
 $i = 1, \dots, c$

Generative modeling for classification

If we have c different classes, we define a discriminant function, $d_i(\mathbf{x})$ for $i = 1, \dots, c$

If $P(y = i|\mathbf{x}) > P(y = j|\mathbf{x})$ for $i \neq j$, then we classify feature \mathbf{x} to class i

$$P(y = i|\mathbf{x}) = \frac{P(\mathbf{x}|y = i)P(y = i)}{P(\mathbf{x})}$$

$$= \frac{P(\mathbf{x}|y = i)P(y = i)}{\sum_{i=1}^c P(\mathbf{x}|y = i)P(y = i)}$$

Bayes' Rule: $P(Y|\mathbf{X}) = \frac{\text{Likelihood}}{\text{Posterior}} \cdot \frac{\text{Prior}}{\text{Evidence}}$

$$P(Y|\mathbf{X}) = \frac{P(\mathbf{X}|Y)P(Y)}{P(\mathbf{X})}$$

Denominator is the same for all classes i , so it won't help us tell which class's posterior is higher relative to other classes, so we ignore it going forward

We can define the discriminant function as:

$$d_i(x) = P(\mathbf{x}|y = i)P(y = i)$$

If we know the **true likelihood and prior** for our data, this process yields our **Bayes' classifier** (minimum misclassification error classifier)

Generative modeling for classification

$$d_i(x) = P(\mathbf{x}|y = i)P(y = i)$$

We **rarely** know our **true likelihood** for our data so we need to assume a form for the distributions and approximate

- 1 Assume a form for $P(\mathbf{x}|y = i)$

Gaussian → Linear and Quadratic
Discriminant Analysis
Gaussian mixture models
Nonparametric density estimates

If we assume **independent features** → **Naïve Bayes**

(remember $d_i(x)$ proportional to the posterior)

- 2 Assign the class, i , for which $d_i(x)$ is largest

Applies to both binary and multiclass problems

Covariance matrix

$$\Sigma = \frac{1}{N} \sum_{i=1}^N (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \rightarrow [D \times D]$$

$[D \times 1][1 \times D]$

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} & \cdots & \Sigma_{1D} \\ \Sigma_{21} & \Sigma_{22} & \cdots & \Sigma_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{D1} & \Sigma_{D2} & \cdots & \Sigma_{DD} \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \Sigma_{12} & \cdots & \Sigma_{1D} \\ \Sigma_{21} & \sigma_2^2 & \cdots & \Sigma_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{D1} & \Sigma_{D2} & \cdots & \sigma_D^2 \end{bmatrix}$$

$$\begin{aligned} \Sigma_{jk} &= \frac{1}{N} \sum_{i=1}^N (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k) \\ &= \text{cov}(X_j, X_k) \\ &= E[(X_j - \mu_j)(X_k - \mu_k)] \end{aligned}$$

$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iD} \end{bmatrix}$$

Vector of observation i

x_{ij}

Observation index Predictor index

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1D} \\ x_{21} & x_{22} & \cdots & x_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{ND} \end{bmatrix}$$

Predictors →
Observations ↓

Mean of each predictor
 \downarrow
 If $\bar{x}_j = 0$ for all j

This will be the case IF the data are standardized

$$\Sigma_{jk} = \frac{1}{N} \sum_{i=1}^N x_{ij} x_{ik}$$

$$= \frac{1}{N} \mathbf{x}_j^T \mathbf{x}_k$$

$$= E[X_j X_k]$$

$$\Sigma = \frac{1}{N} \mathbf{X}^T \mathbf{X}$$

Covariance and Correlation

Relationship between covariance and correlation

$$\text{corr}(X, Y) = \frac{\text{cov}(X, Y)}{\sqrt{\text{var}(X)\text{var}(Y)}}$$

When $\text{var}(X) = \text{var}(Y) = 1$, then:

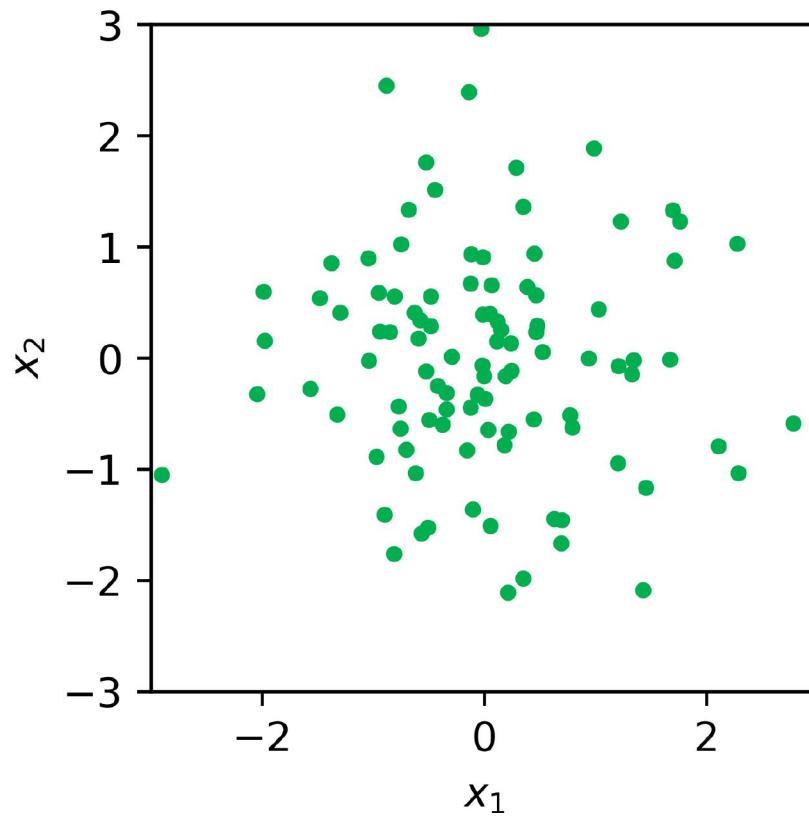
$$\text{corr}(X, Y) = \text{cov}(X, Y)$$

If each of the features have been standardized, this means this matrix is:

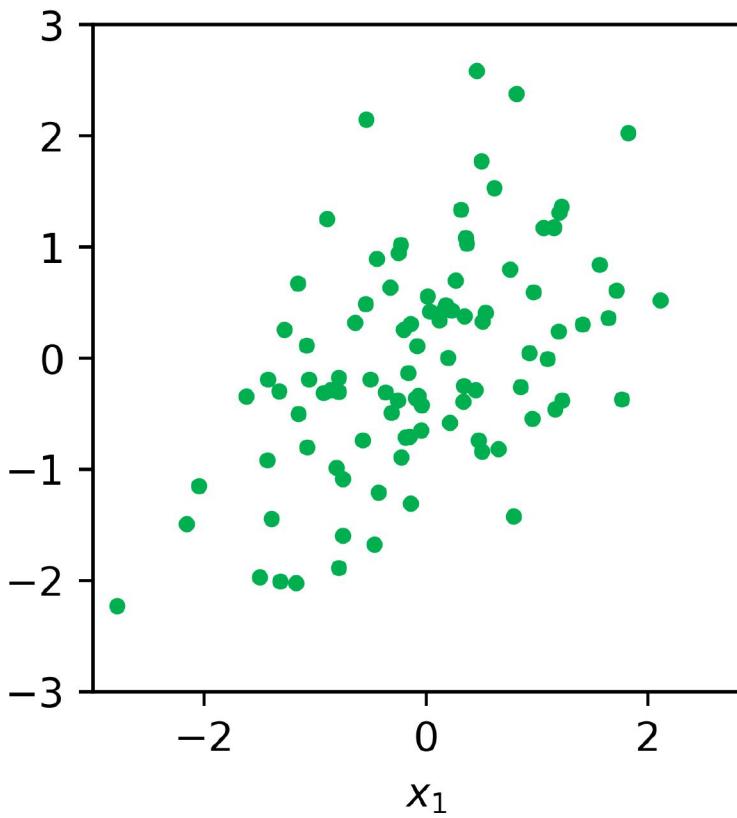
$$\Sigma = \begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1D} \\ \rho_{21} & 1 & \cdots & \rho_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{D1} & \rho_{D2} & \cdots & 1 \end{bmatrix}$$

Covariance Matrix Examples

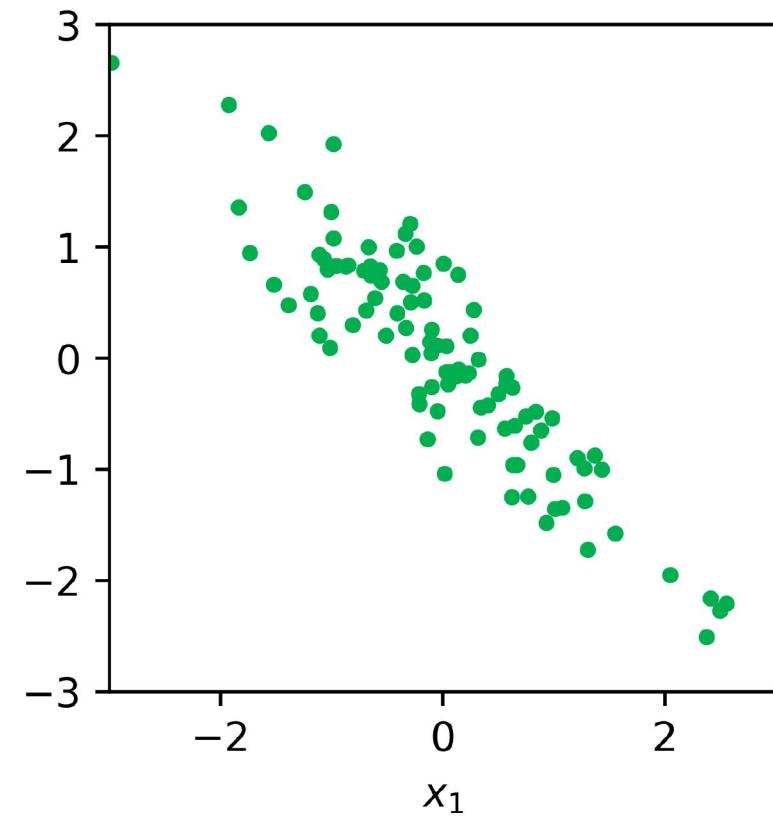
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



$$\Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$



$$\Sigma = \begin{bmatrix} 1 & -0.9 \\ -0.9 & 1 \end{bmatrix}$$



Key model differences

Quadratic Discriminant Analysis (QDA)

$$\Sigma_0 \neq \Sigma_1$$

Each class (e.g., 0 or 1) may have a unique covariance matrix

Linear Discriminant Analysis (LDA)

$$\Sigma_0 = \Sigma_1$$

Every class (e.g., 0 or 1) has an identical covariance matrix

Naïve Bayes with Gaussian Likelihoods

$$\Sigma_i = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_D^2 \end{bmatrix}$$

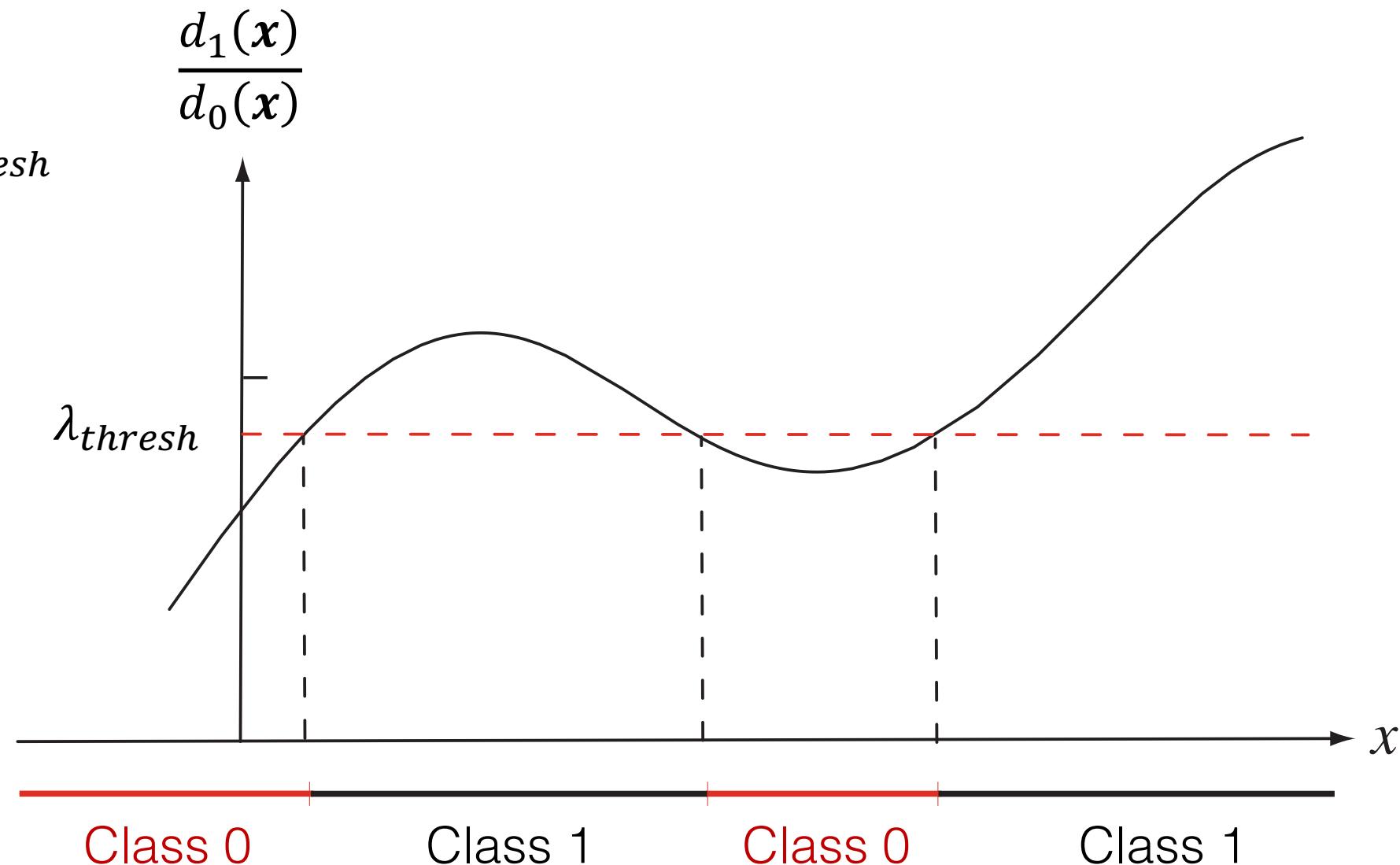
Every class (e.g., $i = 0$ or 1) has a diagonal covariance matrix

