Classification: Discriminant Analysis

-Introduction to Data Science-

ISL: Chapters 4.4

Lecturer: Darren Homrighausen, PhD

Classification

Logistic regression, which is the main type of GLM we have considered, directly models

$$\pi(X) = \mathbb{P}(Y = 1|X)$$

using the logistic function.

There is an alternative approach that models the distribution of the $X's$ directly and then inverts the probability via Bayes' theorem.

(Note: there is no direct relationship between the Bayes' rule and Bayes' theorem. They are just named after the same person)

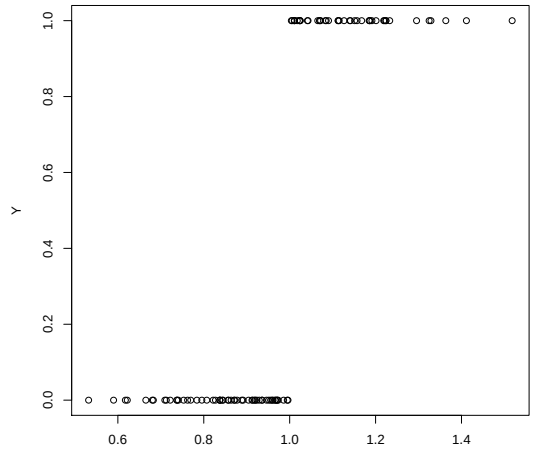
Why Would we Want to do That?

There are several drawbacks to logistic regression:

- If the classes are well-separated, logistic regression is unstable (or undefined)
- It is awkward to use when the supervisor has multiple levels

  (However it is still possible. This is called multinomial logistic (if supervisor is nominal) and proportional odds logistic regression (if the supervisor is ordinal))

Example of Well Separated Classes:



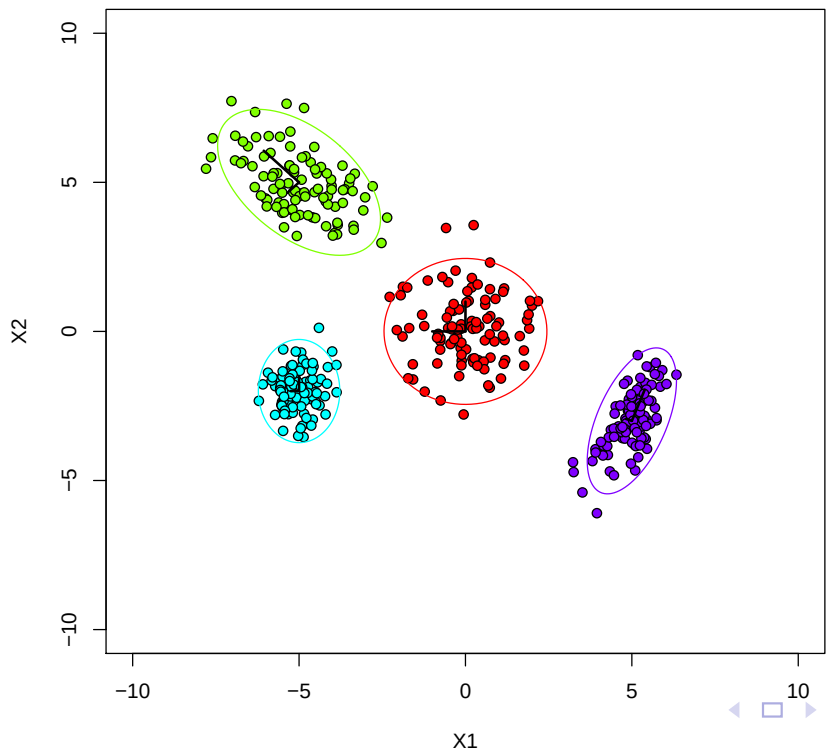

```
> glm(Y~X,family='binomial')
(Intercept)            X
     -986.2         974.2
Degrees of Freedom: 99 Total (i.e. Null);  98 Residual
Null Deviance:      138.3
Residual Deviance: 1.989e-08  AIC: 4
Warning messages:
1: glm.fit: algorithm did not converge
2: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

What is a Gaussian?

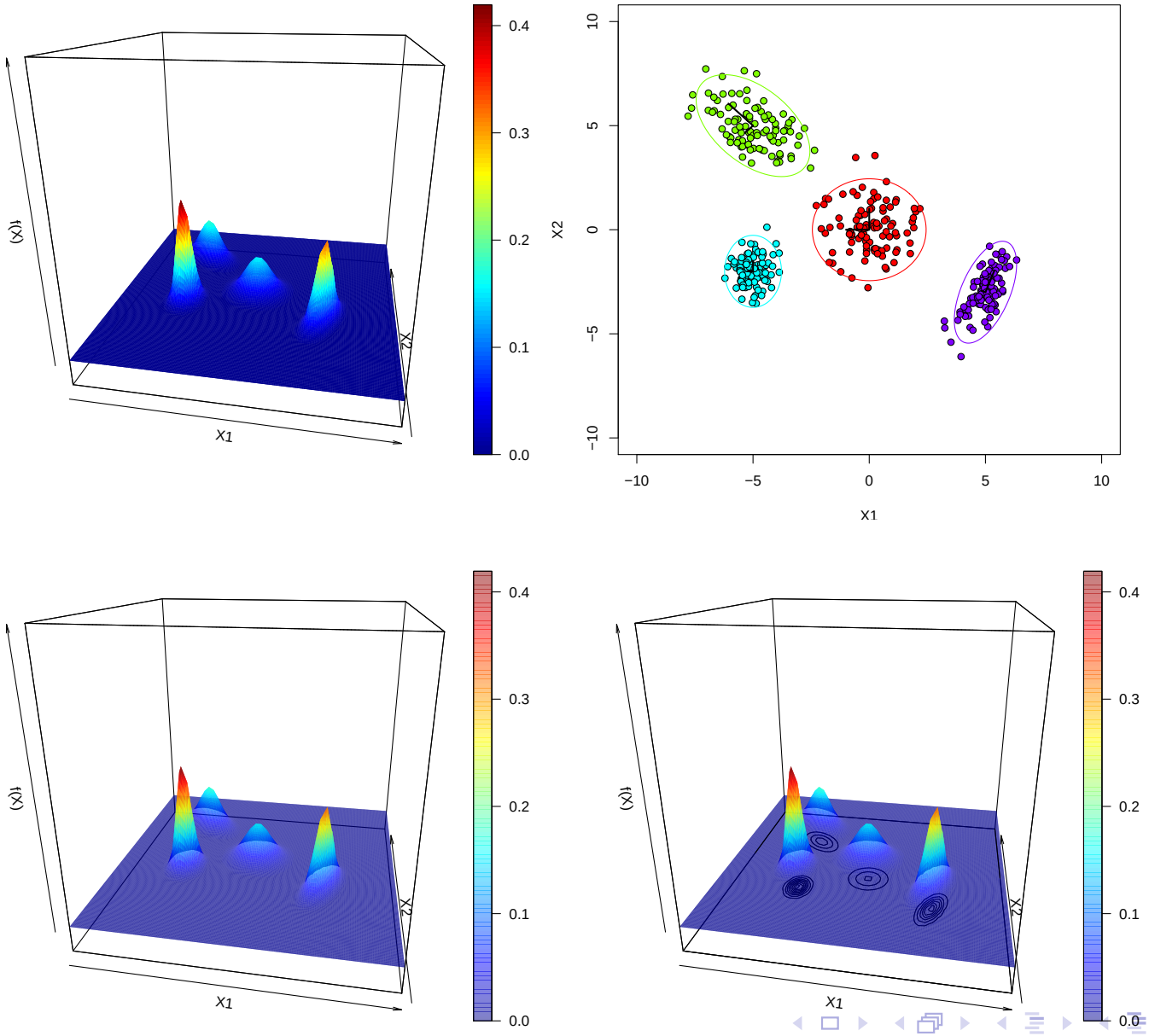
Suppose

$$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim N\left(\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \Sigma = \begin{bmatrix} \mathrm{var}(X_1) & \mathrm{cov}(X_1, X_2) \\ \mathrm{cov}(X_2, X_1) & \mathrm{var}(X_2) \end{bmatrix}\right)$$

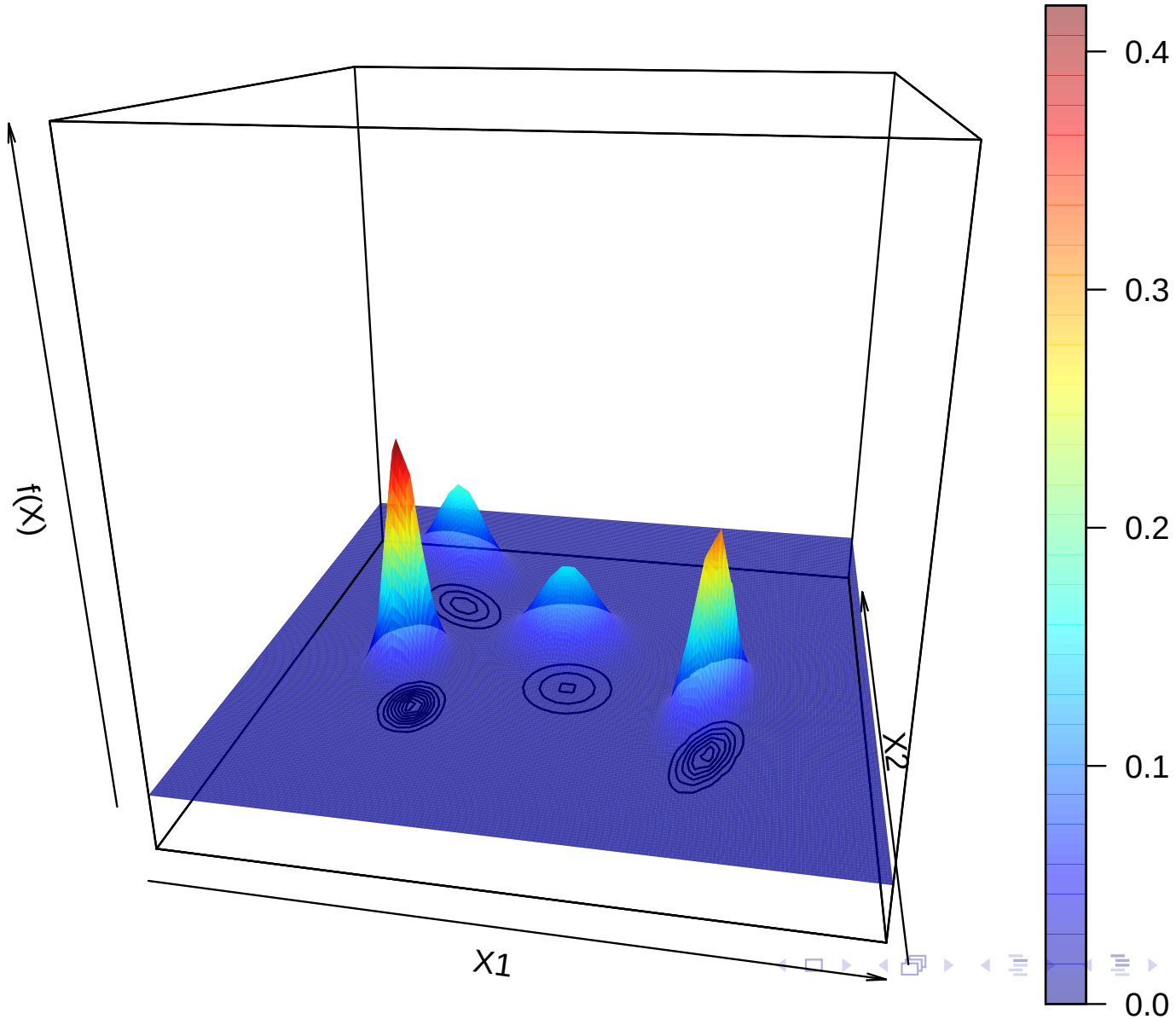
Here are $n = 100$ draws from four different Gaussian distributions.



What is a Gaussian?



What is a Gaussian?



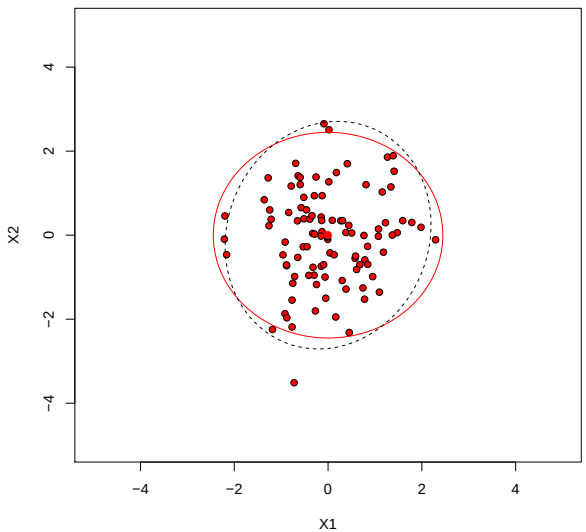
Estimate μ and Σ?

Suppose we make $n = 100$ independent observations

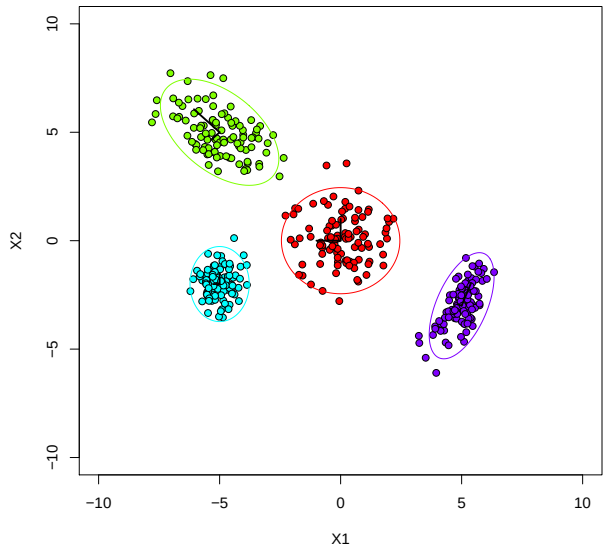
$$X_1, \ldots, X_{100} \sim N\left(\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i = \begin{bmatrix} 0.0012 \\ 0.001 \end{bmatrix}$$

$$\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X})(X_i - \overline{X})^\top = \frac{1}{n}(\mathbb{X} - \overline{\mathbb{X}})^\top(\mathbb{X} - \overline{\mathbb{X}}) = \begin{bmatrix} 0.8 & 0.1 \\ 0.1 & 1.2 \end{bmatrix}$$



Estimating μ and Σ with several Gaussians



Suppose we want to estimate different Gaussians at the same time

Let $g = 1, \ldots, G$ index these groups

($G = 4$ in figure)

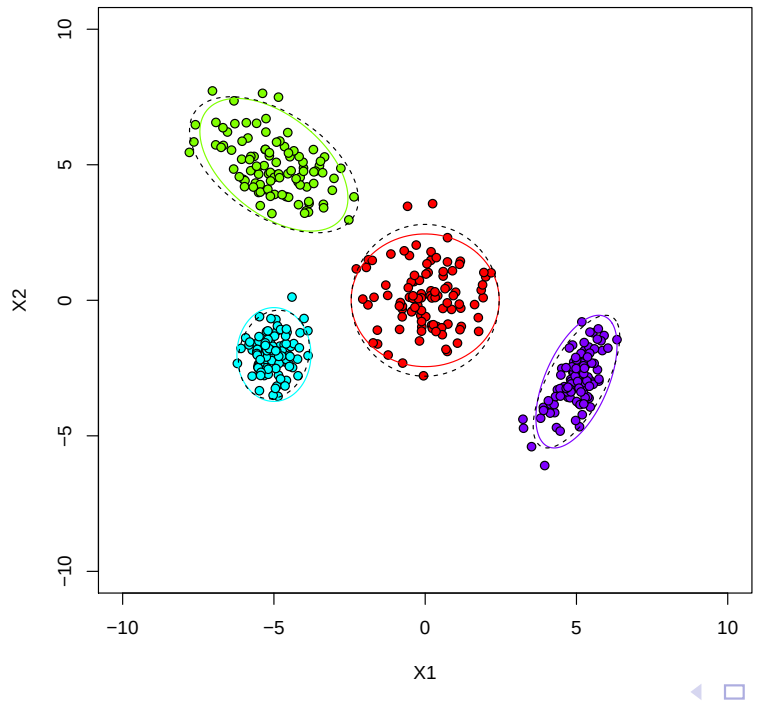
- $X_1^g, \ldots, X_{n_g}^g$ be from group g
