CS 189 Introduction to Machine Learning
Spring 2022 Marvin Zhang

DIS3

1 Logistic Regression

Assume that we have n i.i.d. data points $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, where each y_i is a binary label in $\{0, 1\}$. We model the posterior probability as a Bernoulli distribution and the probability for each class is the sigmoid function, i.e., $p(y|\mathbf{x}; \mathbf{w}) = q^y(1-q)^{1-y}$, where $q = s(\mathbf{w}^\top \mathbf{x})$ and $s(z) = \frac{1}{1+\exp\{-z\}}$ is the sigmoid function.

(a) **Show that the derivative of the sigmoid function is:** $s'(z) = s(z)(1 - s(z))$

Solution: Use the quotient rule:

$$\begin{aligned} s'(z) &= \frac{(\frac{d}{dz}1)(1+\exp\{-z\}) - (\frac{d}{dz}(1+\exp\{-z\}))(1)}{(1+\exp\{-z\})^2} \\ &= \frac{\exp\{-z\}}{(1+\exp\{-z\})^2} \\ &= \frac{1}{1+\exp\{-z\}}\frac{\exp\{-z\}}{1+\exp\{-z\}} \\ &= s(z)(1-s(z)) \end{aligned}$$

(b) **Write out the likelihood and log likelihood functions, along with the MLE objective.** Comment on whether it is possible to find a closed form maximum likelihood estimate of $\mathbf{w}$, and describe an alternate approach.

Solution:

The likelihood is:

$$L(\mathbf{w}) = \prod_{i=1}^{n} p(y = y_i|\mathbf{x}_i) = \prod_{i=1}^{n} q_i^{y_i}(1-q_i)^{1-y_i}.$$

Now maximizing the likelihood of the training data as a function of the parameters $\mathbf{w}$, we get:

$$\begin{aligned} \hat{\mathbf{w}} &= \arg\max_{\mathbf{w}} L(\mathbf{w}) = \arg\max_{\mathbf{w}} \prod_{i=1}^{n} q_i^{y_i}(1-q_i)^{1-y_i} \\ &= \arg\max_{\mathbf{w}} \sum_{i=1}^{n} y_i \log(q_i) + (1-y_i)\log(1-q_i) \\ &= \arg\max_{\mathbf{w}} \sum_{i=1}^{n} y_i \log\Big(s(\mathbf{w}^\top \mathbf{x}_i)\Big) + (1-y_i)\log\Big(1 - s(\mathbf{w}^\top \mathbf{x}_i)\Big) \end{aligned}$$

Multiplying everything by -1 and switching to minimization we have:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \sum_{i=1}^{n} -y_i \log\Big(s(\mathbf{w}^\top \mathbf{x}_i)\Big) - (1-y_i)\log\Big(1 - s(\mathbf{w}^\top \mathbf{x}_i)\Big)$$

This is the binary cross entropy cost function that you may have seen/used before!

(c) **Calculate the gradient, and write the stochastic gradient descent update step with learning rate η, where the gradient step is calculated on a single data point $(\mathbf{x}_i, y_i)$.**

Solution:

Let us denote $J(\mathbf{w}) = \sum_{i=1}^{n} -y_i \log(s(\mathbf{w}^\top \mathbf{x}_i)) - (1-y_i)\log(1 - s(\mathbf{w}^\top \mathbf{x}_i))$. Notice that $J(\mathbf{w})$ is convex in $\mathbf{w}$, so there is a global minimum. Recall that $s'(\zeta) = s(\zeta)(1 - s(\zeta))$. Now let us take the gradient of $J(\mathbf{w})$ w.r.t $\mathbf{w}$:

$$\begin{aligned}
\nabla_{\mathbf{w}} J(\mathbf{w}) &= \sum_{i=1}^{n} -y_i \nabla_{\mathbf{w}} \log\Big(s(\mathbf{w}^\top \mathbf{x}_i)\Big) - (1-y_i)\nabla_{\mathbf{w}} \log\Big(1 - s(\mathbf{w}^\top \mathbf{x}_i)\Big) \\
&= \sum_{i=1}^{n} -y_i(\mathbf{x}_i \cdot s'(\mathbf{w}^\top \mathbf{x}_i) \cdot \frac{1}{s(w^\top \mathbf{x}_i)}) - (1-y_i)(\mathbf{x}_i \cdot -s'(\mathbf{w}^\top \mathbf{x}_i) \cdot \frac{1}{1 - s(w^\top \mathbf{x}_i)}) \\
&= \sum_{i=1}^{n} -y_i(\mathbf{x}_i \cdot s_i(1-s_i) \cdot \frac{1}{s_i}) - (1-y_i)(\mathbf{x}_i \cdot -s_i(1-s_i) \cdot \frac{1}{1-s_i}) \\
&= \sum_{i=1}^{n} -y_i(\mathbf{x}_i \cdot (1-s_i)) - (1-y_i)(\mathbf{x}_i \cdot -s_i) \\
&= \sum_{i=1}^{n} -y_i(1-s_i)\mathbf{x}_i + (1-y_i)s_i\mathbf{x}_i \\
&= \sum_{i=1}^{n} (s_i - y_i)\mathbf{x}_i \\
&= \mathbf{X}^\top(\mathbf{s} - \mathbf{y})
\end{aligned}$$

where, $s_i = s(\mathbf{w}^\top \mathbf{x}_i), \mathbf{s} = (s_1, \ldots, s_n)^\top, \mathbf{y} = (y_1, \ldots, y_n)^\top$ and $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix}$.

When we take the gradient $\nabla_{\mathbf{w}} \log(s(\mathbf{w}^\top \mathbf{x}_i))$, we're taking the gradient of a composed function of the form $\nabla_{\mathbf{w}} f(g(h(\mathbf{w})))$, where $f(g) = \log g$, $g(h) = s(h)$, $h(\mathbf{w}) = \mathbf{w}^\top \mathbf{x}$. We can apply the chain rule for this computation:

$$\nabla_{\mathbf{w}} f(g(h(\mathbf{w}))) = (\frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial h} \cdot \frac{\partial h}{\partial \mathbf{w}})^\top \qquad (\nabla_{\mathbf{w}} f(\mathbf{w}) = (\frac{\partial f}{\partial \mathbf{w}})^\top)$$

$$
\begin{aligned}
&= (\frac{\partial h}{\partial \mathbf{w}})^\top \cdot (\frac{\partial g}{\partial h})^\top \cdot (\frac{\partial f}{\partial g})^\top \\
&= \nabla_{\mathbf{w}} h \cdot \nabla_h g \cdot \nabla_g f \\
&= \nabla_{\mathbf{w}} (\mathbf{w}^\top \mathbf{x}) \cdot \nabla_{\mathbf{w}^\top \mathbf{x}} s(\mathbf{w}^\top \mathbf{x}) \cdot \nabla_s \log s \\
&= \mathbf{x} \cdot s'(\mathbf{w}^\top \mathbf{x}) \cdot \frac{1}{s(\mathbf{w}^\top \mathbf{x})}
\end{aligned}
$$

Unfortunately from our gradient calculated above, we can't get a closed form estimate for $\mathbf{w}$ by setting the derivative to zero. However, we can iteratively optimize using gradient descent.

The batch gradient descent update is therefore:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{X}^\top (\mathbf{s} - \mathbf{y})$$

To get the SGD update, modify the batch GD update to only use one data point:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta (s(\mathbf{w}_t^\top \mathbf{x}_i) - y_i) \mathbf{x}_i$$

2 Gaussian Classification: LDA and QDA

Gaussian discriminant analysis (GDA) is a generative classification method, which involves modeling the posterior probability by approximating the underlying class-conditional data distribution $p_\theta(\mathbf{x} \mid y)$ and class priors $p_\theta(y)$. We call this "generative" because we model the *generating* distribution of the data.