Discriminant Function: 2 classes

Decision rule:

Class 1 if: $\frac{d_1(x)}{d_0(x)} > \lambda_{thresh}$

Otherwise, class 0



Duda, Hart, and Stork, Pattern Classification, 2006

Discriminant Function: 2 classes

We build a classifier that assigns the class with the higher posterior probability:

If $\frac{d_1(\mathbf{x})}{d_0(\mathbf{x})} = \frac{P(\mathbf{x}|y=1)P(y=1)}{P(\mathbf{x}|y=0)P(y=0)} > 1$ Assign class 1, else class 0

Assumes these likelihoods are normal

We can write our decision rule as a function of likelihoods (**likelihood ratio**)

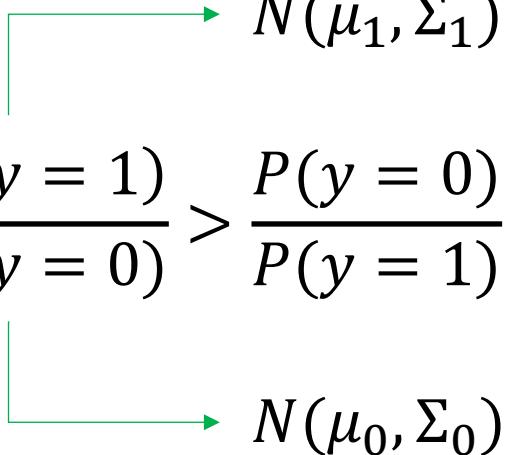
$$\frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)} > \frac{P(y=0)}{P(y=1)}$$

Estimate the class-conditional mean and covariance matrix from the data

Discriminant Function: 2 classes

We build a classifier that assigns the class with the higher posterior probability:

Likelihood ratio: $\frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)} > \frac{P(y=0)}{P(y=1)}$


$$N(\mu_1, \Sigma_1)$$
$$N(\mu_0, \Sigma_0)$$

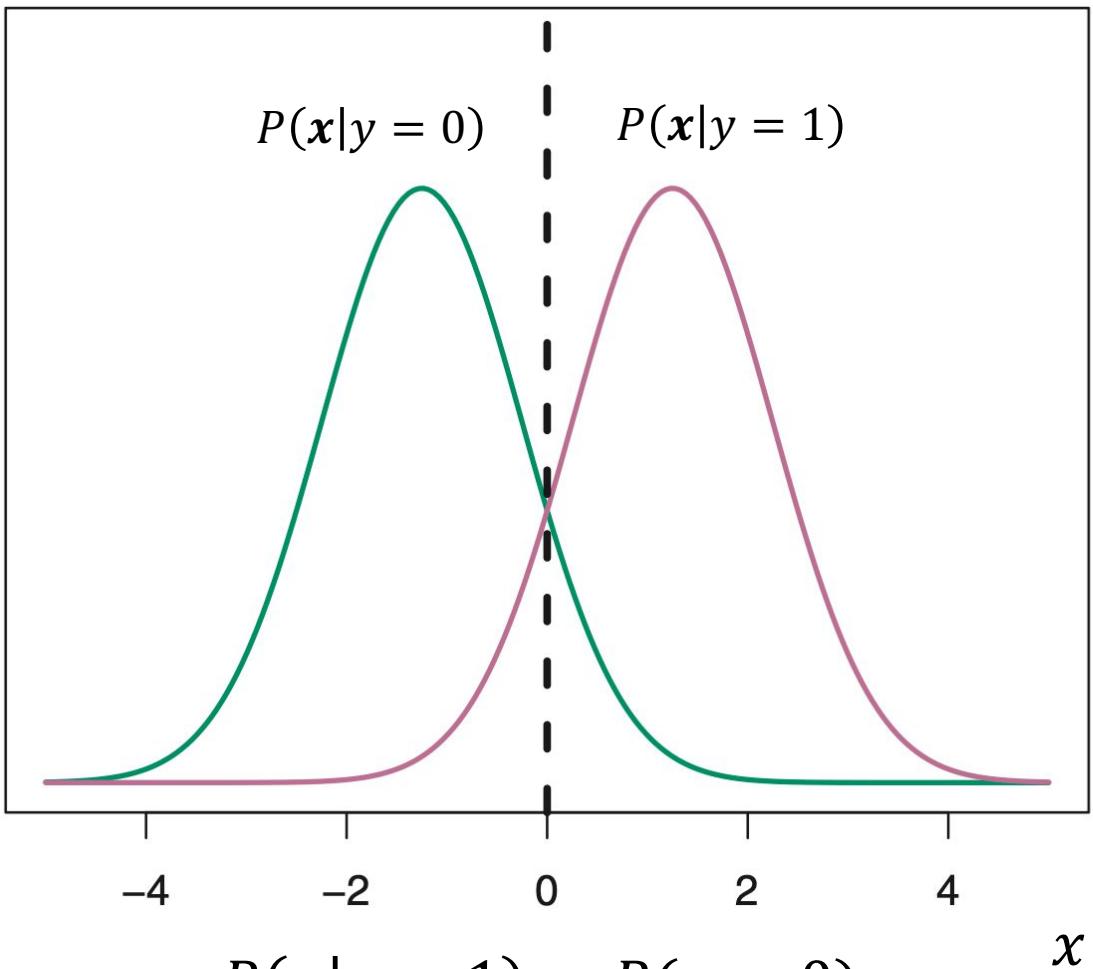
If we assume the class conditional distributions are **Gaussian**, this represents

Quadratic Discriminant Analysis

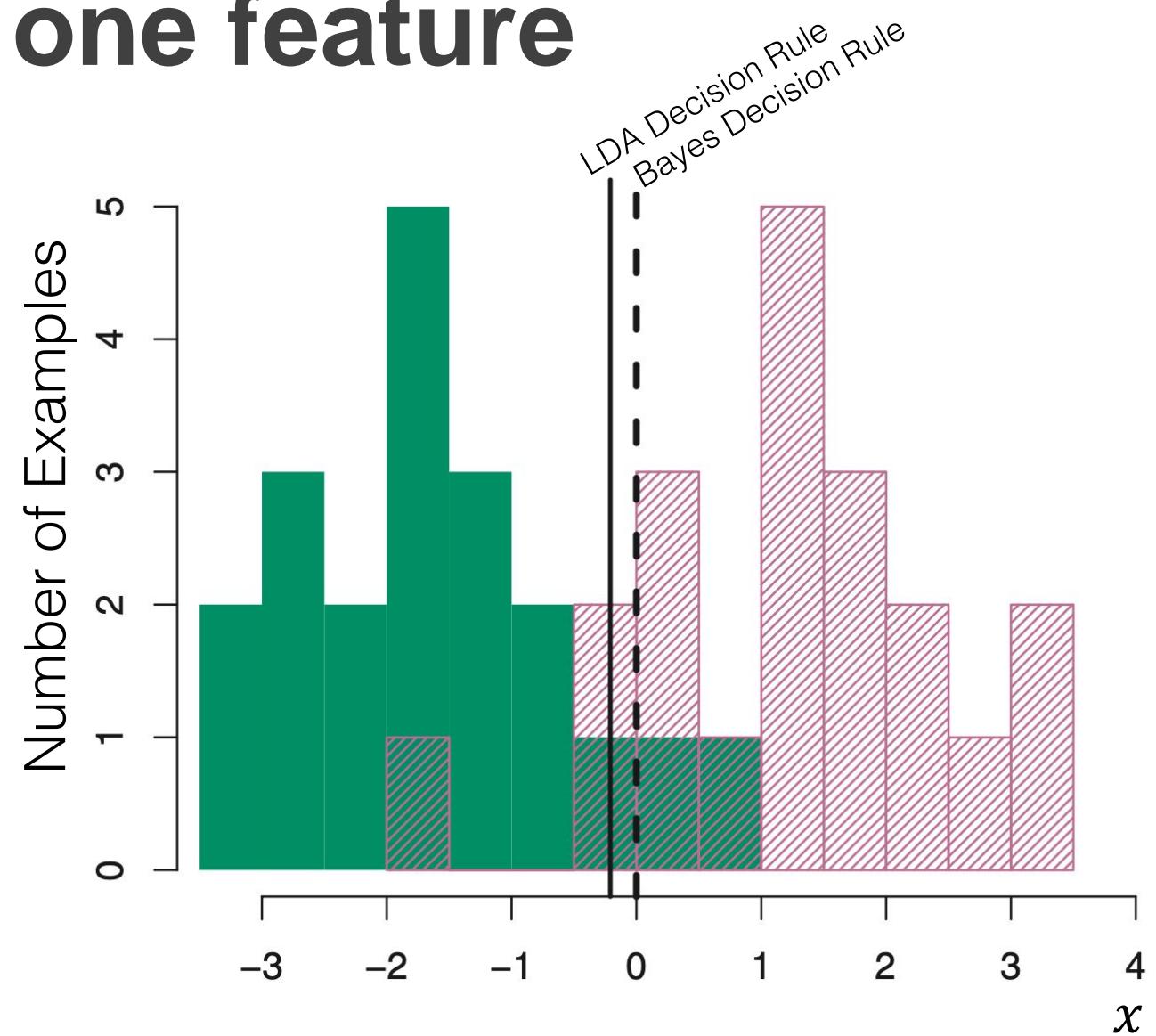
If we further assume the **covariance matrices for each class are the same**, $\Sigma_0 = \Sigma_1$, this represents

Linear Discriminant Analysis

Simple example with one feature



$$\frac{P(x|y=1)}{P(x|y=0)} > \frac{P(y=0)}{P(y=1)}$$



Figures from James et al. - Introduction to Statistical Learning

Multivariate Gaussian

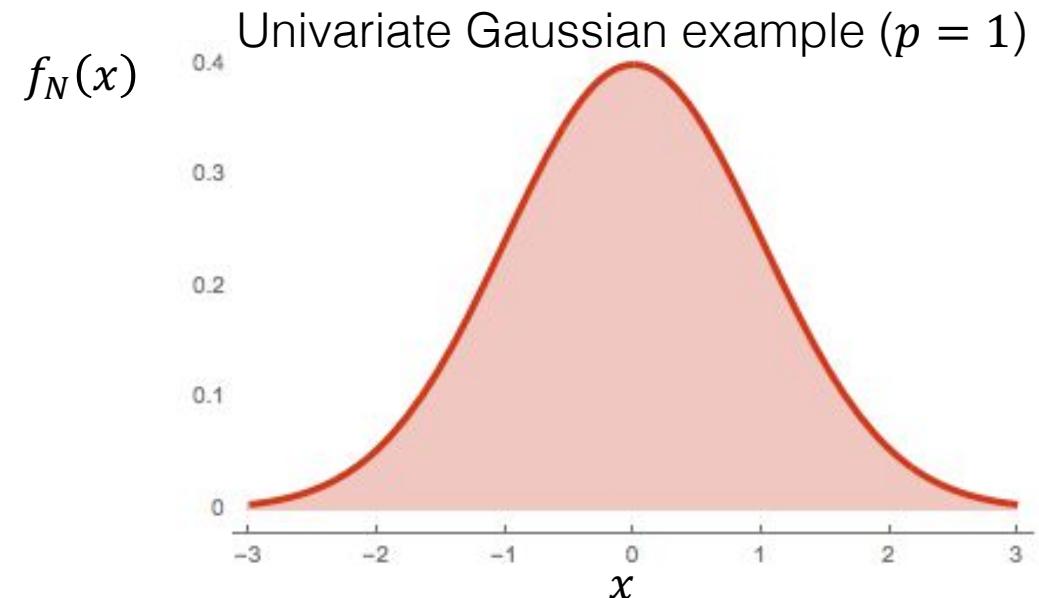
Univariate Gaussian (1 predictor)

$$f_N(x) = \frac{1}{(2\pi)^{1/2}\sigma} \exp\left[-\frac{1}{2\sigma^2}(x - \mu)^2\right]$$