- n_g be the number of observations in the g^{th} group
- $n = \sum_{g=1}^{G} n_g$

Estimating several different Gaussians

We can estimate these groups with

$$\overline{X}_g = \frac{1}{n_g} \sum_{i=1}^{n_g} X_i^g$$

$$\hat{\Sigma}_g = \frac{1}{n_g} \sum_{i=1}^{n_g} (X_i^g - \overline{X}_g)(X_i^g - \overline{X}_g)^\top$$



Estimating several different Gaussians

A problem with this approach: a lot of parameters

Each covariance matrix has: $p(p+1)/2$ parameters
(As $\hat{\Sigma}_g$ must be symmetric)

For G groups, this means $Gp(p+1)/2$ parameters

For this problem:

$$Gp(p+1)/2 = 12$$

This can be very large for even moderately large p or G

For $p = 50$:

$$Gp(p+1)/2 = 5100$$

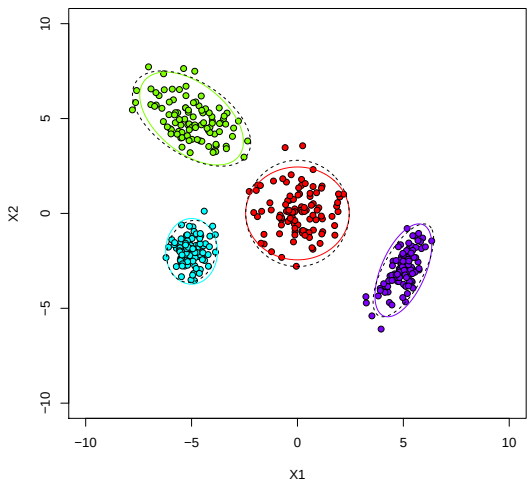
How to estimate μ and Σ with a mixture of Gaussians

There isn't much we can do about the $p(p+1)/2$ part

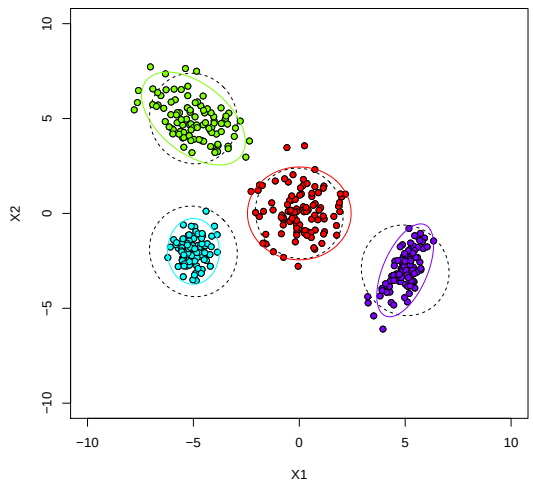
But, we can make this simplification: Assume $\Sigma_g = \Sigma$

(This means we use all observations to estimate a single covariance)

$$\hat{\Sigma} = \frac{1}{n}\sum_{g=1}^{G}\sum_{i=1}^{n_g}(X_i^g - \overline{X}_g)(X_i^g - \overline{X}_g)^\top$$



Different $\hat{\Sigma}_g$



All same $\hat{\Sigma}$

Linear Discriminant Analysis

Linear Discriminant Analysis (LDA)

Suppose our supervisor can take on G different levels:

$$Y = \begin{cases} 1 \\ \vdots \\ G \end{cases}$$

1. We model the features as a Gaussian random variable
   $(X|Y = g \sim N(\mu_g, \Sigma))$
2. Specify unconditional probabilities that $Y = g$
   $(\pi_g = \mathbb{P}(Y = g))$
3. Turn this into a conditional distribution of Y given X