The fundamental assumption that GDA makes is that the class-conditional data distribution is Gaussian, and the priors over classes form a Bernoulli distribution (or a multinomial distribution with > 2 classes):

$$p_\theta(\mathbf{x} \mid y = C_i) \sim \mathcal{N}(\mu_i, \mathbf{\Sigma}_i)$$

$$p_\theta(y) \sim \text{Bernoulli}(\pi)$$

The three steps for performing GDA are as follows:

1. Find the parameters of the Gaussian class-conditional data distribution and the prior probabilities using MLE on the (labeled) dataset.
2. Combine the two distributions to produce a quantity proportional to the posterior:

$$p_\theta(y \mid \mathbf{x}) \propto p_\theta(\mathbf{x} \mid y) p_\theta(y)$$

3. Construct a classifier to determine the class of an input point based on the class with the maximum posterior probability:

$$\text{Classifier}_{\text{GDA}}(\mathbf{x}) = \underset{C_1, \ldots, C_k}{\arg\max}\ p_\theta(Y = C_i \mid \mathbf{x})$$

We will focus the binary classification case in this worksheet. However, one nice property of GDA is that it scales up to multi-class classification very easily.

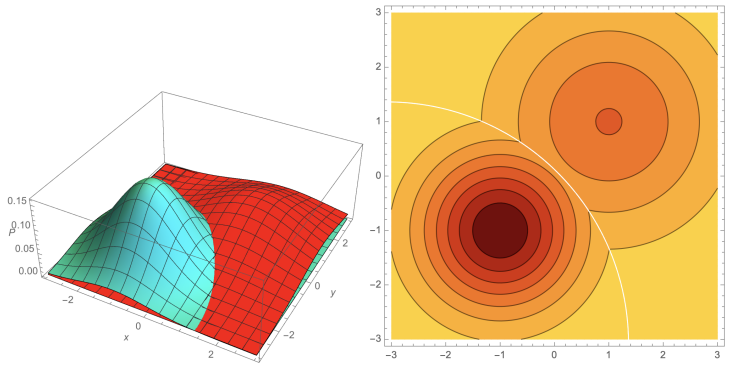

Figure 1: Figure taken from Professor Shewchuck's notes

(a) (Step 1) **Given a set of points $\mathcal{D}_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_{n_i}\}$ labeled as the class C_i, what are the MLE estimates for the mean μ_i and covariance matrix Σ_i for this class?**

Solution:

For the mean:

$$\begin{aligned}
\hat{\mu}_{i,\text{MLE}} &= \arg\max_{\mu_i} \; p(\mathbf{x}_1, \ldots \mathbf{x}_{n_i}; \mu_i, \Sigma_i) \\
&= \arg\max_{\mu_i} \; \sum_{j=1}^{n_i} \log p(\mathbf{x}_j; \mu_i, \Sigma_i) \\
&= \arg\max_{\mu_i} \; \sum_{j=1}^{n_i} -\log \sqrt{(2\pi)^d |\Sigma_i|} - \frac{1}{2}(\mathbf{x}_j - \mu_i)^\top \Sigma_i^{-1}(\mathbf{x}_j - \mu_i) = \arg\max_{\mu_i} \; \ell(\mu_i)
\end{aligned}$$

$$\begin{aligned}
\nabla_{\mu_i} \ell(\mu_i) &= \sum_{j=1}^{n_i} \nabla_{\mu_i}(-\frac{1}{2}(\mathbf{x}_j - \mu_i)^\top \Sigma_i^{-1}(\mathbf{x}_j - \mu_i)) \\
&= \sum_{j=1}^{n_i} -\frac{1}{2}\nabla_{\mu_i}(\mathbf{x}_j^\top \Sigma_i^{-1}\mathbf{x}_j - 2\mathbf{x}_j^\top \Sigma_i^{-1}\mu_i + \mu_i^\top \Sigma_i^{-1}\mu_i) \\
&= \sum_{j=1}^{n_i} -\frac{1}{2}(-2\Sigma_i^{-1}\mathbf{x}_j + 2\Sigma_i^{-1}\mu_i) \\
&= \sum_{j=1}^{n_i} \Sigma_i^{-1}\mathbf{x}_j - \Sigma_i^{-1}\mu_i = \mathbf{0} \\
\implies n_i\mu_i &= \sum_{j=1}^{n_i} \mathbf{x}_j \quad \text{(multiplying on the left by } \Sigma_i \text{ and rearranging)} \\
\hat{\mu}_{i,\text{MLE}} &= \frac{1}{n_i}\sum_{j=1}^{n_i} \mathbf{x}_j
\end{aligned}$$