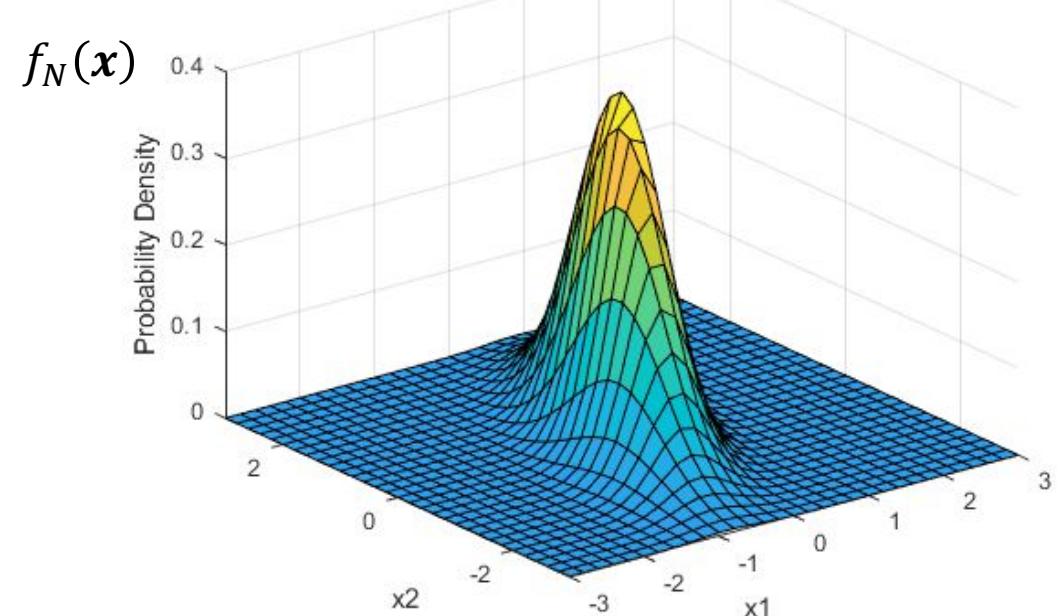
Multivariate Gaussian (p predictors)

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

Estimate our means and variances and we can use this as our likelihood



Multivariate Gaussian example ($p = 2$)



Images from Brilliant.org (top) and the Mathworks (bottom)

Linear Discriminant Analysis ($\Sigma_0 = \Sigma_1$)

We build a classifier that assigns the class with the higher posterior probability:

Likelihood ratio:

$$\begin{aligned} \frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)} &= \frac{\frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_1)^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_1)\right]}{\frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_0)^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_0)\right]} \\ &= \frac{\exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_1)^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_1)\right]}{\exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_0)^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_0)\right]} \end{aligned}$$

$$\ln \left| \frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)} \right| = -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_1)^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_1) + \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_0)^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_0)$$

Derivation 1/4

Linear Discriminant Analysis ($\Sigma_0 = \Sigma_1$)

$$\ln \left| \frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)} \right| = -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_1) + \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_0)$$

Expanding this expression yields:

These combine since $\mathbf{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i = \boldsymbol{\mu}_i^T \boldsymbol{\Sigma}^{-1} \mathbf{x}$ for symmetric matrices

$$\begin{aligned} &= -\frac{1}{2} [\cancel{\mathbf{x}^T \boldsymbol{\Sigma}^{-1} \mathbf{x}} - \mathbf{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_1 - \boldsymbol{\mu}_1^T \boldsymbol{\Sigma}^{-1} \mathbf{x} + \boldsymbol{\mu}_1^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_1 \\ &\quad - \cancel{\mathbf{x}^T \boldsymbol{\Sigma}^{-1} \mathbf{x}} + \mathbf{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_0 + \boldsymbol{\mu}_0^T \boldsymbol{\Sigma}^{-1} \mathbf{x} - \boldsymbol{\mu}_0^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_0] \\ &= \frac{1}{2} [2\mathbf{x}^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) - (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)] \\ &= \mathbf{x}^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) - \frac{1}{2} (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) \end{aligned}$$

Derivation 2/4

Linear Discriminant Analysis: 2 Class

Σ is the same for both classes

$$\ln \left| \frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)} \right| = \mathbf{x}^T \Sigma^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) - \frac{1}{2} (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)^T \Sigma^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

Since our decision rule is to classify as class 1 if the following is true:

$$\ln \left| \frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)} \right| > \ln \left| \frac{P(y=0)}{P(y=1)} \right|$$

We can rewrite our decision rule as:

(see appendix slides
for full derivation)

$$\underline{\mathbf{x}^T \Sigma^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) > \frac{1}{2} (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)^T \Sigma^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) + \ln \left| \frac{P(y=0)}{P(y=1)} \right|}$$

Project the data into a
1-dimensional space

Set the threshold for classifying as
class 1 or 0

Derivation 3/4

Linear Discriminant Analysis: 2 Class

Σ is the same for both classes

$$\ln \left| \frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)} \right| = \mathbf{x}^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) - \frac{1}{2} (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

Since our decision rule is to classify as class 1 if the following is true:

$$\ln \left| \frac{P(\mathbf{x}|y=1)}{P(\mathbf{x}|y=0)} \right| > \ln \left| \frac{P(y=0)}{P(y=1)} \right|$$

We can rewrite our decision rule as:

$$\mathbf{x}^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) > \frac{1}{2} (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) + \ln \left| \frac{P(y=0)}{P(y=1)} \right|$$

Or simply as: $\mathbf{x}^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) > \lambda_{thresh}$

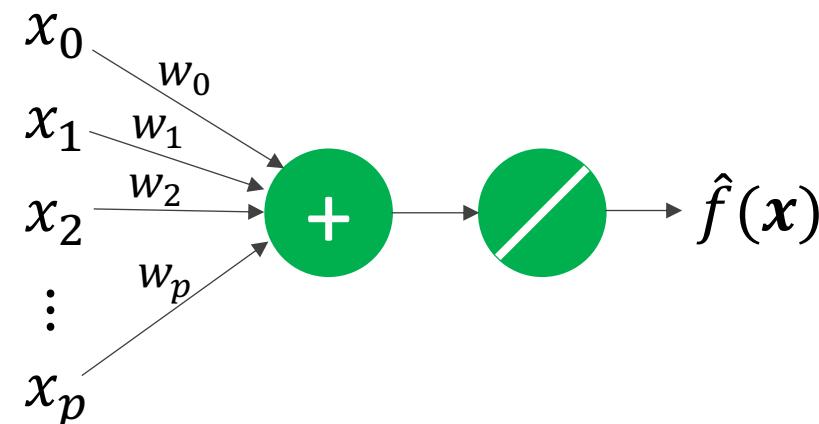
If we define $\mathbf{w} = \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$, then this becomes $\mathbf{x}^T \mathbf{w} = \mathbf{w}^T \mathbf{x} > \lambda_{thresh}$

Derivation 4/4

Remember linear models?

Linear Regression

$$\hat{f}(\mathbf{x}) = \sum_{i=0}^p w_i x_i$$
$$\mathbf{w}^T \mathbf{x}$$

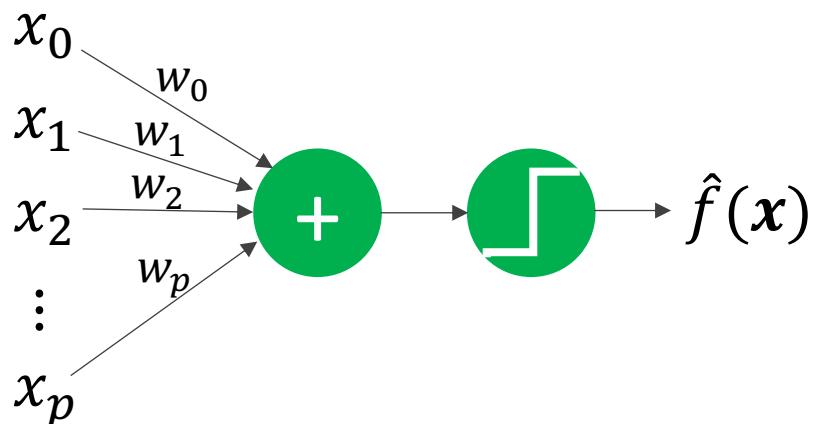


Linear Classification

Perceptron

$$\hat{f}(\mathbf{x}) = sign \left(\sum_{i=0}^p w_i x_i \right)$$

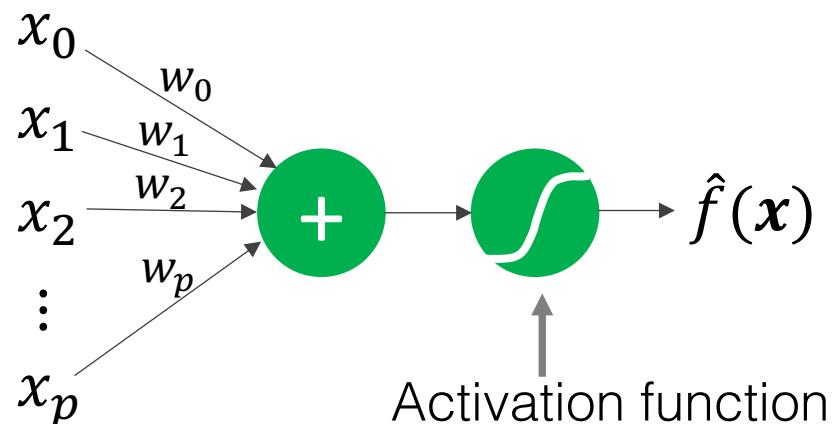
$$sign(x) = \begin{cases} 1 & x > 0 \\ -1 & \text{else} \end{cases}$$



Logistic Regression

$$\hat{f}(\mathbf{x}) = \sigma \left(\sum_{i=0}^p w_i x_i \right)$$

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



Source: Abu-Mostafa, Learning from Data, Caltech

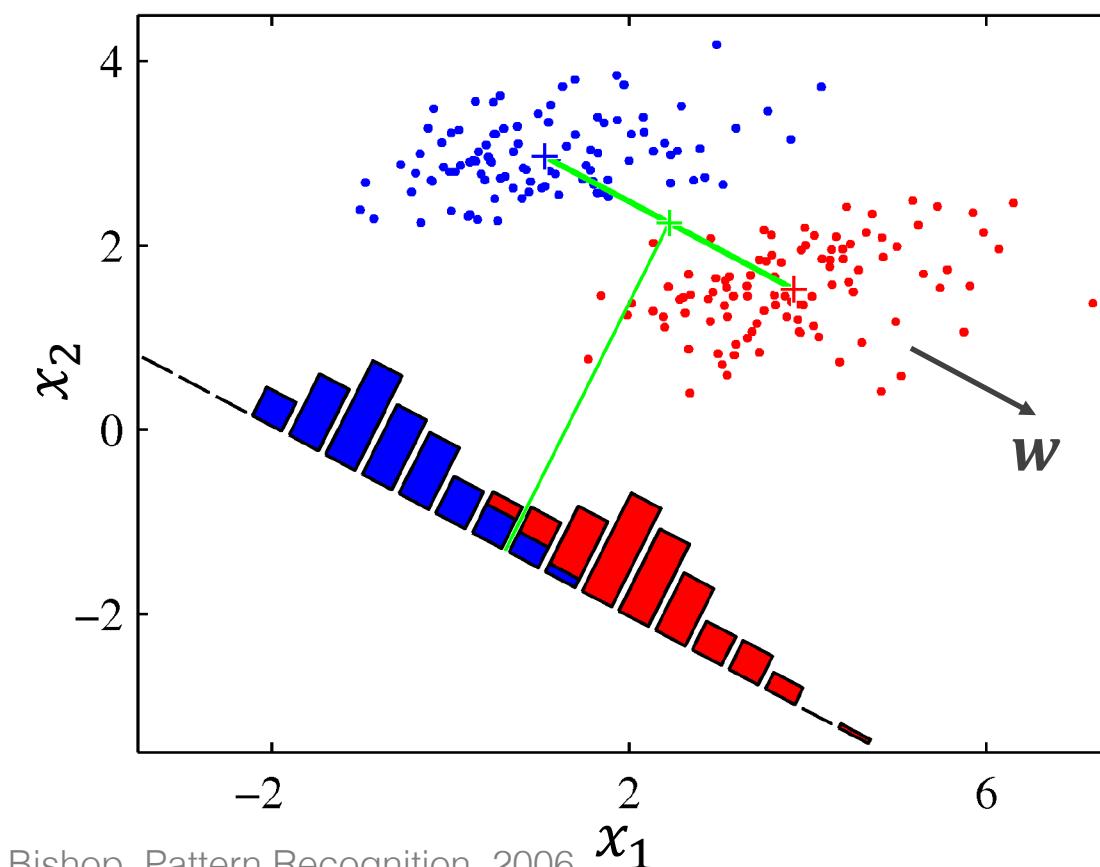
**This approach is a supervised
dimensionality reduction technique
that we use for classification**

(To extend to >2 classes pick the one with the largest $d_i(x)$)