   (Using Bayes' theorem)
4. Find the best possible classifier
   (This is the Bayes' rule)
5. This depends on the unknown parameters
   $\pi_1, \ldots, \pi_G, \mu_1, \ldots, \mu_G, \Sigma$
6. Estimate these parameters with their sample versions

What is Bayes' theorem?

Here, we are interested in the class label $Y = g$ at particular feature value X

That is, we want

$$\mathbb{P}(Y = g|X)$$

(Recall, this is the main ingredient to the Bayes' rule)

Bayes' theorem:

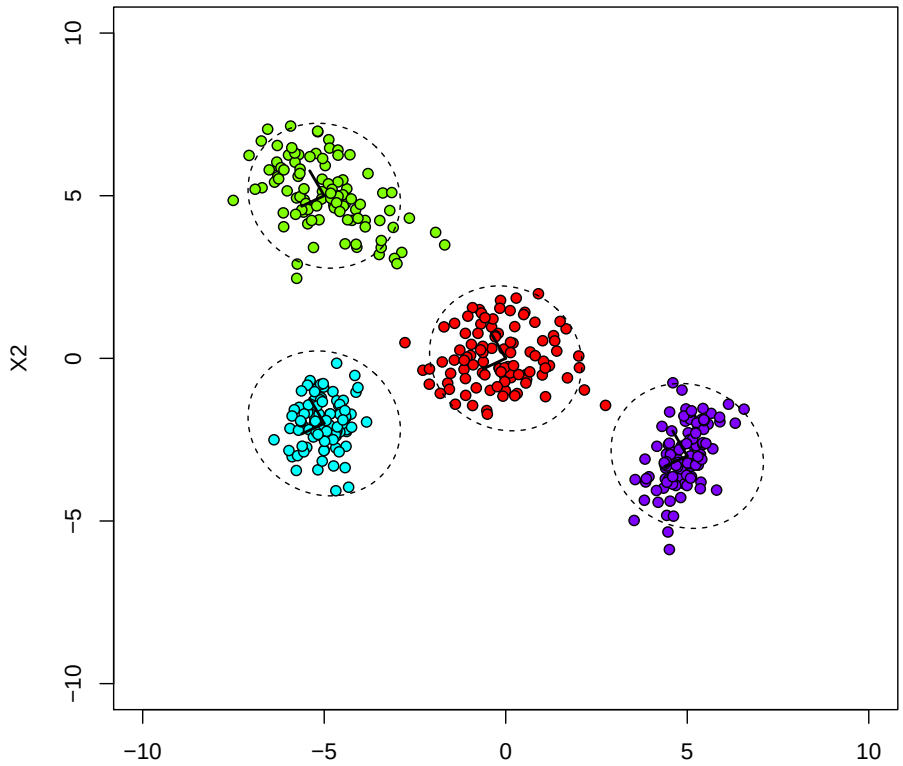
$$\mathbb{P}(Y = g|X) = \frac{\mathbb{P}(X|Y = g)\mathbb{P}(Y = g)}{\mathbb{P}(X)}$$

- $\mathbb{P}(X|Y = g) = N(\mu_g, \Sigma)$
  (This is usually referred to as the likelihood)
- $\mathbb{P}(Y = g) = \pi_g$
  (This is usually referred to as the prior)

Building intuition for LDA

Intuition

How would you classify a point with this data?

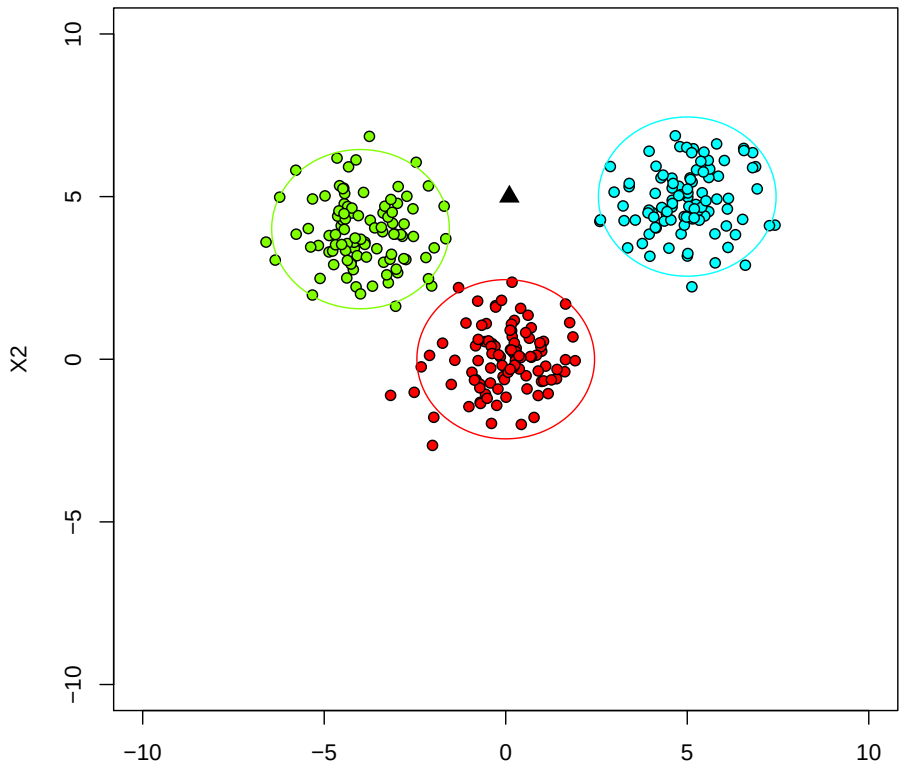


We could just classify an observation to the closest group

What do we mean by close? (Need to define distance)

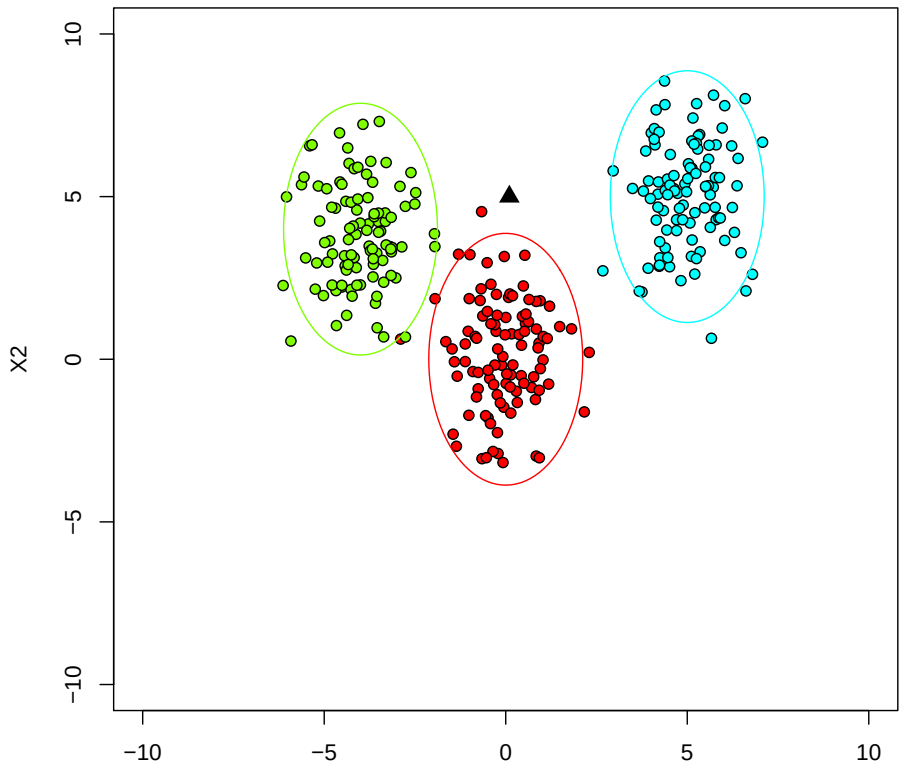
Intuition

What if the data looked like this?



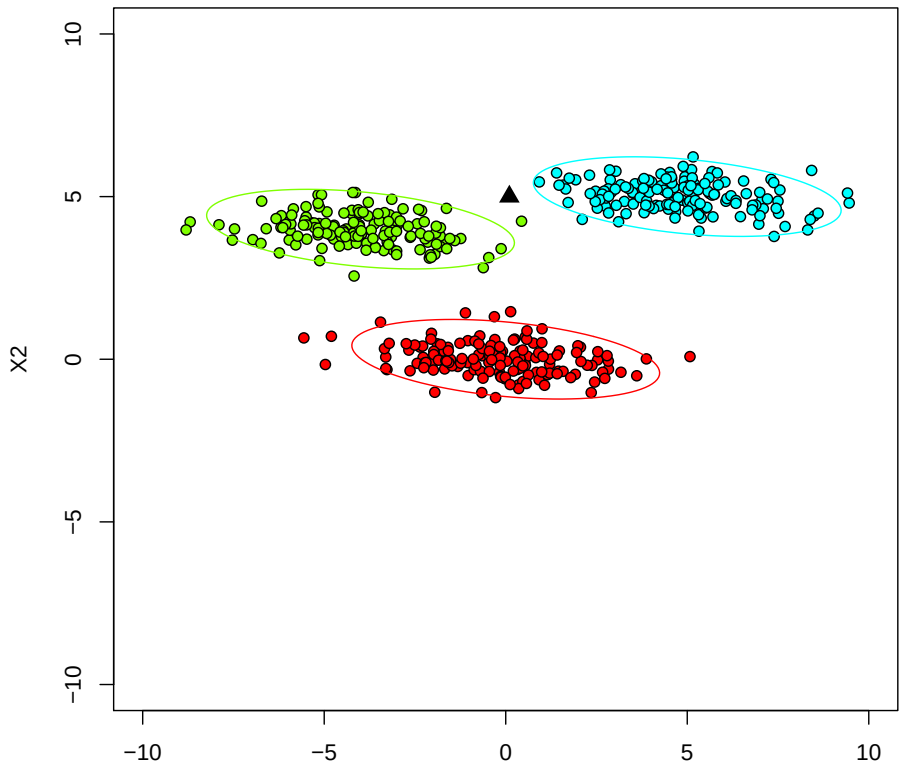
Intuition

Or this?



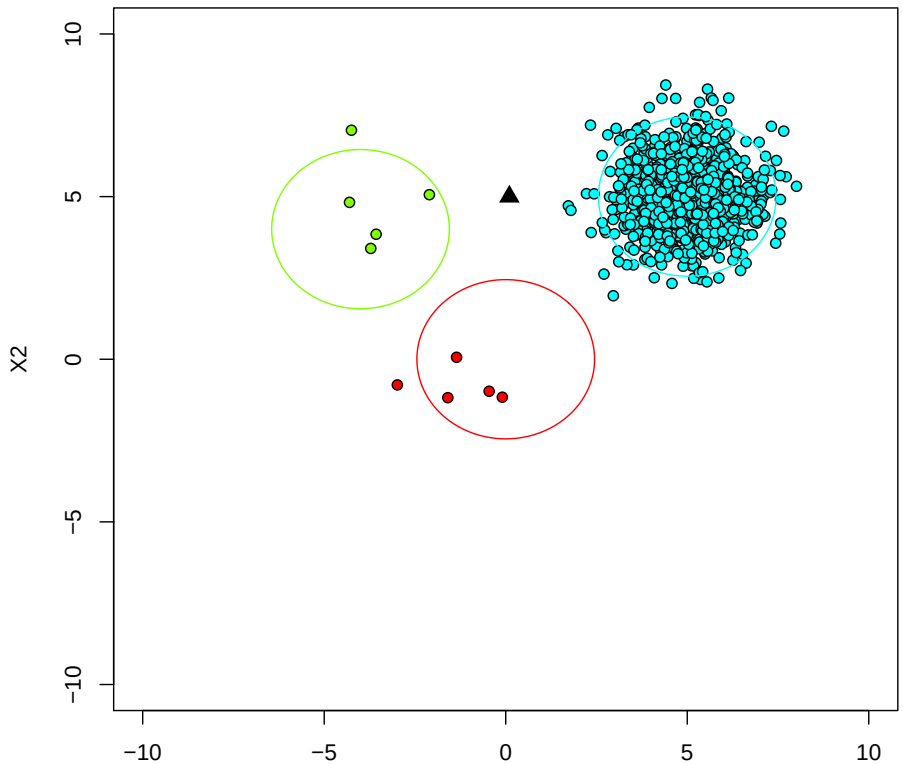
Intuition

How about this?



Intuition

What about now?



Intuition

All of these examples show that we need to take into account

- The shape of distribution (size and eccentricity of the ellipse)
- The relative number of points in each group

These are the two main ingredients in LDA

LDA

Linear discriminant analysis (LDA)

We use the linear discriminant function