For the covariance matrix (see Discussion 0 vector calculus review and Lecture 6 for more explanation):

$$\begin{aligned}
\nabla_{\Sigma_i^{-1}} \ell(\Sigma_i) &= \sum_{j=1}^{n_i} \nabla_{\Sigma_i^{-1}}(-\log \sqrt{(2\pi)^d |\Sigma_i|} - \frac{1}{2}(\mathbf{x}_j - \mu_i)^\top \Sigma_i^{-1}(\mathbf{x}_j - \mu_i)) \\
&= \sum_{j=1}^{n_i} \frac{1}{2}\nabla_{\Sigma_i^{-1}}(-\log |\Sigma_i| - (\mathbf{x}_j - \mu_i)^\top \Sigma_i^{-1}(\mathbf{x}_j - \mu_i)) \qquad \text{(removing constants)} \\
&= \sum_{j=1}^{n_i} \frac{1}{2}\nabla_{\Sigma_i^{-1}}(\log |\Sigma_i^{-1}| - (\mathbf{x}_j - \mu_i)^\top \Sigma_i^{-1}(\mathbf{x}_j - \mu_i)) \qquad (|\Sigma^{-1}| = \frac{1}{|\Sigma|}) \\
&= \sum_{j=1}^{n_i} \frac{1}{2}(\Sigma_i - (\mathbf{x}_j - \mu_i)(\mathbf{x}_j - \mu_i)^\top) = \mathbf{0} \quad (\nabla_{\Sigma^{-1}} \log |\Sigma^{-1}| = \Sigma, \nabla_{\Sigma^{-1}} \mathbf{v}^\top \Sigma^{-1}\mathbf{v} = \mathbf{v}\mathbf{v}^\top)
\end{aligned}$$

$$\implies n_i\Sigma = \sum_{j=1}^{n_i}(\mathbf{x}_j - \mu_i)(\mathbf{x}_j - \mu_i)^\top$$

$$\hat{\Sigma}_{i,\text{MLE}} = \frac{1}{n_i}\sum_{j=1}^{n_i}(\mathbf{x}_j - \mu_i)(\mathbf{x}_j - \mu_i)^\top$$

(b) (Step 1) **Express the MLE estimates of the priors $\pi_0 = p_\theta(Y = 0), \pi_1 = p_\theta(Y = 1)$ in terms of the number of data points in each class n_0, n_1.**

Solution:

First, notice that $\pi_0 = 1 - \pi_1$.

Next, let's find the MLE estimate for π_1.

$$\begin{aligned}
\hat{\pi}_{1,\text{MLE}} &= \arg\max_{\pi_1} \prod_{j=1}^{n} p(y_j; \pi_1) \\
&= \arg\max_{\pi_1} \prod_{j=1}^{n} (\pi_1)^{y_j}(1-\pi_1)^{1-y_j} \\
&= \arg\max_{\pi_1} \sum_{j=1}^{n} \log(\pi_1)^{y_j}(1-\pi_1)^{1-y_j} \\
&= \arg\max_{\pi_1} \sum_{j=1}^{n} y_j \log(\pi_1) + (1-y_j)\log(1-\pi_1) = \arg\max_{\pi_1} \ell(\pi_1)
\end{aligned}$$

$$\begin{aligned}
\nabla_{\pi_1}\ell(\pi_1) &= \sum_{j=1}^{n} \frac{y_j}{\pi_1} - \frac{1-y_j}{1-\pi_1} \\
&= \sum_{j:y_j=1} \frac{y_j}{\pi_1} - \sum_{j:y_j=0} \frac{1-y_j}{1-\pi_1} = 0
\end{aligned}$$

$$\begin{aligned}
\frac{n_1}{\pi_1} - \frac{n_0}{1-\pi_1} &= 0 \\
\hat{\pi}_{1,\text{MLE}} &= \frac{n_1}{n_0+n_1} = \frac{n_1}{n} \\
\implies \hat{\pi}_{0,\text{MLE}} &= \frac{n_0}{n_0+n_1} = \frac{n_0}{n}
\end{aligned}$$

The MLE solution results in the priors being equal to the observed label distribution (number of examples of class i over total number of examples).