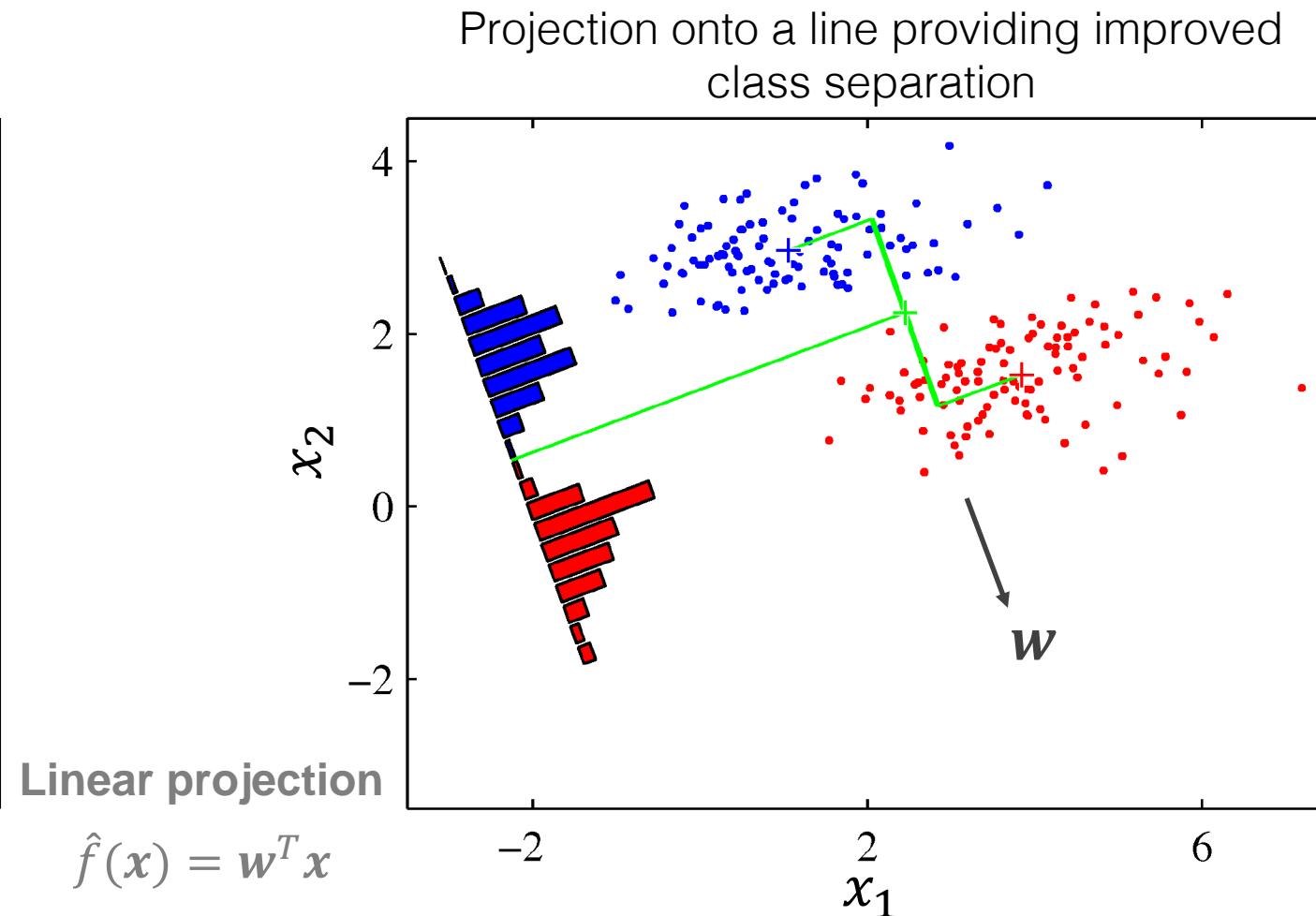
Linear Discriminant Analysis

Looks for the projection into the one dimension that “best” separates the classes

Projection onto line connecting the means



Projection onto a line providing improved class separation



Linear projection

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

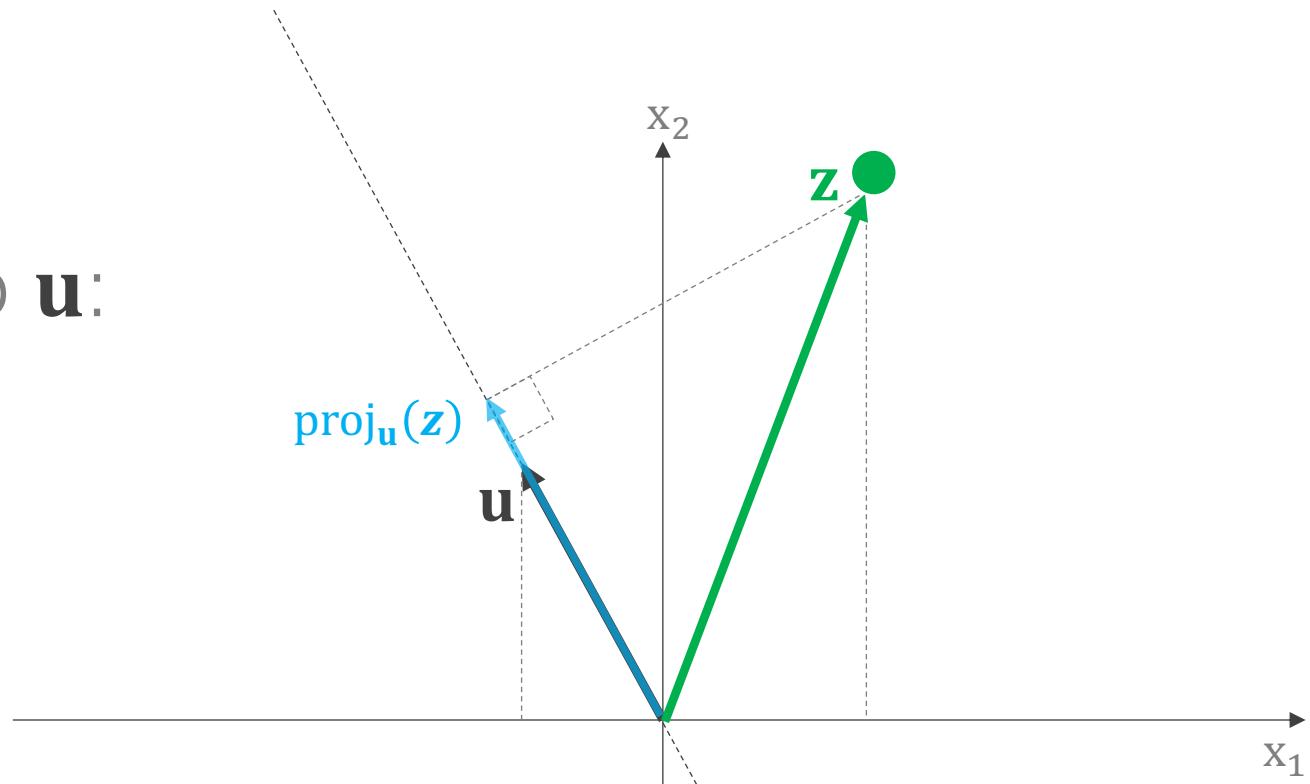
Intuitively interpreting LDA in terms of projections

(We'll also use this for our discussion of principle components analysis in a later lecture)

Projections

The vector projection of \mathbf{z} onto \mathbf{u} :

$$\text{proj}_{\mathbf{u}}(\mathbf{z}) = \left(\frac{\mathbf{u}^T \mathbf{z}}{\|\mathbf{u}\|} \right) \frac{\mathbf{u}}{\|\mathbf{u}\|}$$



The scalar length (Euclidean or L₂ norm) of the vector \mathbf{u} is $\|\mathbf{u}\|$

If we assume \mathbf{u} is a unit vector then $\|\mathbf{u}\| = 1$

$$\text{proj}_{\mathbf{u}}(\mathbf{z}) = (\mathbf{u}^T \mathbf{z}) \mathbf{u}$$



Magnitude of projection onto direction of \mathbf{u}

Projections

$$\mathbf{u}^T \mathbf{z} = [u_1 \quad u_2] \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

$$= u_1 z_1 + u_2 z_2$$

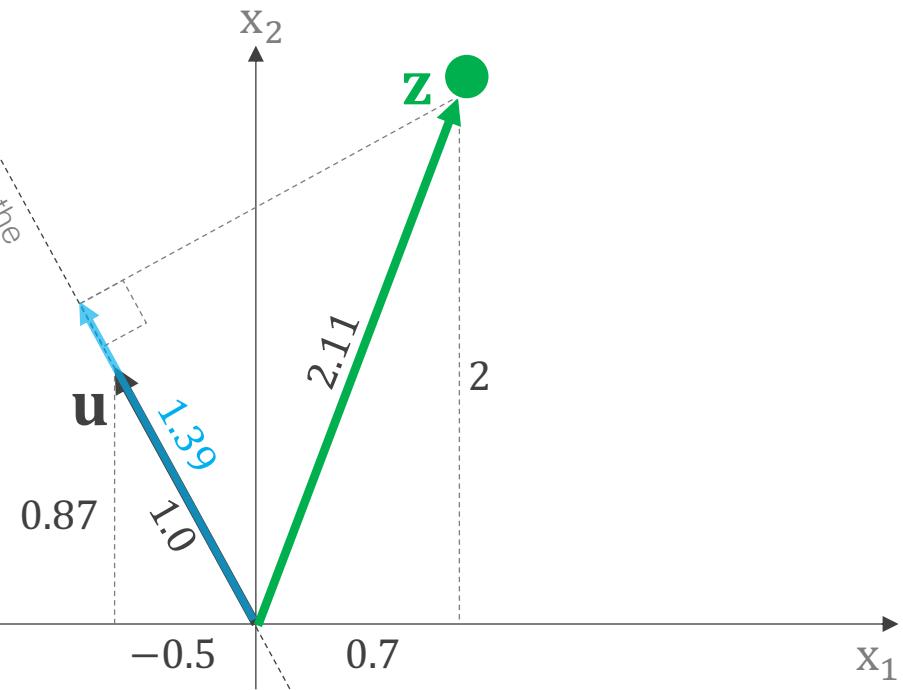
$$= (-0.5)(0.7) + (0.87)(2)$$

$$= 1.39$$

Length (magnitude) of the projection of \mathbf{z} onto \mathbf{u}

This is an inner product, but assuming \mathbf{u} is a unit vector computes this as the magnitude (length) of the projection of \mathbf{z} onto \mathbf{u}

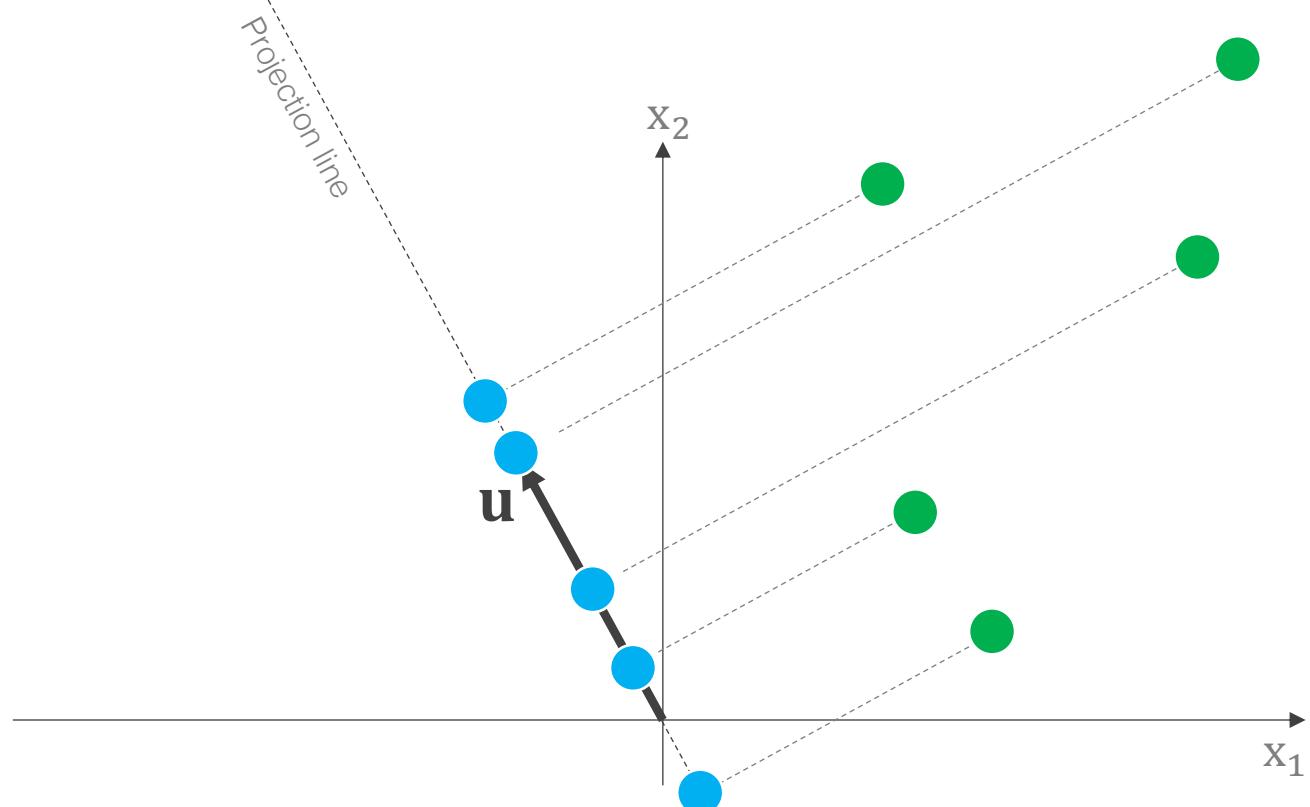
This process projects the data onto the line defined by the direction of \mathbf{u}