$$\delta_g(X) = \underbrace{X^\top \hat{\Sigma}^{-1} \overline{X}_g - \frac{1}{2} \overline{X}_g^\top \hat{\Sigma}^{-1} \overline{X}_g}_{\textit{likelihood}} + \underbrace{\log(\hat{\pi}_g)}_{\textit{prior}}$$

Here, $\hat{\pi}_g$ is the fraction of observations in group g (that is, $\frac{n_g}{n}$)

We assign an observation to $\hat{g}$, where

$$\hat{g} = \arg\max_g \delta_g(x)$$

Linear discriminant analysis (LDA)

Intuitively, assigning observations to the nearest $\overline{X}_g$ (but ignoring the covariance) would amount to

$$\begin{aligned}\tilde{g} &= \underset{g}{\operatorname{argmin}} ||X - \overline{X}_g||_2^2 \\ &= \underset{g}{\operatorname{argmin}} X^\top X - 2X^\top \overline{X}_g + \overline{X}_g^\top \overline{X}_g \\ &= \underset{g}{\operatorname{argmin}} -X^\top \overline{X}_g + \frac{1}{2}\overline{X}_g^\top \overline{X}_g\end{aligned}$$

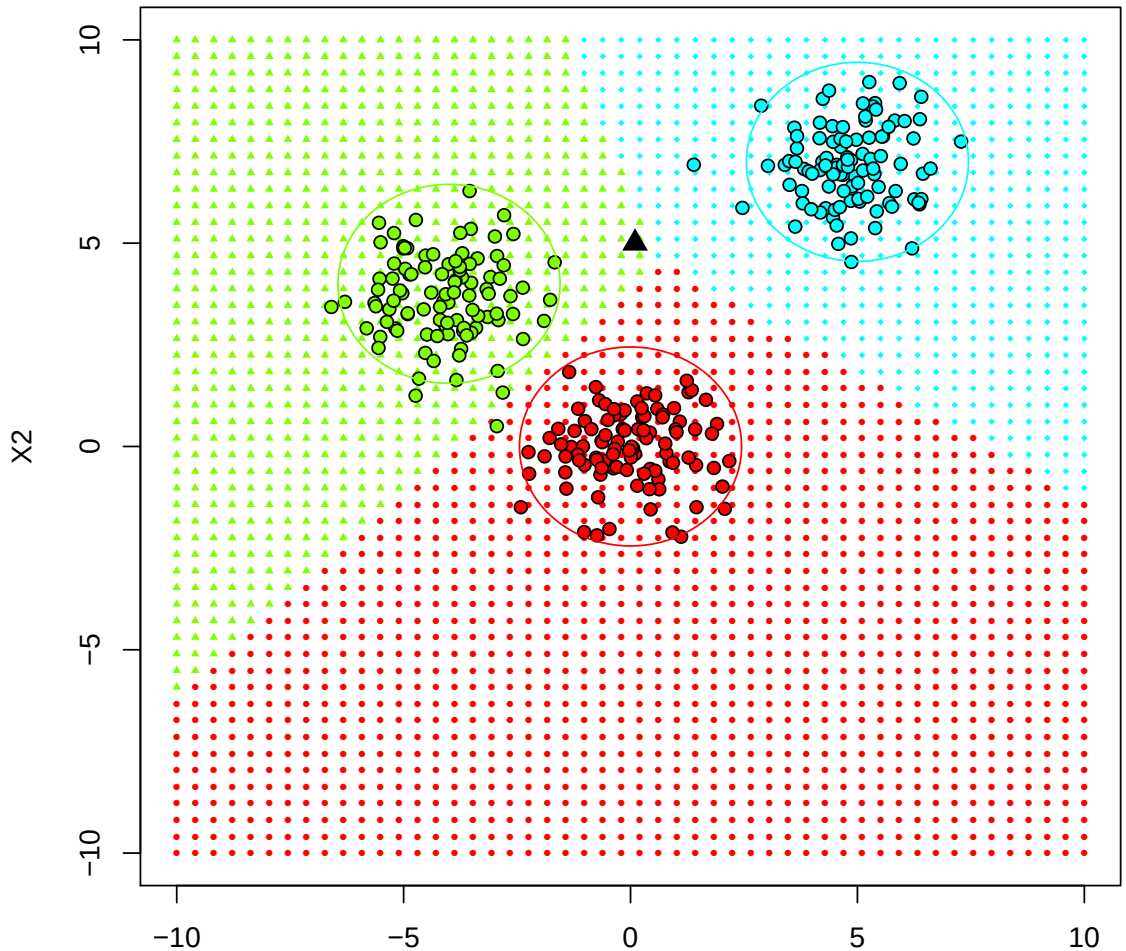
compare this to:

$$\hat{g} = \underset{g}{\arg\max} \underbrace{X^\top \hat{\Sigma}^{-1} \overline{X}_g - \frac{1}{2}\overline{X}_g^\top \hat{\Sigma}^{-1} \overline{X}_g}_{\textit{likelihood}} + \underbrace{\log(\hat{\pi}_g)}_{\textit{prior}}$$

The difference is we weight the distance by $\hat{\Sigma}^{-1}$ and weight the class assignment by fraction of observations in each class.

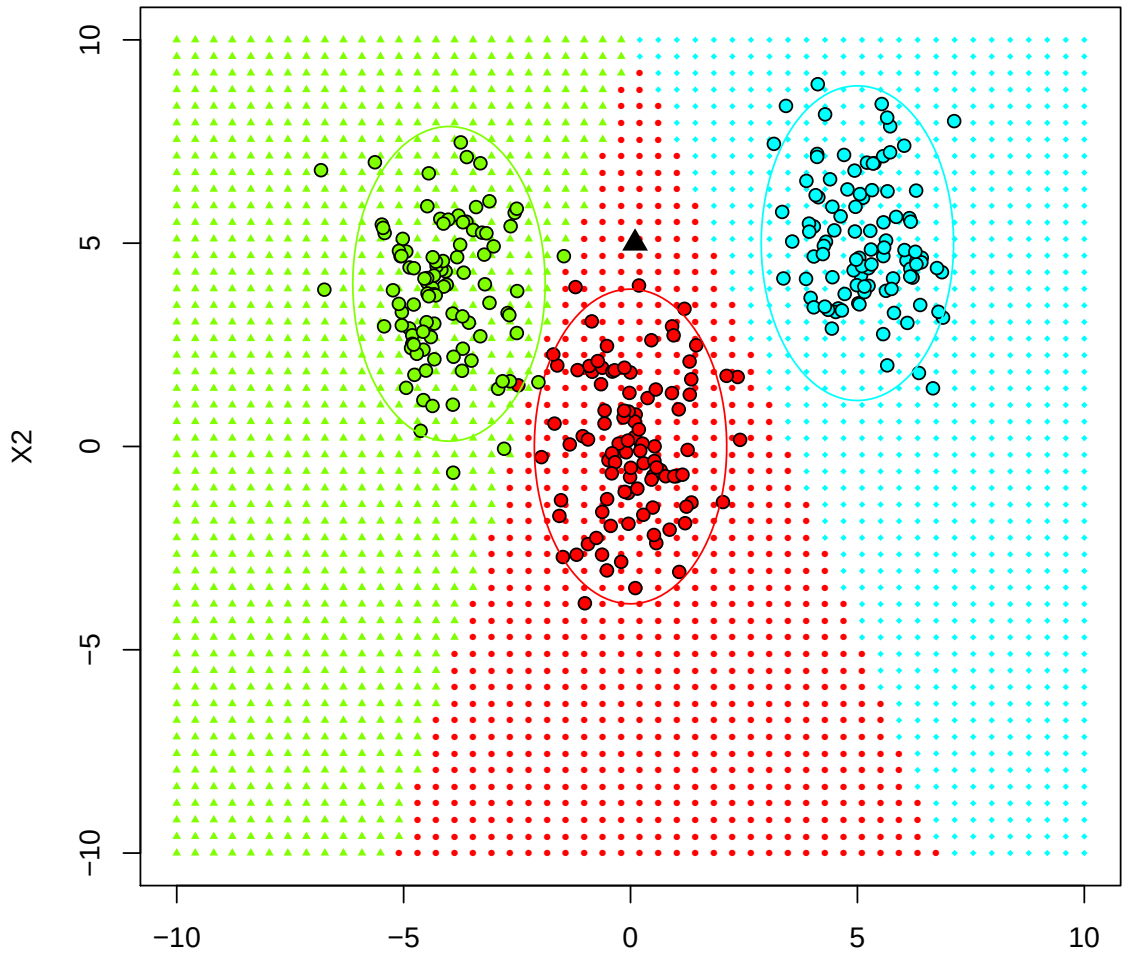
Intuition

What if the data looked like this?



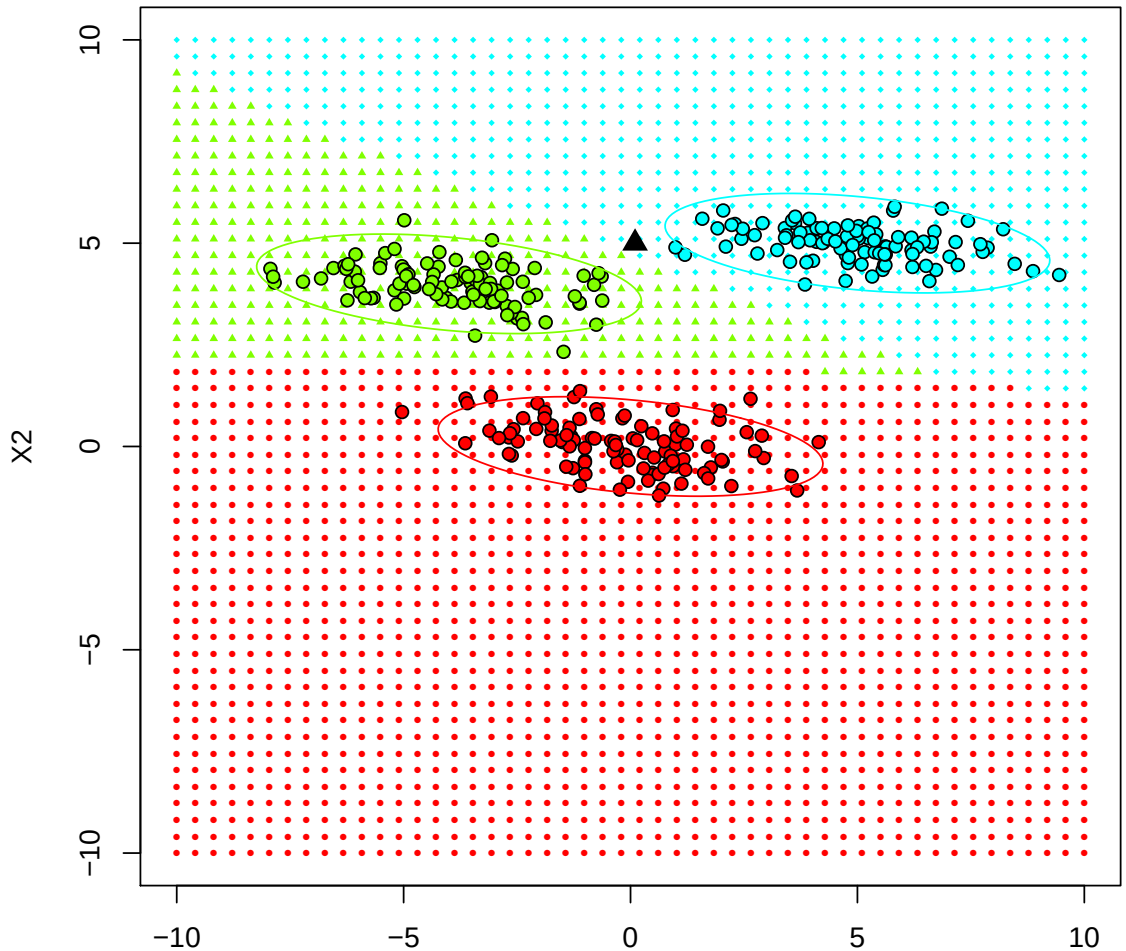
Intuition

Or this?



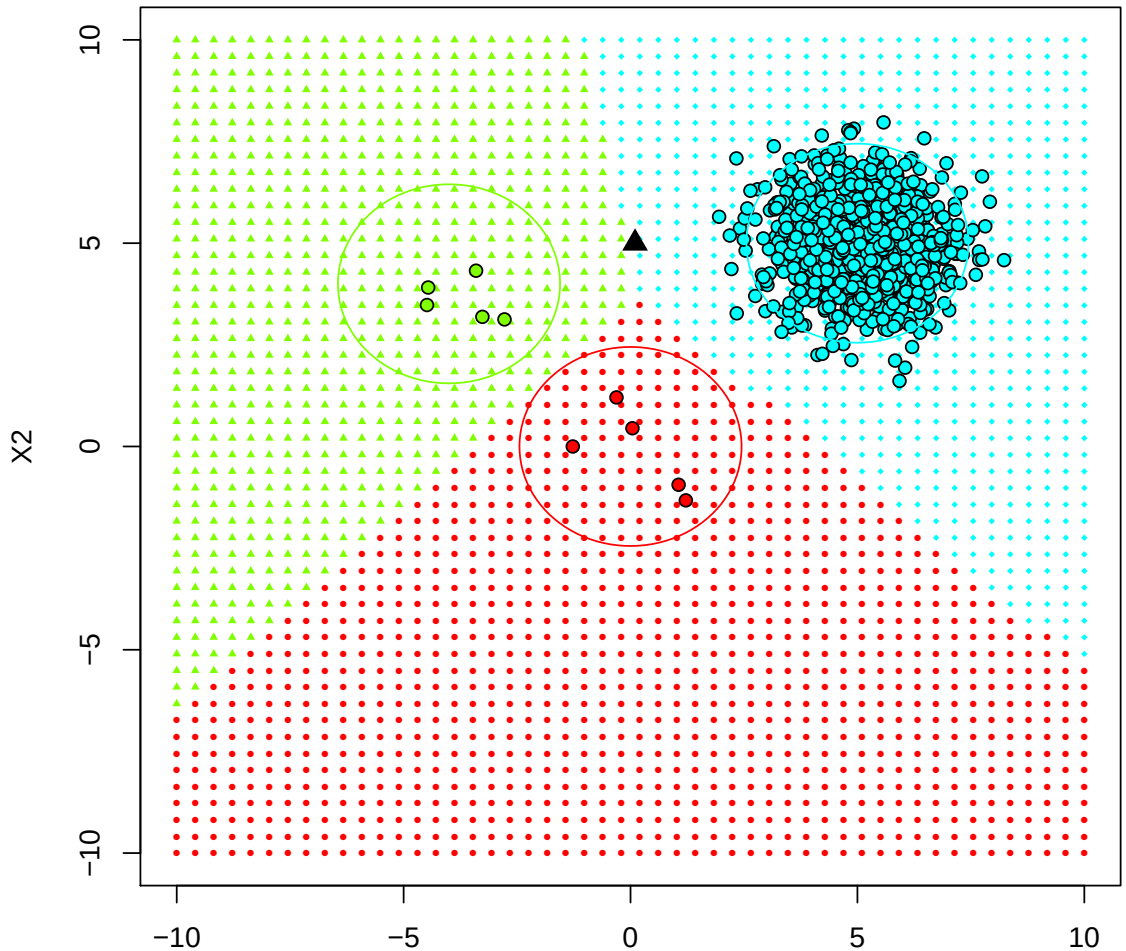
Intuition

How about this?



Intuition

What about now?



LDA in R

We can do this readily in R

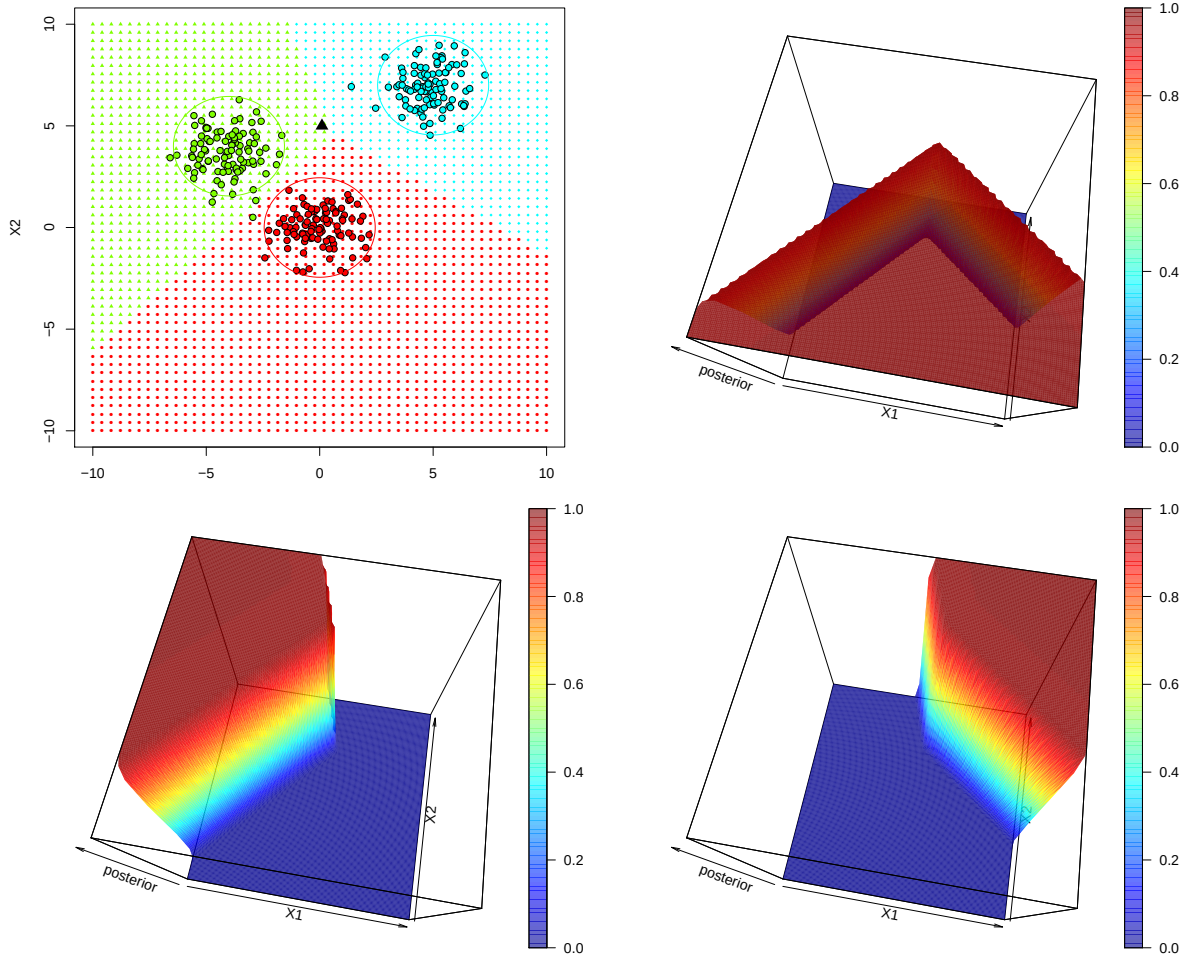
```
library(MASS)
lda.fit  = lda(Y~.,data=X)