(c) (Steps 2-3) **Write an equation describing the decision boundary where the posterior probabilities are equal.** Leave it as a quadratic form, no need to simplify fully.

$$p_{\theta_0=(\mu_0,\Sigma_0)}(Y = 0 \mid \mathbf{x}) = p_{\theta_1=(\mu_1,\Sigma_1)}(Y = 1 \mid \mathbf{x})$$

As a reminder, the PDF for a d-dimensional multivariate Gaussian is:

$$f(\mathbf{x} \in \mathbb{R}^d) = \frac{1}{\sqrt{(2\pi)^d |\mathbf{\Sigma}|}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)\right\}.$$

Solution:

$$\begin{aligned}
p_{\theta_0=(\mu_0,\Sigma_0)}(Y=0 \mid \mathbf{x}) &= p_{\theta_1=(\mu_1,\Sigma_1)}(Y=1 \mid \mathbf{x}) \\
p_\theta(\mathbf{x} \mid Y=0)p_\theta(Y=0) &= p_\theta(\mathbf{x} \mid Y=1)p_\theta(Y=1) \quad \text{(the } p(\mathbf{x}) \text{ cancels out)} \\
\log(p_\theta(\mathbf{x} \mid Y=0)p_\theta(Y=0)) &= \log(p_\theta(\mathbf{x} \mid Y=1)p_\theta(Y=1)) \\
-\log\sqrt{(2\pi)^d|\Sigma_0|} - \frac{1}{2}(\mathbf{x}-\mu_0)^\top\Sigma_0^{-1}(\mathbf{x}-\mu_0) + \log(1-\pi_1) & \\
= -\log\sqrt{(2\pi)^d|\Sigma_1|} - \frac{1}{2}(\mathbf{x}-\mu_1)^\top\Sigma_1^{-1}(\mathbf{x}-\mu_1) + \log\pi_1 &
\end{aligned}$$

$$\begin{aligned}
-\frac{1}{2}((\mathbf{x}-\mu_1)^\top\Sigma_1^{-1}(\mathbf{x}-\mu_1) - (\mathbf{x}-\mu_0)^\top\Sigma_0^{-1}(\mathbf{x}-\mu_0)) & \\
-\frac{1}{2}(\log|\Sigma_1| - \log|\Sigma_0|) + \log\frac{\pi_1}{1-\pi_1} &= 0
\end{aligned}$$

This defines a quadratic decision boundary in $\mathbf{x}$ (see the $\mathbf{x}^\top\Sigma\mathbf{x}$ quadratic terms).

Note: We take the log in the 3rd line because it simplifies the expression to no longer contain exponentials.

(d) The algorithm developed in the previous parts is known as Quadratic Discriminant Analysis (QDA). In lecture, we primarily focused on a simpler variant called Linear Discriminant Analysis (LDA), where each class is assumed to have the *same* covariance matrix $\mathbf{\Sigma}$ rather than modeling many separate covariance matrices.

Assuming a shared covariance matrix $\mathbf{\Sigma}$, write the decision boundary as a linear function in the form $\mathbf{w}^\top\mathbf{x} + b$. Identify $\mathbf{w}$ and b.

$$p_{\theta_0=(\mu_0,\Sigma)}(Y=0 \mid \mathbf{x}) = p_{\theta_1=(\mu_1,\Sigma)}(Y=1 \mid \mathbf{x})$$

Solution:

We start from the solution to the previous part, and plug in the common covariance matrix:

$$\begin{aligned}
-\frac{1}{2}((\mathbf{x}-\mu_1)^\top\Sigma^{-1}(\mathbf{x}-\mu_1) - (\mathbf{x}-\mu_0)^\top\Sigma^{-1}(\mathbf{x}-\mu_0)) - \frac{1}{2}(\log|\Sigma| - \log|\Sigma|) + \log\frac{\pi_1}{1-\pi_1} &= 0 \\
-\frac{1}{2}(\mathbf{x}^\top\Sigma^{-1}\mathbf{x} - 2\mu_1^\top\Sigma^{-1}\mathbf{x} + \mu_1^\top\Sigma^{-1}\mu_1 - \mathbf{x}^\top\Sigma^{-1}\mathbf{x} + 2\mu_0^\top\Sigma^{-1}\mathbf{x} - \mu_0^\top\Sigma^{-1}\mu_0) + \log\frac{\pi_1}{1-\pi_1} &= 0 \\
-\frac{1}{2}(2(\mu_0-\mu_1)^\top\Sigma^{-1}\mathbf{x} + \mu_1^\top\Sigma^{-1}\mu_1 - \mu_0^\top\Sigma^{-1}\mu_0) + \log\frac{\pi_1}{1-\pi_1} &= 0
\end{aligned}$$