This is valid because \mathbf{u} is a unit vector (length is 1: $\|\mathbf{u}\|_2 = \sqrt{u_1^2 + u_2^2} = \sqrt{(-0.5)^2 + (0.87)^2} \cong 1$)

Projections

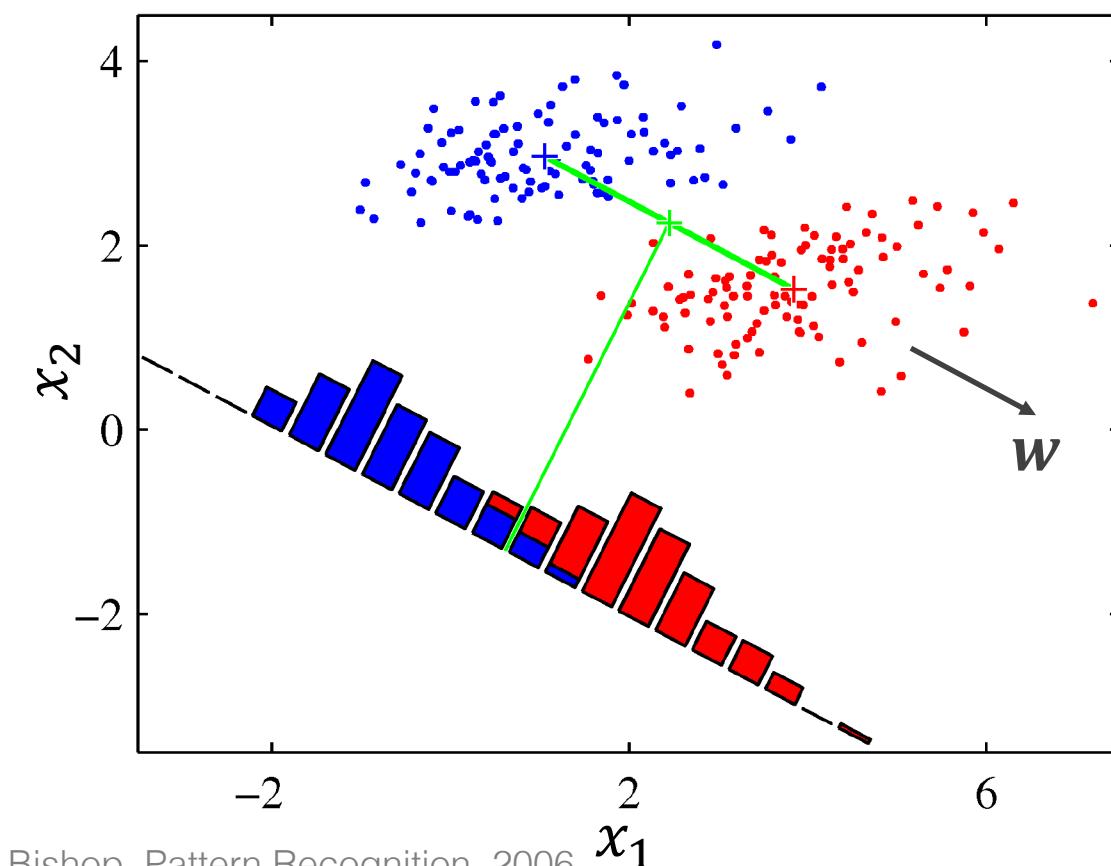
We could project any points in this space onto the line defined by the direction of unit vector \mathbf{u}



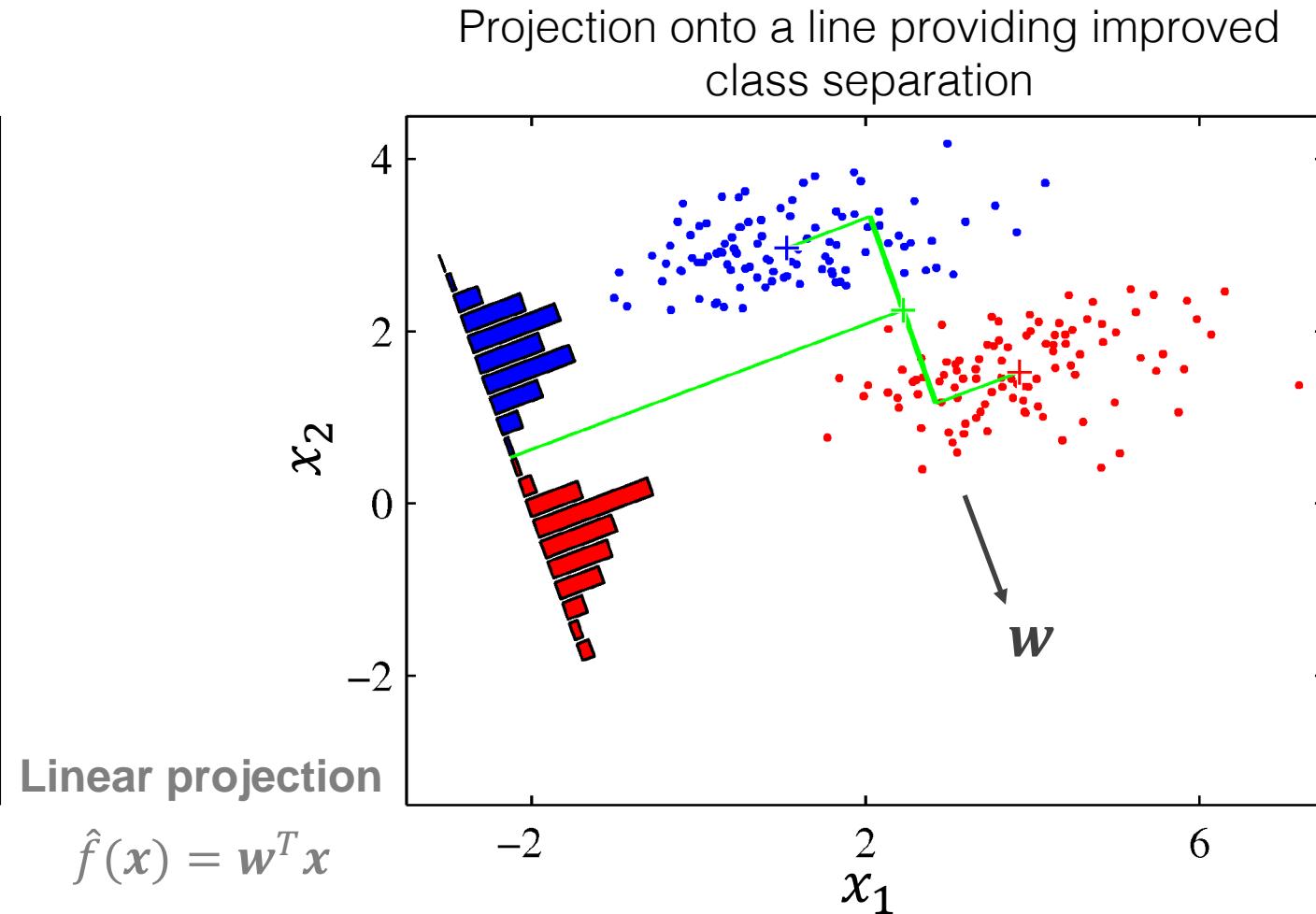
Linear Discriminant Analysis

Looks for the projection into the one dimension that “best” separates the classes

Projection onto line connecting the means



Projection onto a line providing improved class separation



Linear projection

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

Linear Discriminant Analysis (LDA)

- 1 Finds a projection into a lower dimension that “best” separates the classes

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

Consider \mathbf{w} is a unit vector of parameters

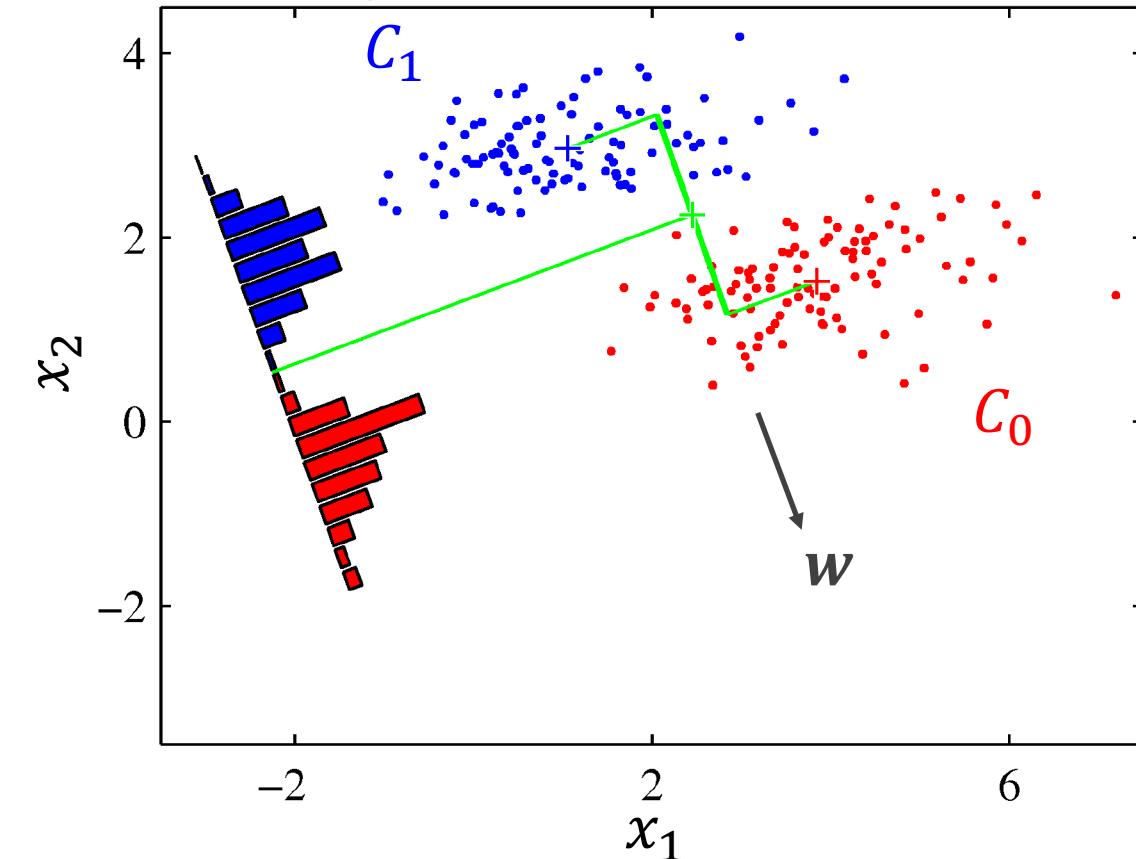
- 2 We then classify the data in this space linearly

Our decision rule becomes:

if	$\mathbf{w}^T \mathbf{x} > \lambda_{thresh}$	Class 1
else		Class 0

2-Class LDA: what does vector w accomplish?

Bishop, Pattern Recognition, 2006



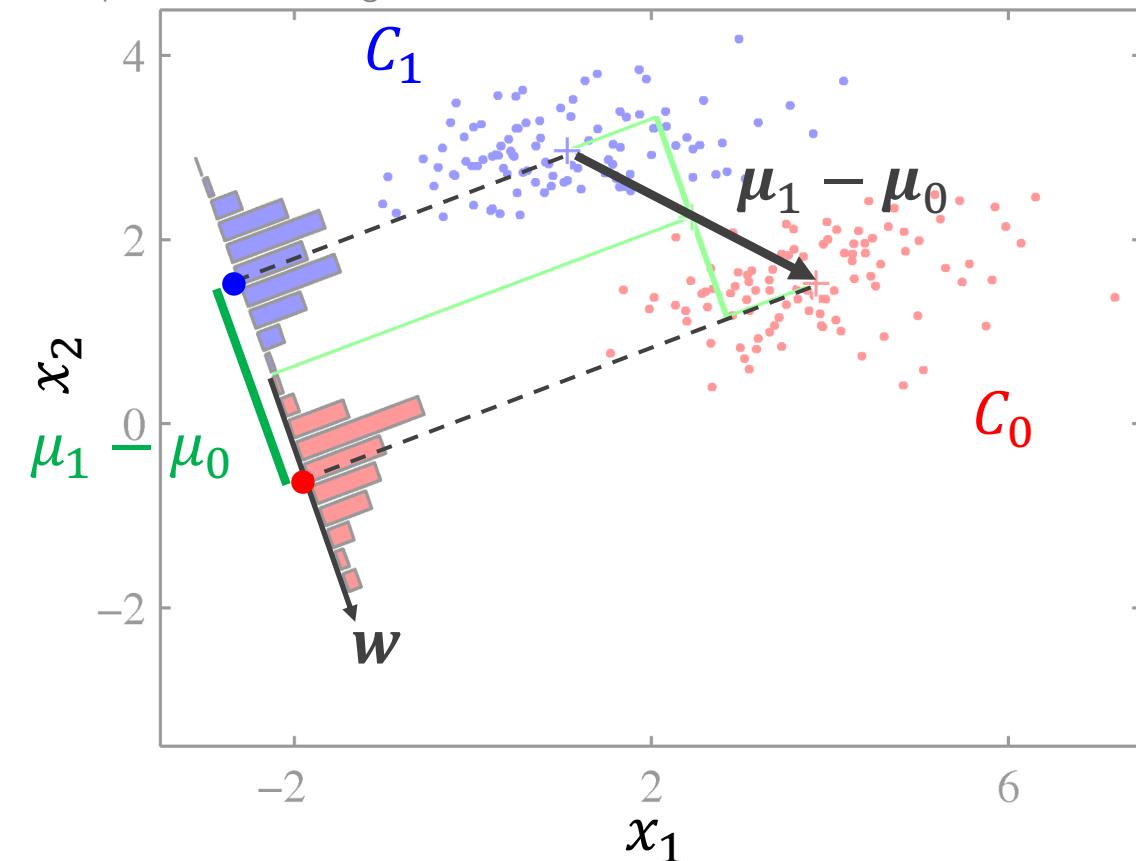
Increases the distance between class **means**

Decreases the **variance** within the classes

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

2-Class LDA: what does vector w accomplish?