> names(lda.fit)
 [1] "prior"   "counts"  "means"   "scaling" "lev"     "svd"

out = predict(lda.fit,X_0)

> out$posterior[1:3,]
          1            2            3
1 0.9999908 9.215567e-06 1.504633e-55
2 0.9999977 2.341924e-06 1.664446e-54
3 0.9999994 5.951430e-07 1.841223e-53
```

What does posterior mean?




```
> print(predict(lda.fit,X_0)$posterior)
          1         2           3
1 0.04883796 0.9477494 0.003412639
```

Recap

Reminder: For every problem, we can define:

$$g_*(X) = \arg\max_g \mathbb{P}(Y = g|X)$$

(That is, we want to maximize the conditional probability)

This is known as the Bayes' rule

Emphasis: The Bayes' rule is unknown/unknowable

With LDA we are trying to estimate it under particular assumptions

(Concept check: What are the assumptions?)

Performance of LDA

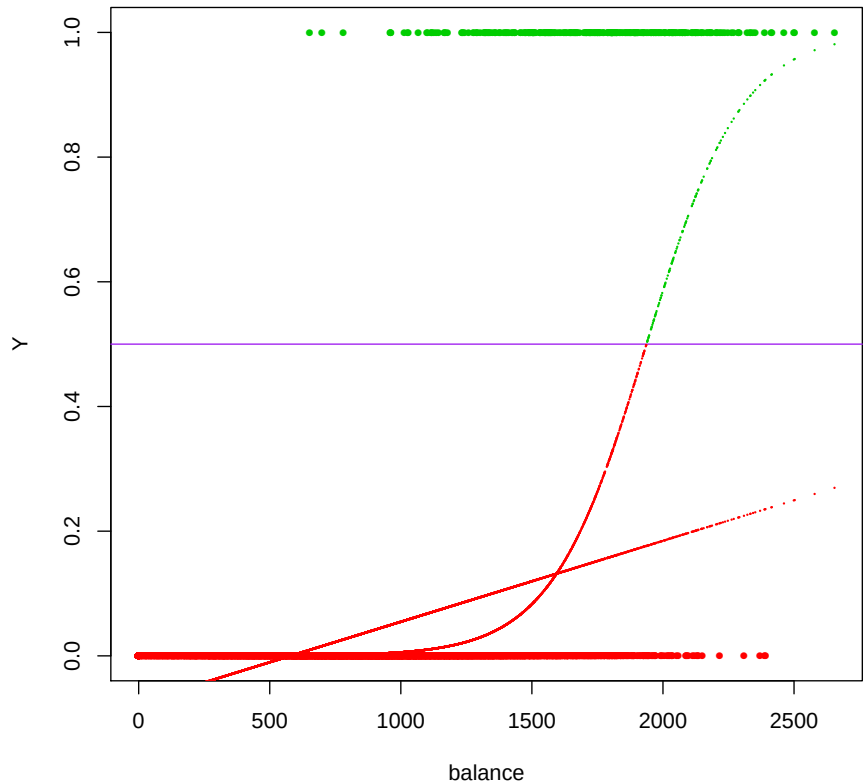
The quality of the classifier produced by LDA depends on:

- The sample size n
  (This determines how accurate the $\hat{\pi}_g$, $\hat{\mu}_g$, and $\hat{\Sigma}$ are)
- How wrong the LDA assumptions are
  (That is: $X|Y = g$ is a Gaussian with mean μ_g and variance Σ)

Recall: The decision boundary of a classifier are the values of X such that the classifier is indifferent between two (or more) levels of Y

A linear decision boundary is when this set of values looks like a line

We've Already Seen Other Examples of Linear Decision Boundaries



LDA: under correct assumptions

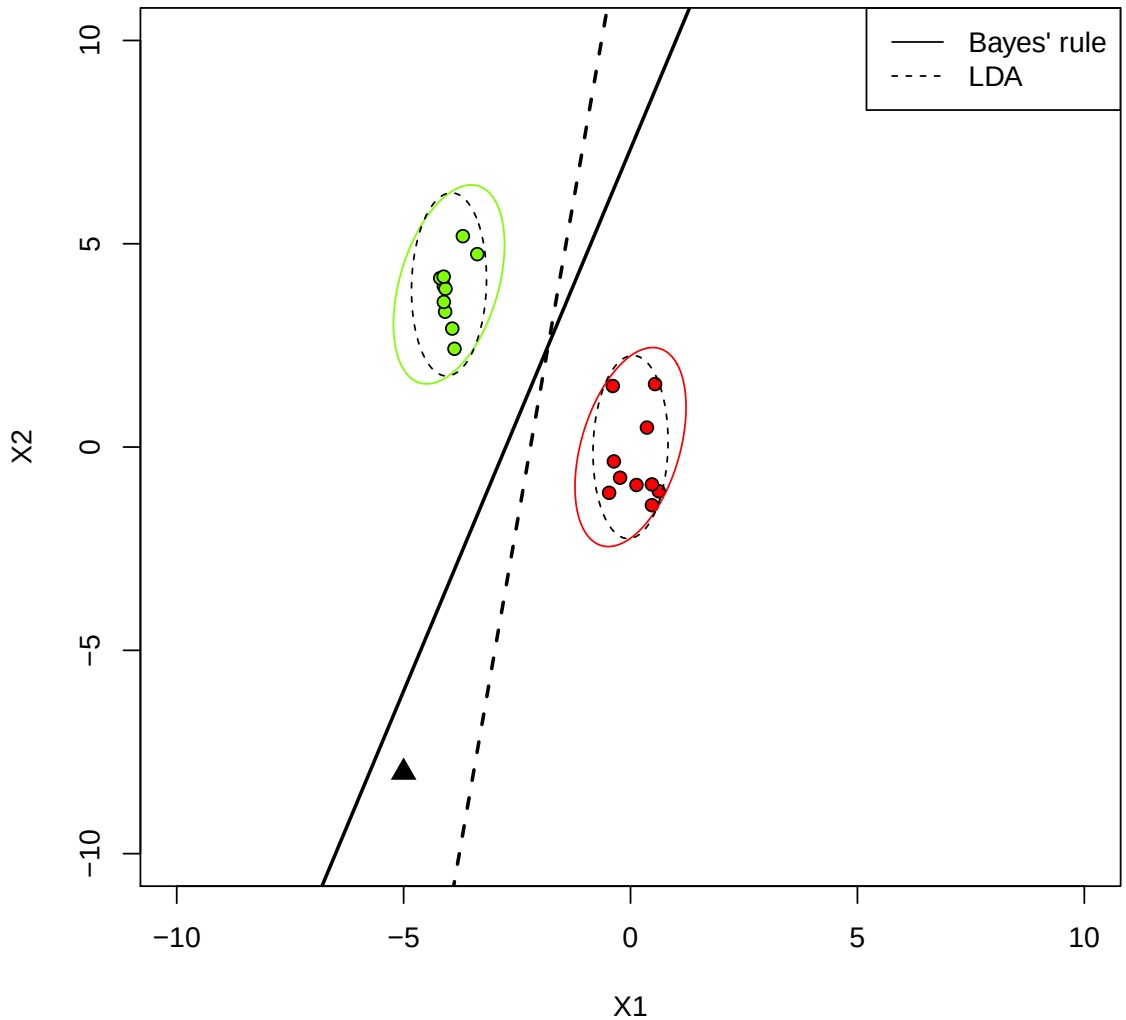


FIGURE: For $n_g = 10$

LDA: under correct assumptions

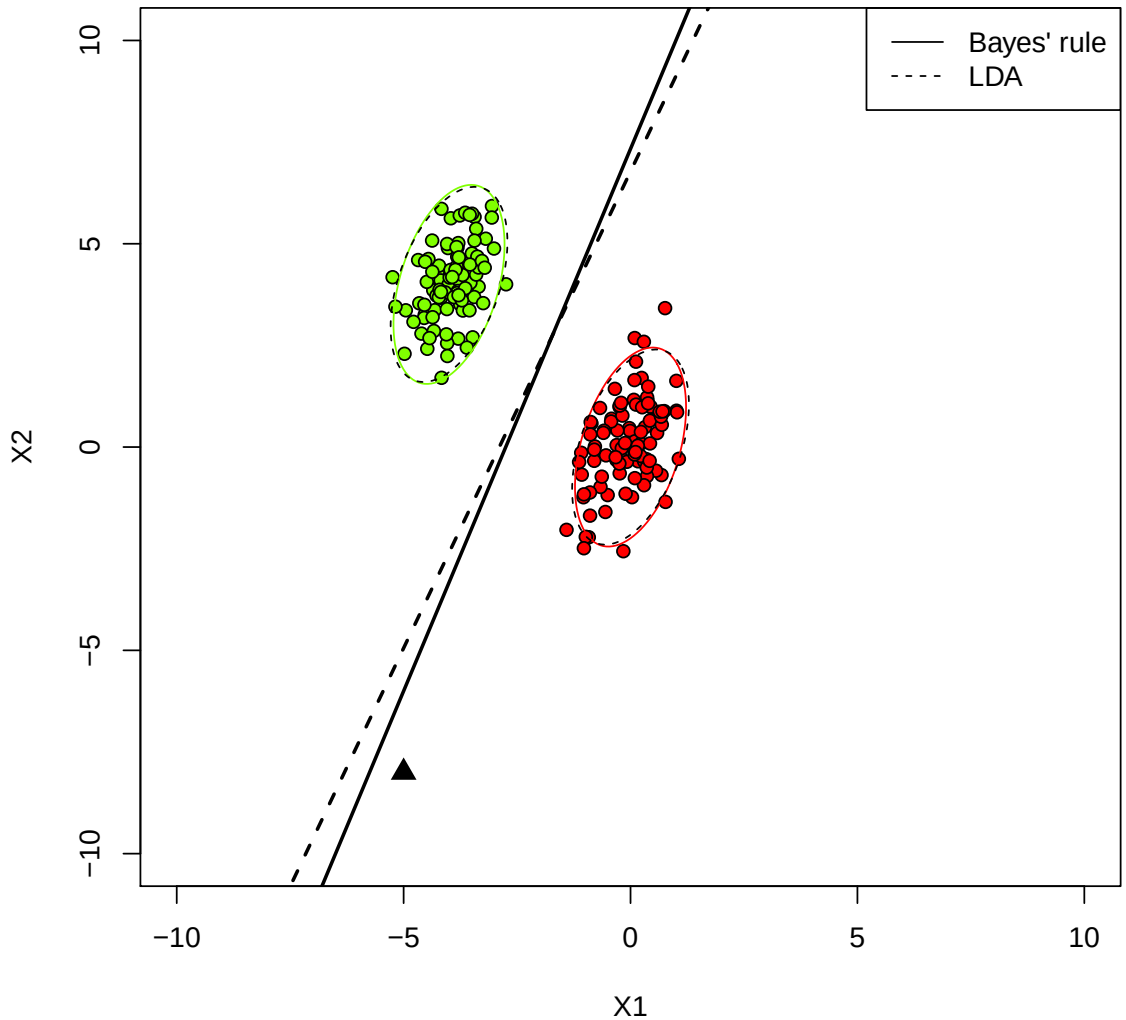


FIGURE: For $n_g = 100$

LDA: under correct assumptions