$$\underbrace{(\mu_1 - \mu_0)^\top \Sigma^{-1}}_{\mathbf{w}^\top} \mathbf{x} + \underbrace{\frac{1}{2}(\mu_0^\top \Sigma^{-1} \mu_0 - \mu_1^\top \Sigma^{-1} \mu_1) + \log \frac{\pi_1}{1-\pi_1}}_{b} = 0$$

(e) **When plugging in the fitted distributions for LDA, what is the posterior probability $p_\theta(Y = 1 \mid \mathbf{x})$?**

Hint: Recall the formula for the sigmoid function $s(z) = \frac{1}{1+\exp\{-z\}}$.

Hint: You should be able to reuse a lot of work from the previous part!

Solution:

$$\begin{aligned}
p_\theta(Y = 1 \mid \mathbf{x}) &= \frac{p_\theta(\mathbf{x} \mid Y = 1) p_\theta(Y = 1)}{p_\theta(\mathbf{x} \mid Y = 1) p_\theta(Y = 1) + p_\theta(\mathbf{x} \mid Y = 0) p_\theta(Y = 0)} \\
&= \frac{\exp\left\{-\frac{1}{2}(\mathbf{x} - \mu_1)^\top \Sigma^{-1}(\mathbf{x} - \mu_1) + \log \pi_1\right\}}{\exp\left\{-\frac{1}{2}(\mathbf{x} - \mu_1)^\top \Sigma^{-1}(\mathbf{x} - \mu_1) + \log \pi_1\right\} + \exp\left\{-\frac{1}{2}(\mathbf{x} - \mu_0)^\top \Sigma^{-1}(\mathbf{x} - \mu_0) + \log(1 - \pi_1)\right\}} \\
&= \frac{1}{1 + \exp\left\{-\frac{1}{2}(\mathbf{x} - \mu_0)^\top \Sigma^{-1}(\mathbf{x} - \mu_0) + \log(1 - \pi_1) - (-\frac{1}{2}(\mathbf{x} - \mu_1)^\top \Sigma^{-1}(\mathbf{x} - \mu_1) + \log \pi_1)\right\}} \\
&= \frac{1}{1 + \exp\{\mathbf{w}^\top \mathbf{x} + b\}} = s(-(\mathbf{w}^\top \mathbf{x} + b))
\end{aligned}$$

$$\implies p_\theta(Y = 0 \mid \mathbf{x}) = 1 - s(-(\mathbf{w}^\top \mathbf{x} + b))$$

The expression in the exponential in the 3rd line is the exact same thing as the linear function found in the previous part.

This highlights the fact that assuming a Gaussian data generating distribution results in a logistic posterior, where the input to the logistic function is a linear function in $\mathbf{x}$. This is exactly the model assumed by logistic regression! The difference is that in GDA, we fit Gaussians to the data to find μ_i, Σ_i, π_i. In logistic regression, we directly optimize to find the $\mathbf{w}, b$ weights that define the posterior.

(f) The Bayes optimal classifier (given a symmetric loss function) is one that classifies any point $\mathbf{x}$ as the class with maximum posterior probability $p(Y = C_i \mid \mathbf{x})$:

$$\text{Classifier}_{\text{Bayes}}(\mathbf{x}) = \arg\max_{C_1, \ldots, C_k} p(Y = C_i \mid \mathbf{x})$$

Why is the decision boundary found by LDA or QDA not generally the Bayes optimal decision boundary?

Hint: What assumptions did we make in GDA?

Solution: The reason why we don't recover the Bayes optimal classifier is because we are only approximating the posterior with a Gaussian class-conditional data distribution. We can never find the true posterior, unless we know exactly how the data was generated.

So, the next best thing we can do is to minimize classification error, assuming that our fitted Gaussians are similar to reality.