Bishop, Pattern Recognition, 2006



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

Increase the distance between the **means**

$$\mu_1 = \frac{1}{N_1} \sum_{i \in C_1} x_i$$

mean of class 1

$$\mu_0 = \frac{1}{N_0} \sum_{i \in C_0} x_i$$

mean of class 0

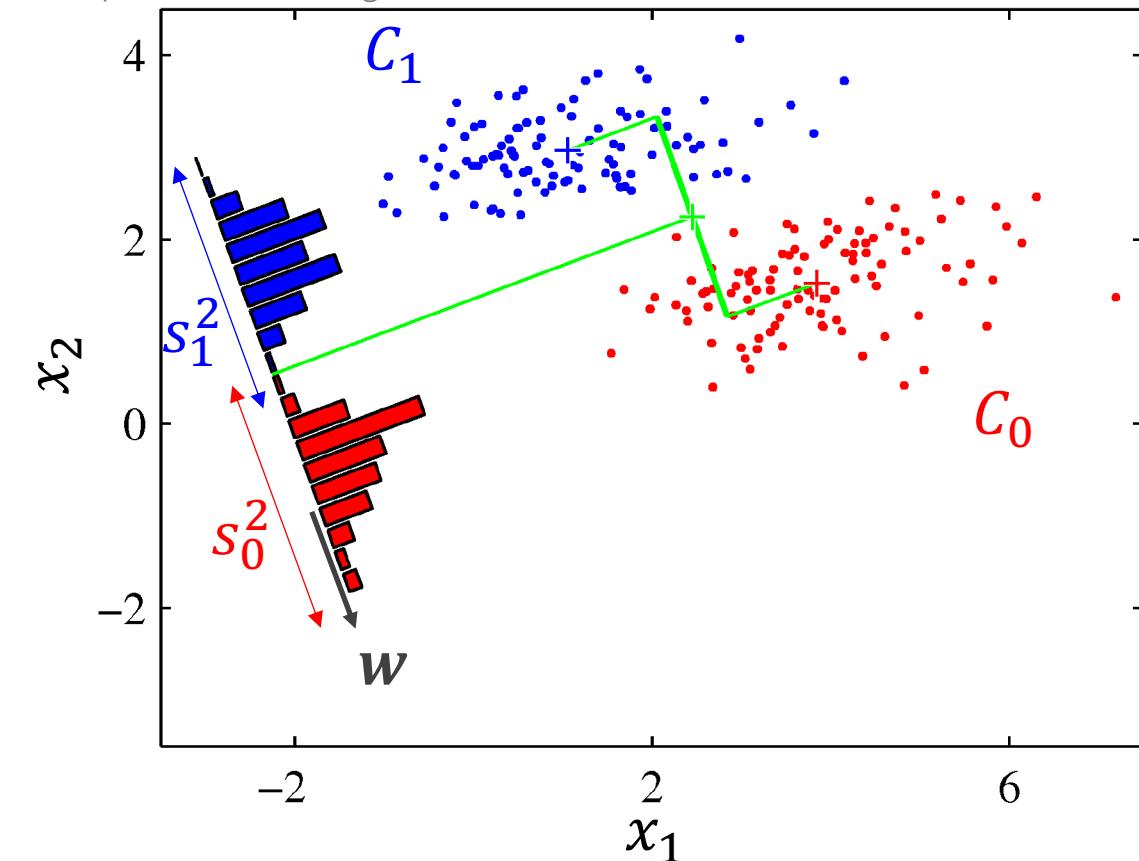
The means projected onto w : $\mu_k = w^T \mu_k$

The distance between the means:

$$\mu_1 - \mu_0 = w^T (\mu_1 - \mu_0)$$

2-Class LDA: what does vector w accomplish?

Bishop, Pattern Recognition, 2006



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

Decreases the **variance** within the classes

The “scatter” of the **projected** data:

$$s_k^2 = \sum_{i \in C_k} (y_i - \mu_k)^2$$

where $\mu_k = w^T \mu_k$

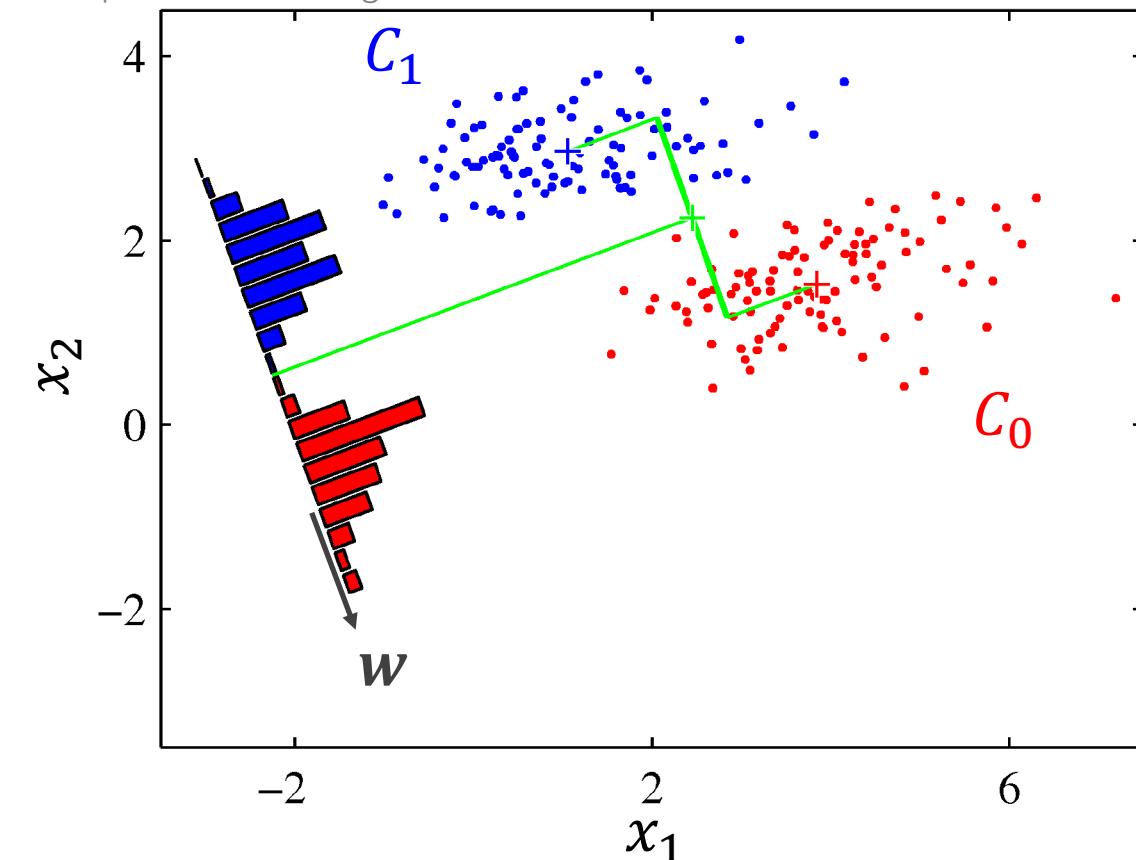
$$y_i = w^T x_i$$

We assume these are the same:

$$S = s_1^2 = s_0^2$$

2-Class LDA: what does vector w accomplish?

Bishop, Pattern Recognition, 2006



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

Increase the distance between the **means**

$$\mu_1 - \mu_0 = w^T(\mu_1 - \mu_0)$$

Decrease the **variance** within each class

$$S = s_1^2 = s_0^2$$

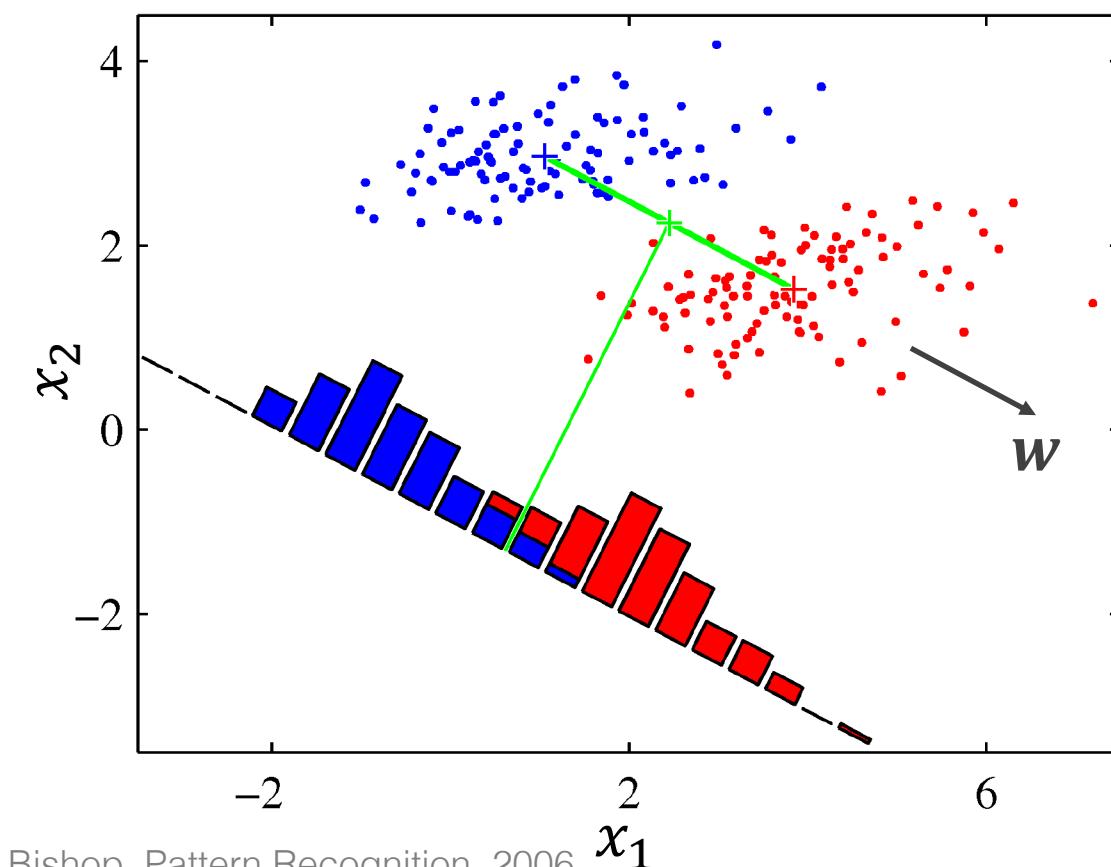
$$w = \Sigma^{-1}(\mu_1 - \mu_0)$$

We use this to project the features into one dimension for classification, $w^T x$

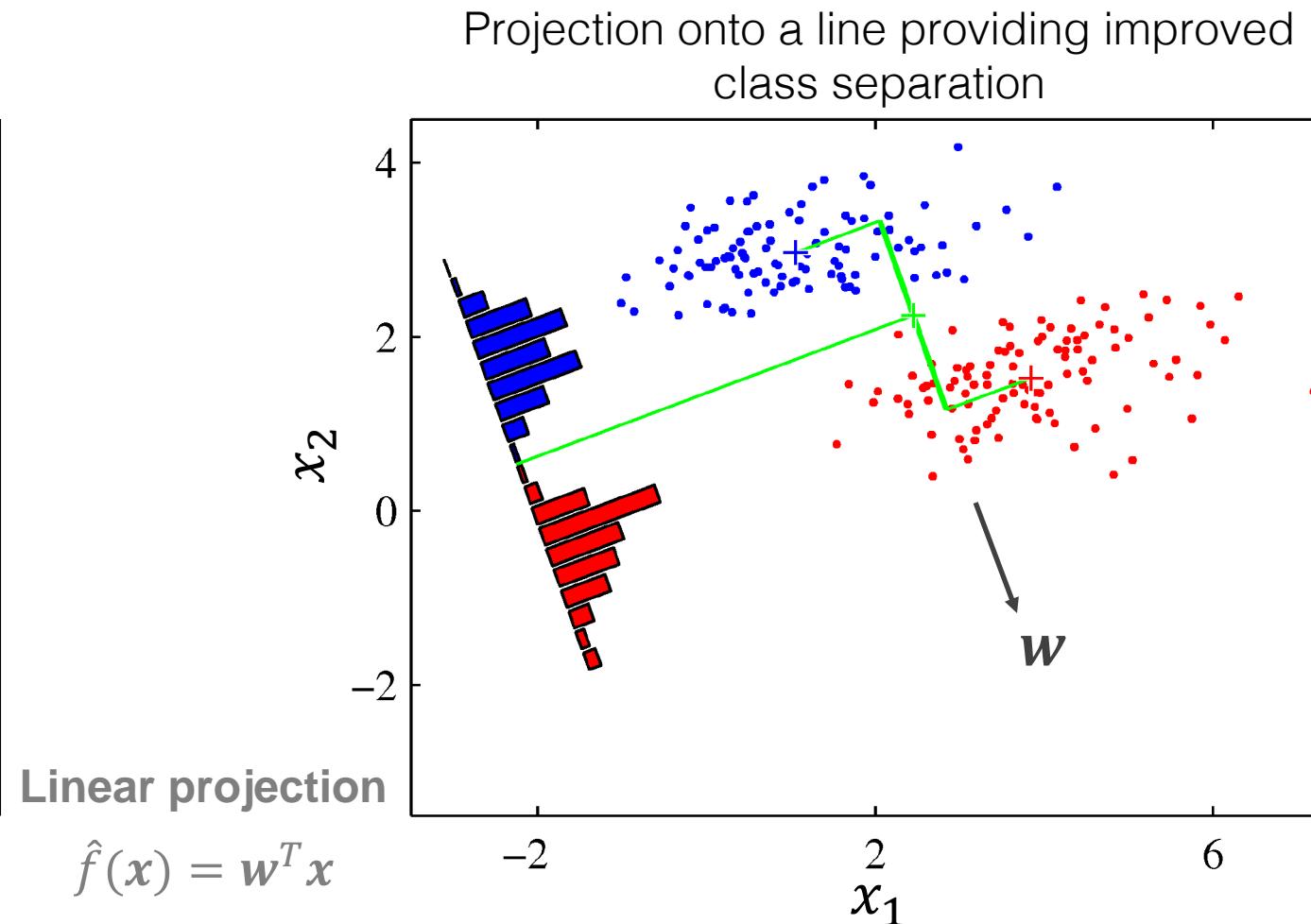
Linear Discriminant Analysis

Looks for the projection into the one dimension that “best” separates the classes

Projection onto line connecting the means



Projection onto a line providing improved class separation



Linear projection

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

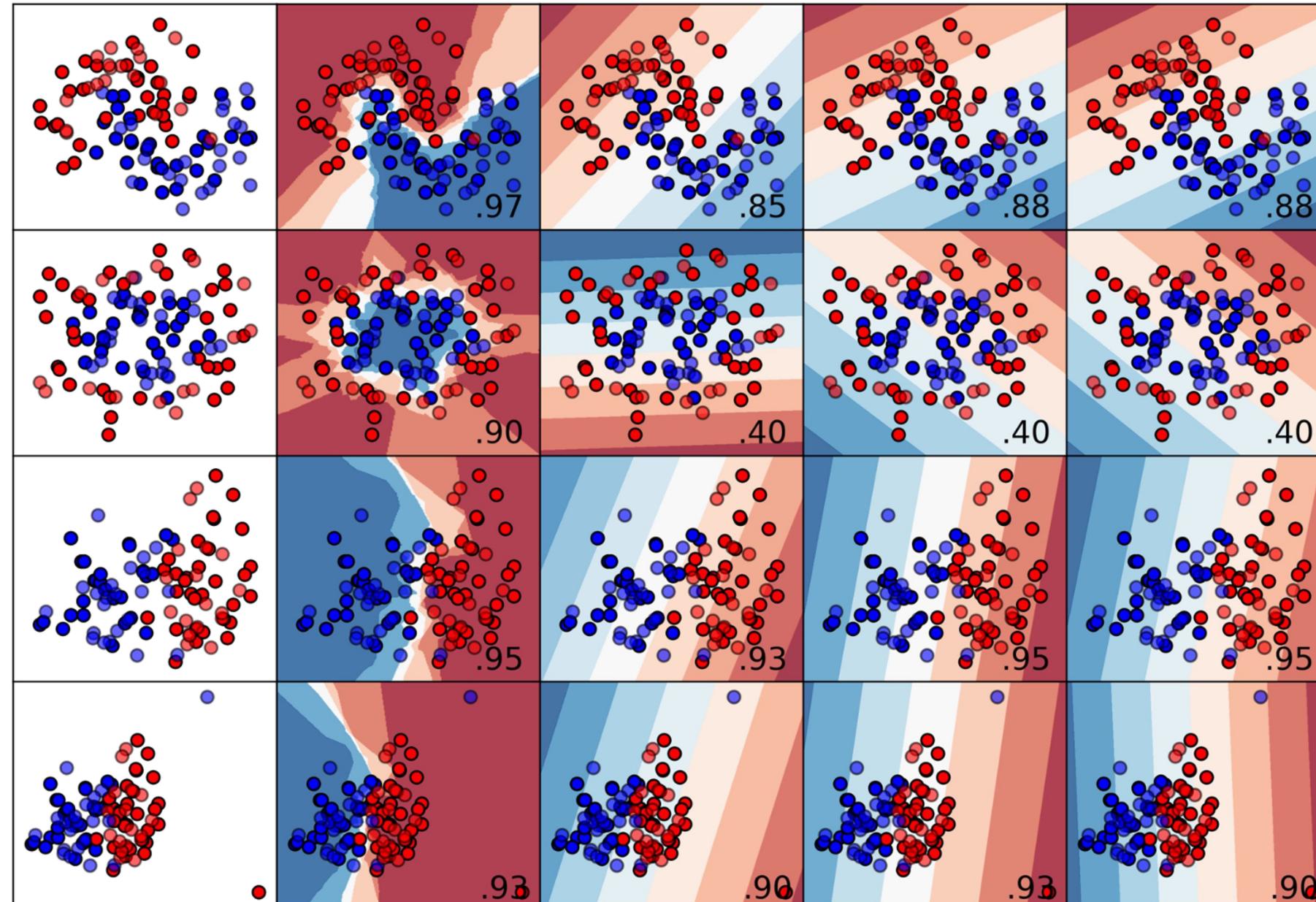
Input data

KNN (k=5)

Perceptron

Logistic Reg.

LDA



Quadratic Discriminant Analysis

We build a classifier that assigns the class with the higher posterior probability:

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

We assume a normal distribution, but **different covariance matrices**

Produces a quadric decision boundary

Input data

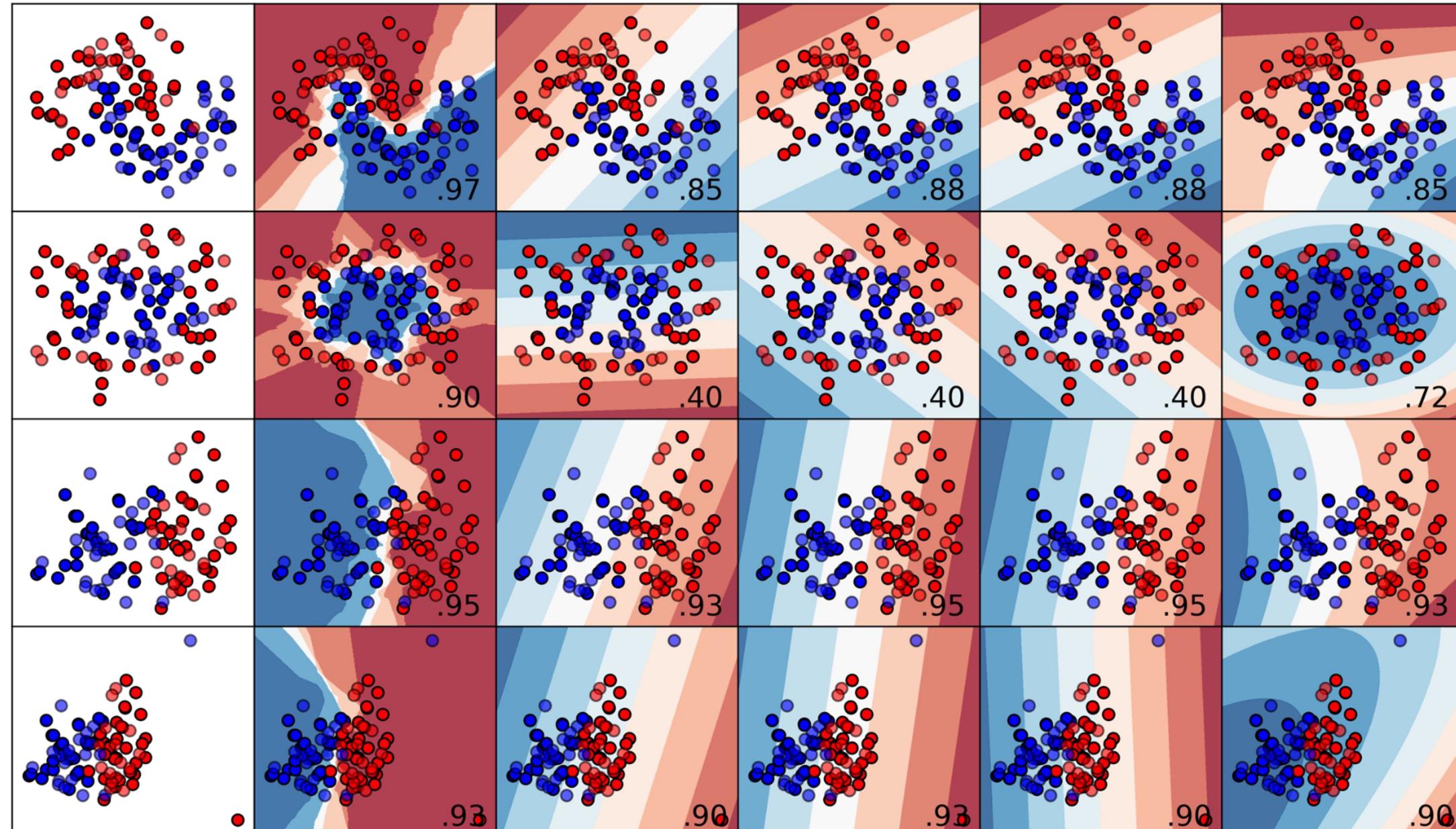
KNN (k=5)

Perceptron

Logistic Reg.

LDA

QDA



Joint vs Marginal Densities

The marginal densities don't factor in relationships between features

What if the joint density is too hard to estimate?

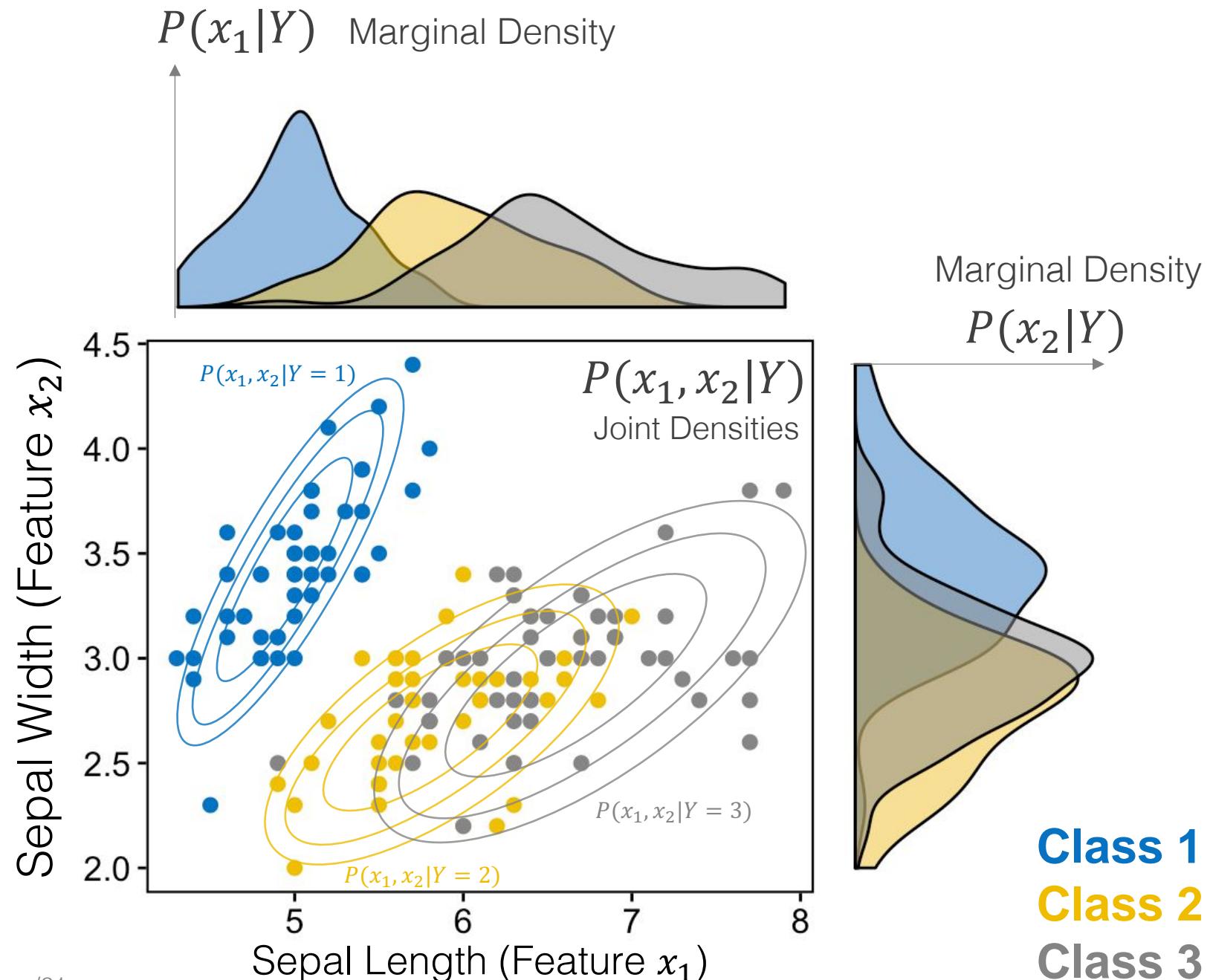


Image adapted from: <https://github.com/daattali/ggExtra/issues/61>

Naïve Bayes

For independent events: A, B, and C
 $P(A \text{ and } B \text{ and } C) = P(A)P(B)P(C)$

Start with our original expression for our discriminant function
(proportional to posterior distribution)

$$d_i(\mathbf{x}) = P(\mathbf{x}|y = i)P(y = i)$$

Write out the full expression with all the terms in \mathbf{x}
(assume p predictors/features)

$$d_i(x_1, x_2, \dots, x_p) = P(x_1, x_2, \dots, x_p | y = i)P(y = i)$$

Assumption: Given the class, the features are independent

$$d_i(x_1, x_2, \dots, x_p) = P(y = i) \prod_{j=1}^p P(x_j | y = i)$$

Predict the class with the highest discriminant function (i.e. posterior probability)

Naïve Bayes

We assign the class that has the largest discriminant (i.e. posterior probability)

$$d_i(x_1, x_2, \dots, x_p) = P(y = i) \prod_{j=1}^p P(x_j | y = i)$$

This implies we estimate the density of each feature **separately**

This independence assumption is a strong assumption that is rarely valid

Considerably simplifies computation and data needs

Is flexible to allow for different distributional forms (i.e. Gaussian) or nonparametric techniques for the likelihood $P(x_j | y = i)$

Naïve Bayes: Gaussian example

We assign the class that has the largest discriminant (i.e. posterior probability)

$$d_i(x_1, x_2, \dots, x_p) = P(y = i) \prod_{j=1}^p P(x_j | y = i)$$

This implies we estimate the density of each feature **separately**

If $P(x_j | y = i)$ is $N(\mu_{ji}, \sigma_{ji}^2)$, so for each class we estimate one mean and variance for each of the p features and for each class. We multiply **univariate** distributions together

$$d_i(x_1, x_2, \dots, x_p) = P(y = i) \prod_{j=1}^p N(\mu_{ji}, \sigma_{ji}^2)$$

Naïve Bayes: Parameters

p predictors, c classes

For each predictor, x_i , and class, y_j :

$$(\mu_{ij}, \sigma_{ij}^2)$$

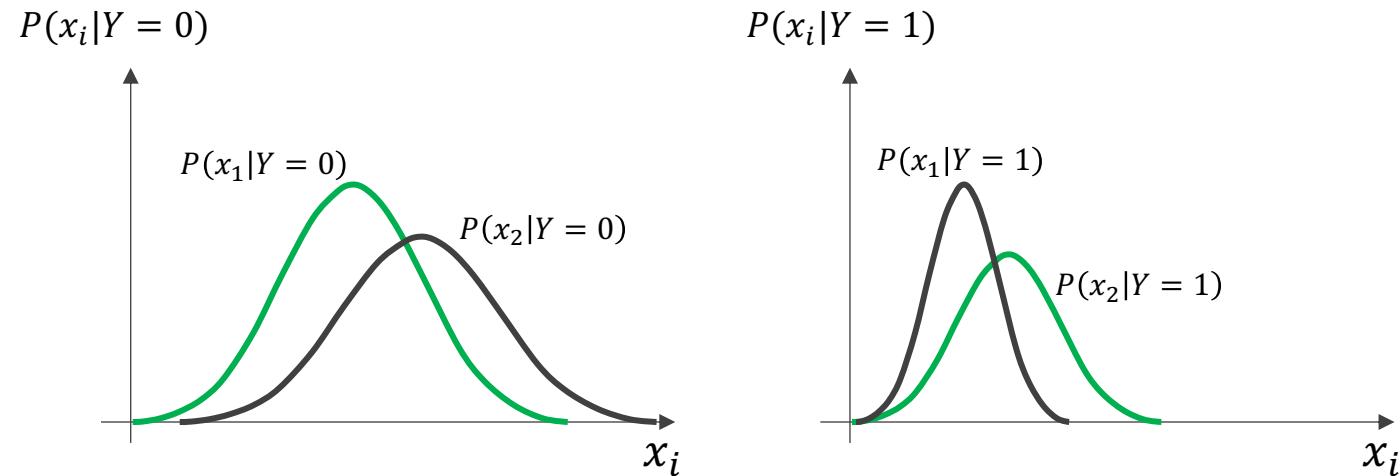
Total parameters = $2cp$

Without the Naïve Bayes independence assumption, each class would be a multivariate Gaussian with $(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$

$$\boldsymbol{\mu}_j = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_p \end{bmatrix}, \boldsymbol{\Sigma}_j = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \cdots & \sigma_{1p}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \cdots & \sigma_{2p}^2 \\ \vdots & \vdots & & \vdots \\ \sigma_{p1}^2 & \sigma_{p2}^2 & \cdots & \sigma_{pp}^2 \end{bmatrix}$$

Total parameters = $c \left(p + \frac{p^2 - p}{2} + p \right)$

Gaussian Naïve Bayes ($p = 2$)



Multivariate Gaussian ($p = 2$)

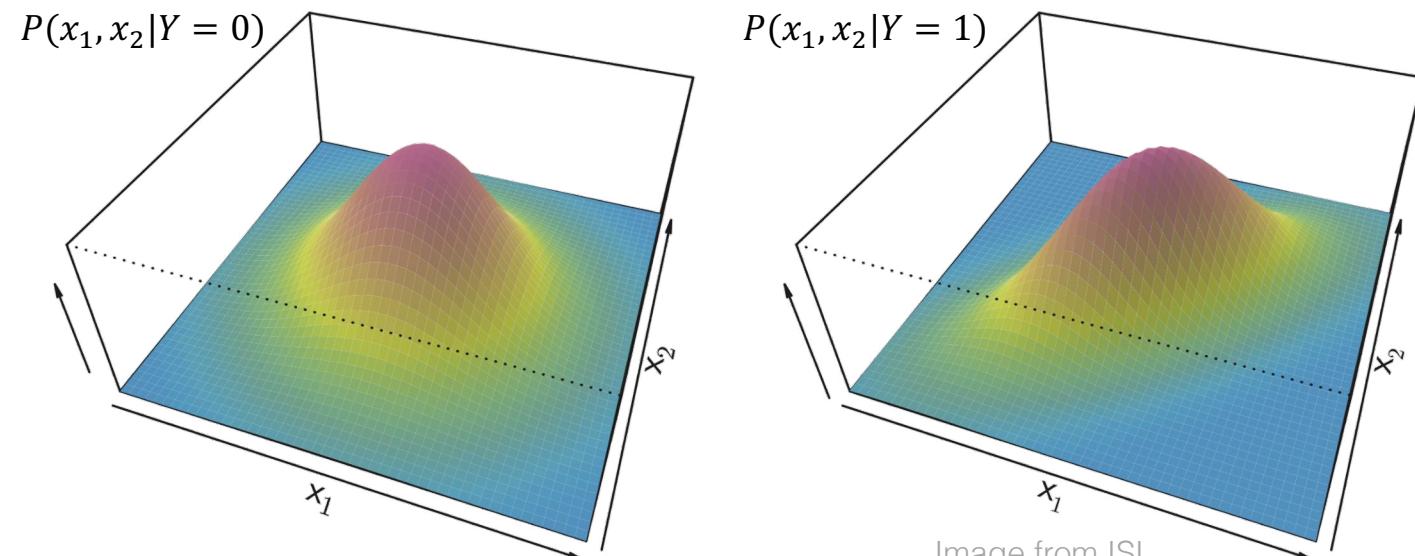


Image from ISL

Input data

KNN (k=5)

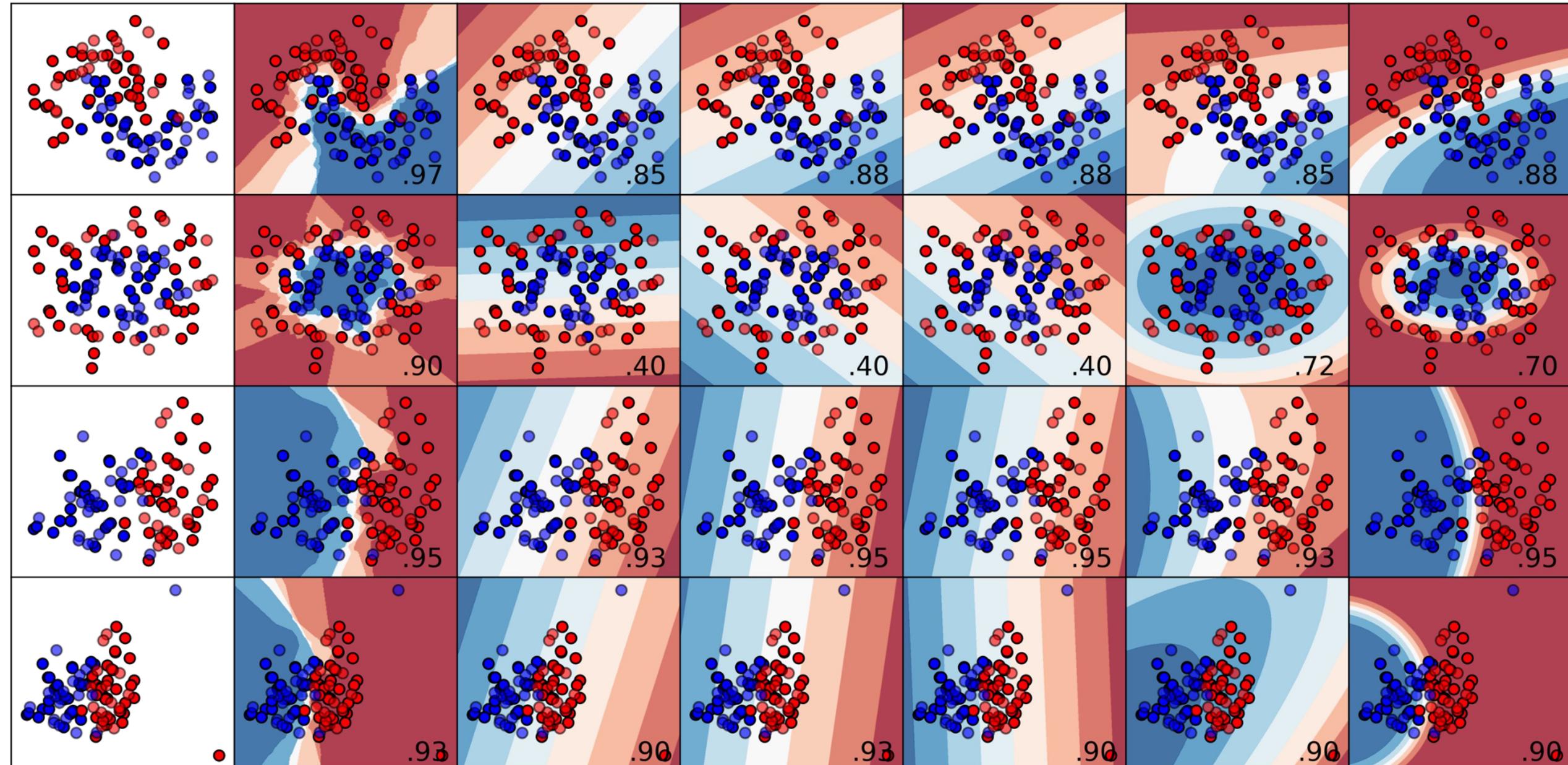
Perceptron

Logistic Reg.

LDA

QDA

Naive Bayes



Classifiers

Covered so far

K-Nearest Neighbors

Perceptron

Logistic Regression

Linear Discriminant Analysis

Quadratic Discriminant Analysis

Naïve Bayes

Have closed-form solutions

Apply to multiclass problems

No hyperparameters

Fast to train

Requires small amounts of training data

Only choice is the form of $P(X|Y)$
(otherwise no parameter choices)

Fast to train