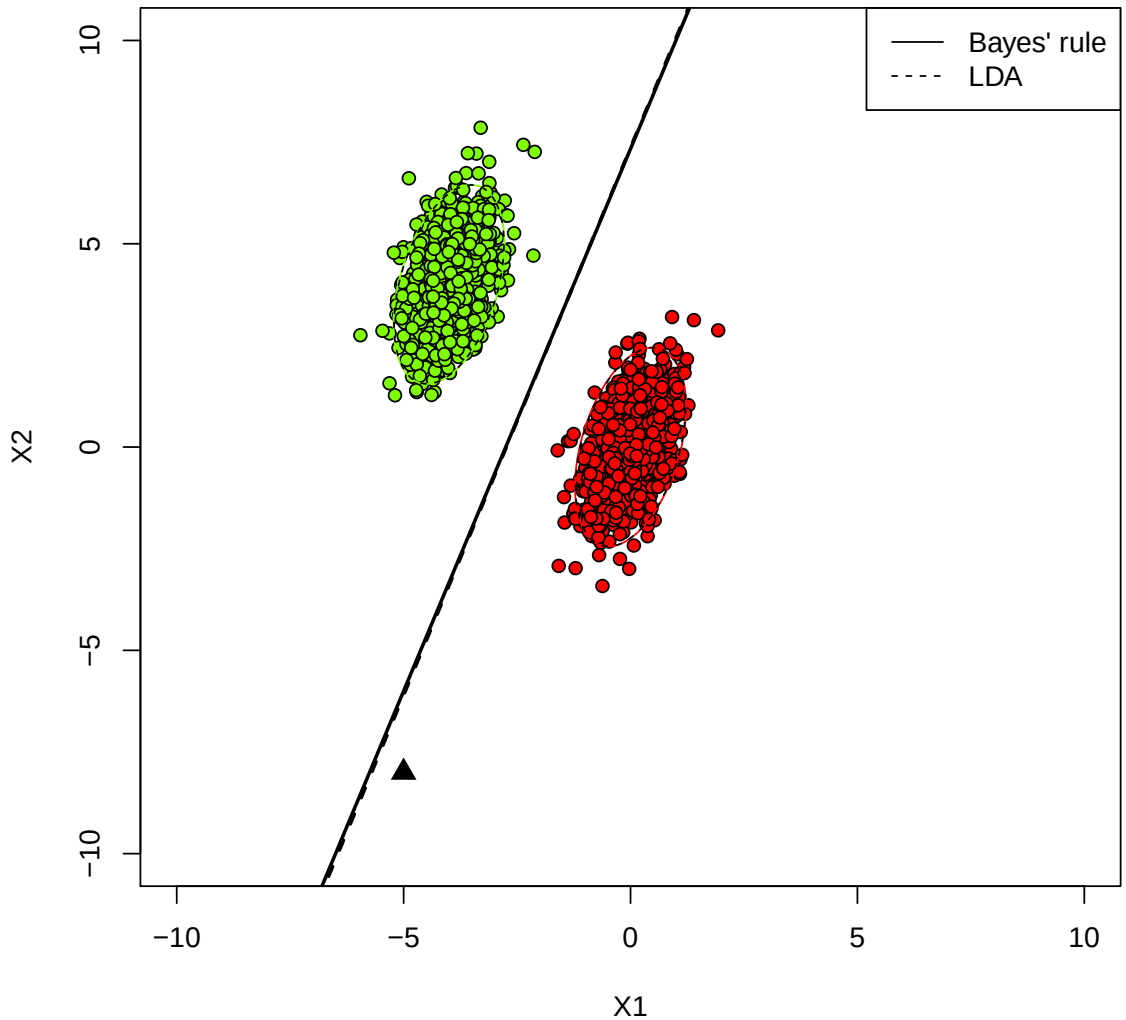


Figure: For $n_g = 1000$

LDA: mildly incorrect assumptions

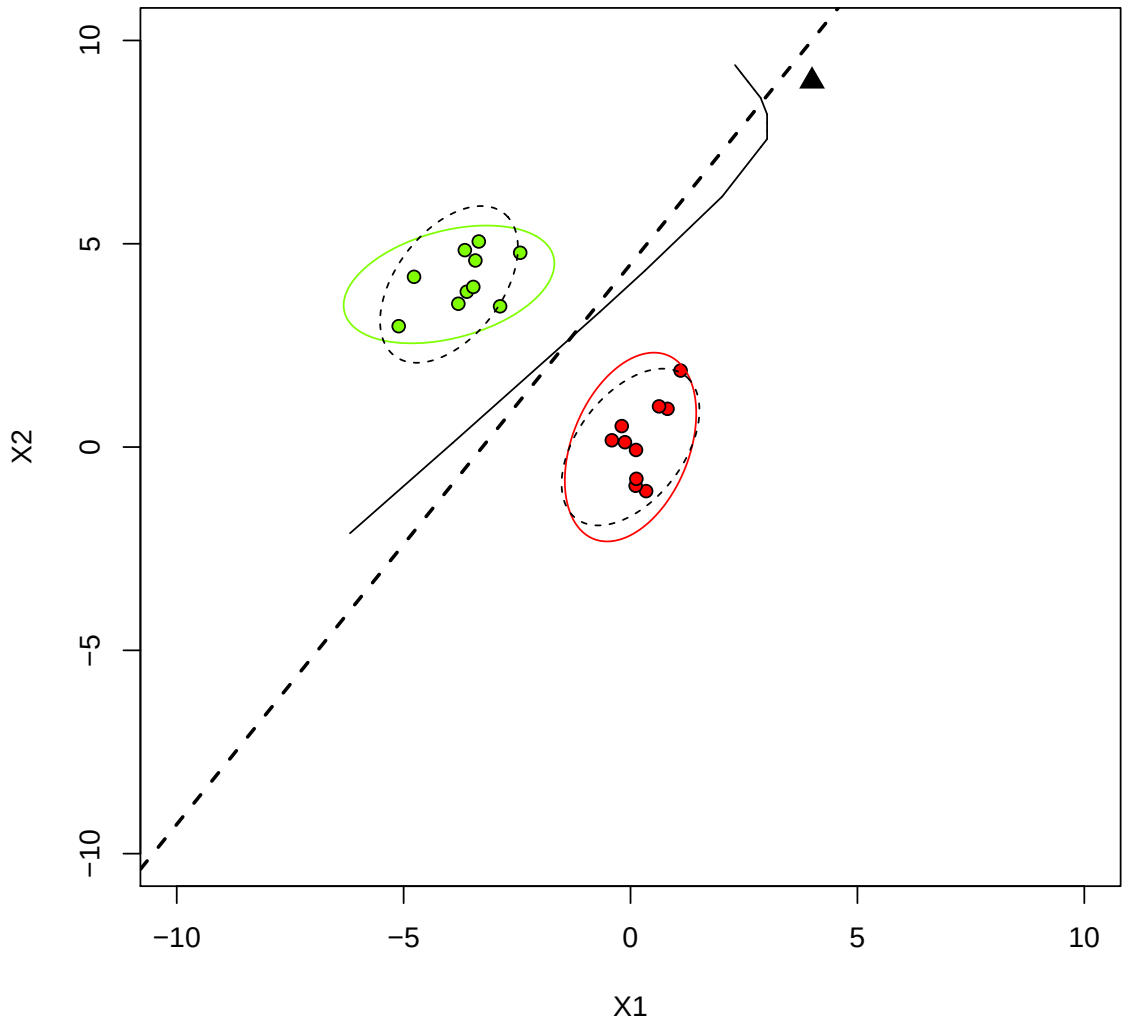


FIGURE: For $n_g = 10$

LDA: mildly incorrect assumptions

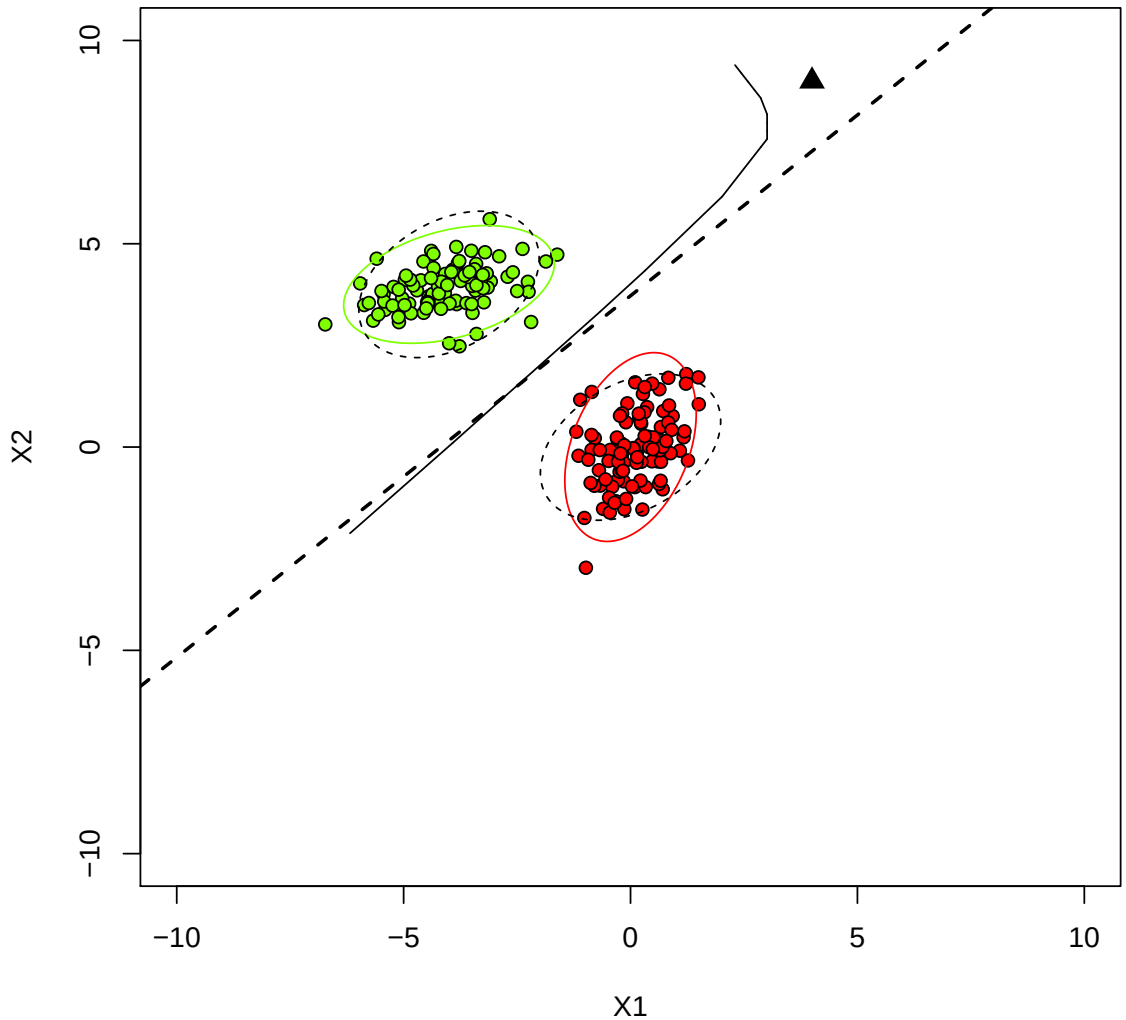


FIGURE: For $n_g = 100$

LDA: mildly incorrect assumptions

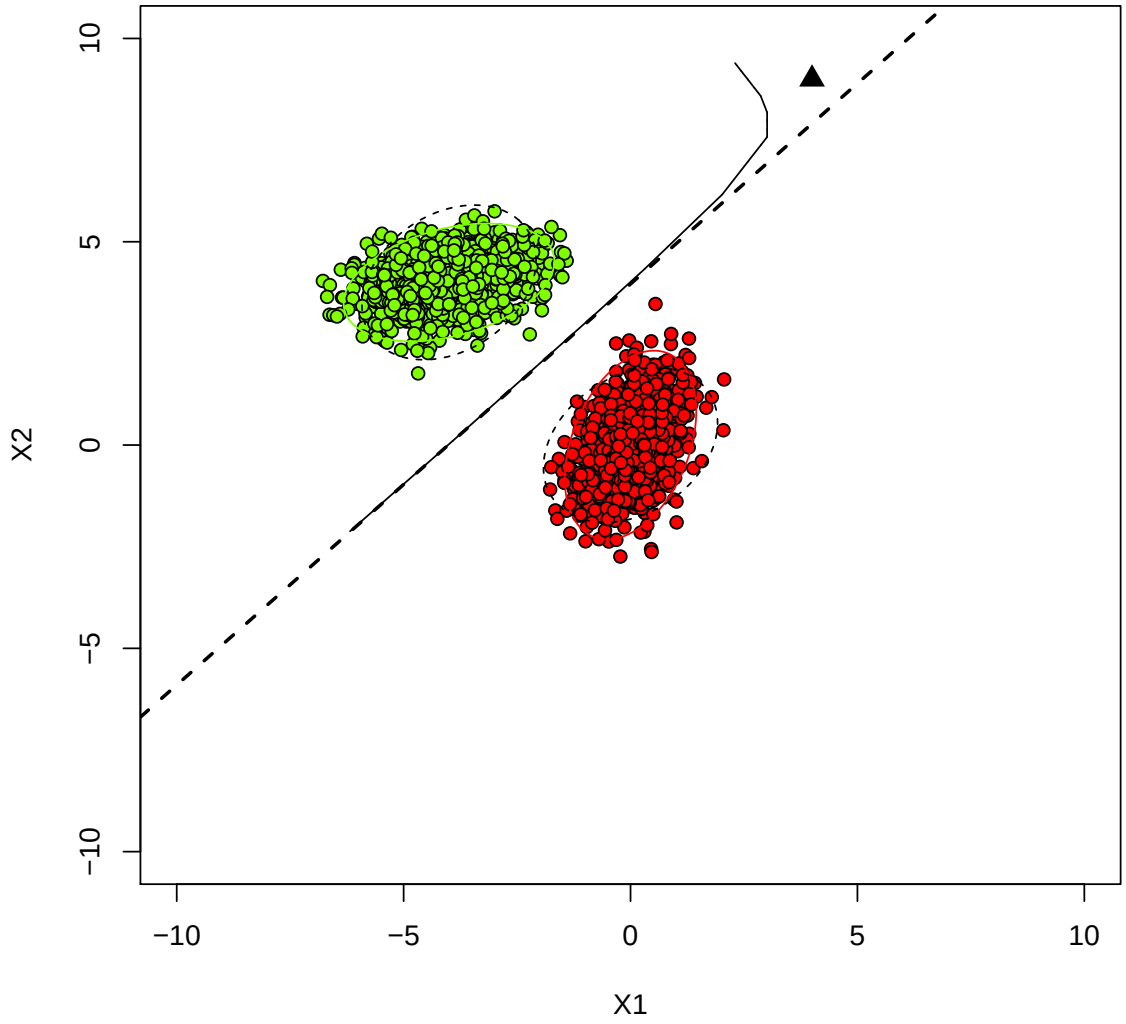


FIGURE: For $n_g = 1000$

LDA: very incorrect assumptions

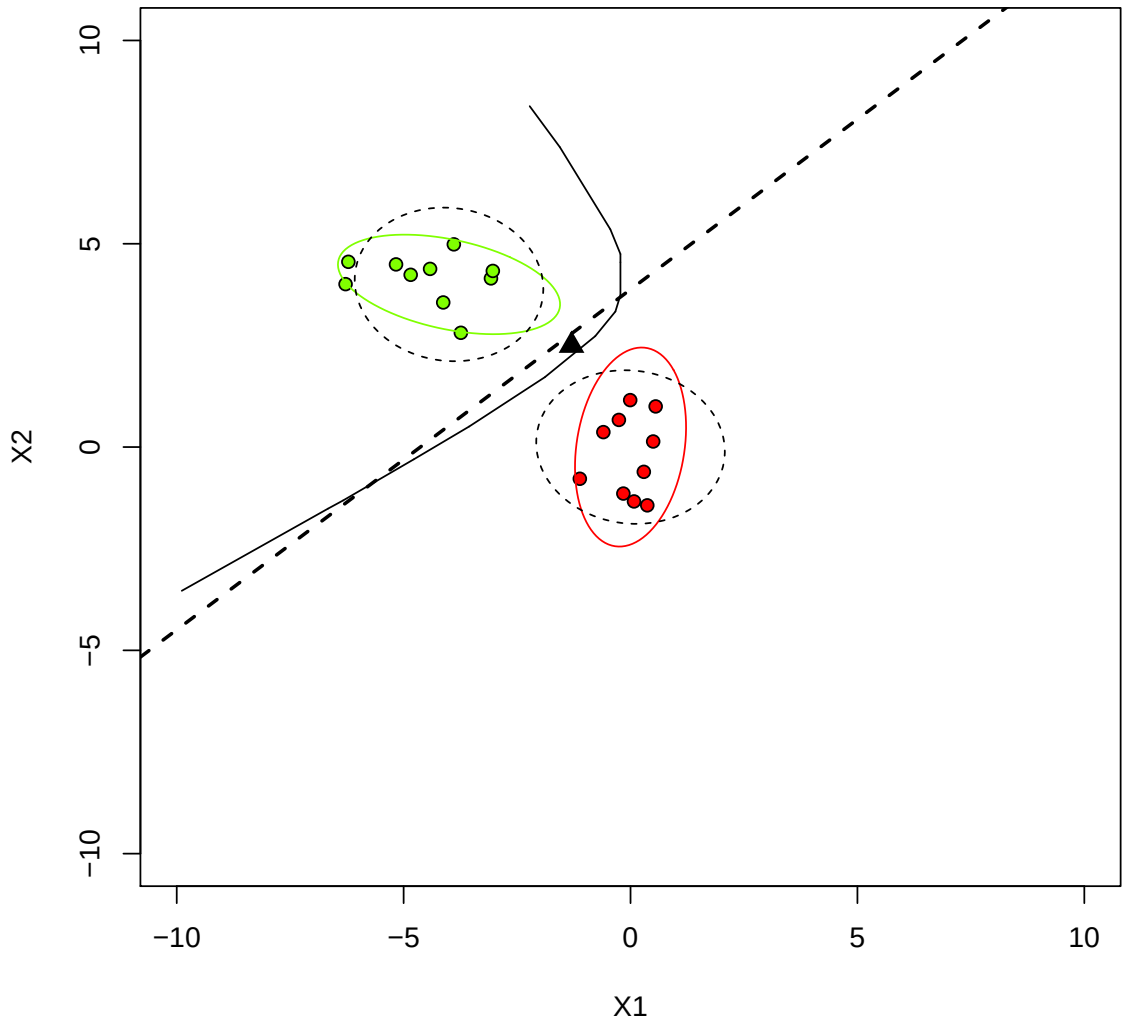


Figure: For $n_g = 10$

LDA: very incorrect assumptions

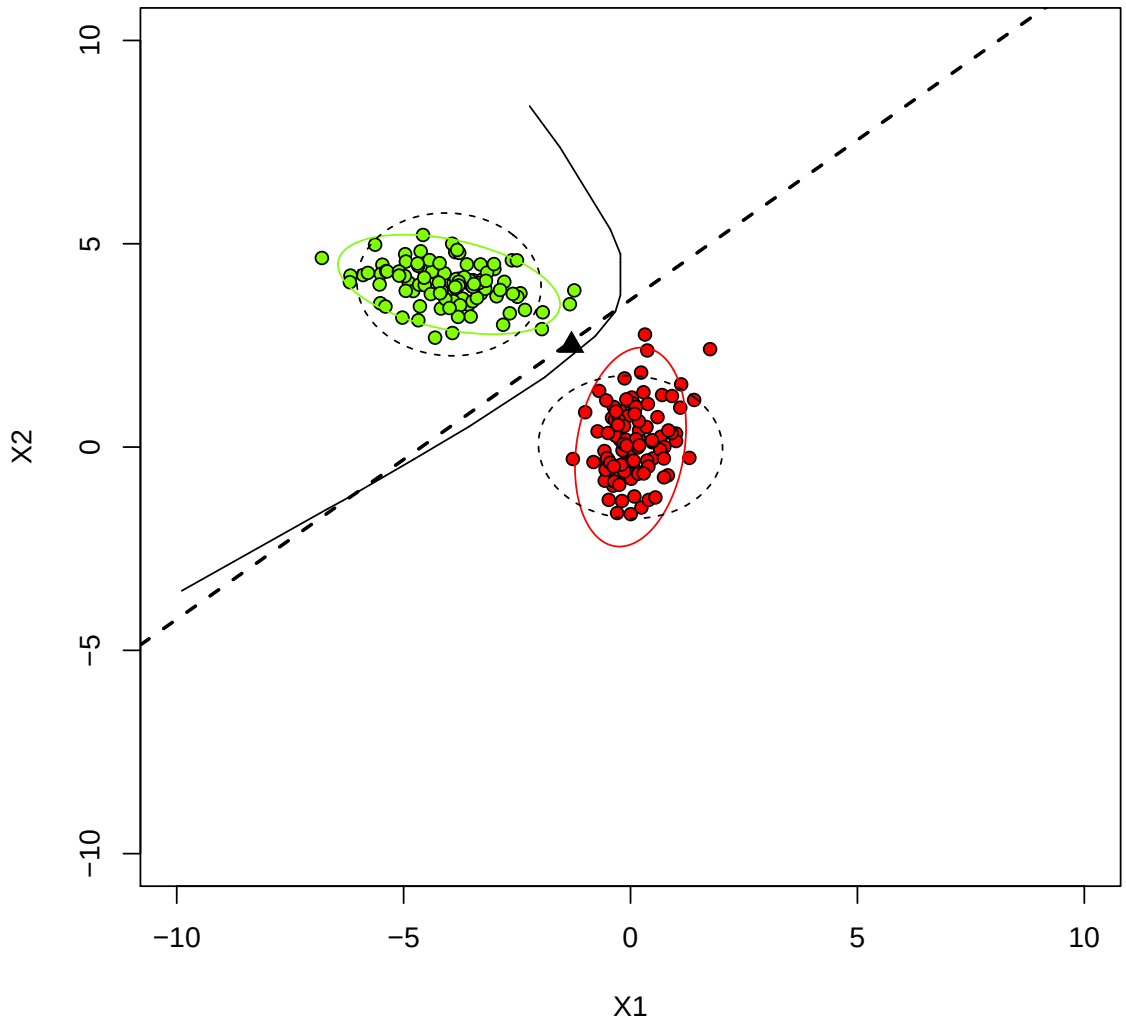


Figure: For $n_g = 100$

LDA: very incorrect assumptions

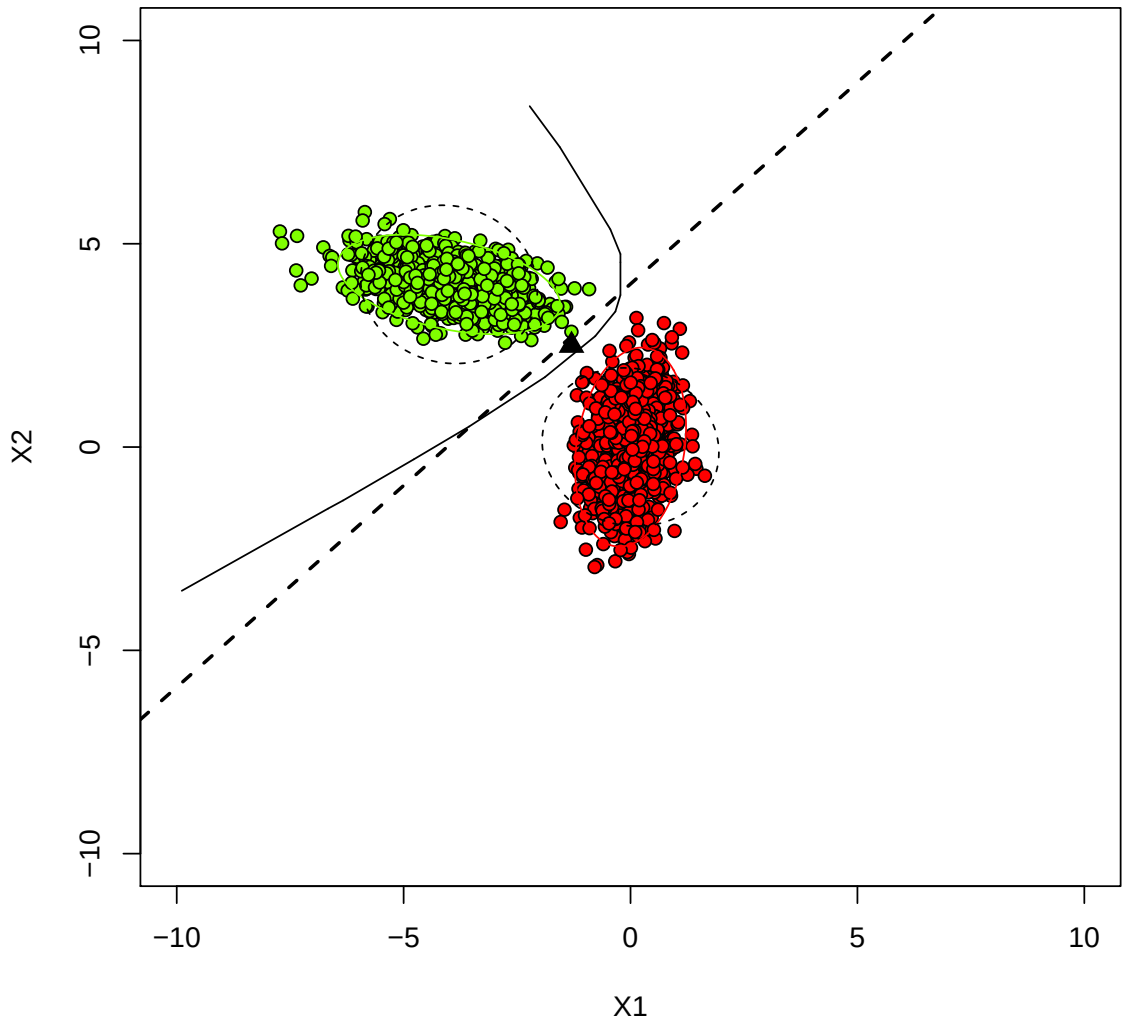


Figure: For $n_g = 1000$

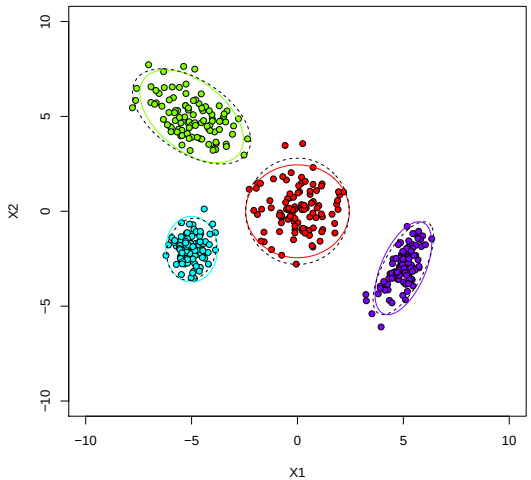
The LDA variance assumption

Returning to the assumption: $\Sigma_g = \Sigma$

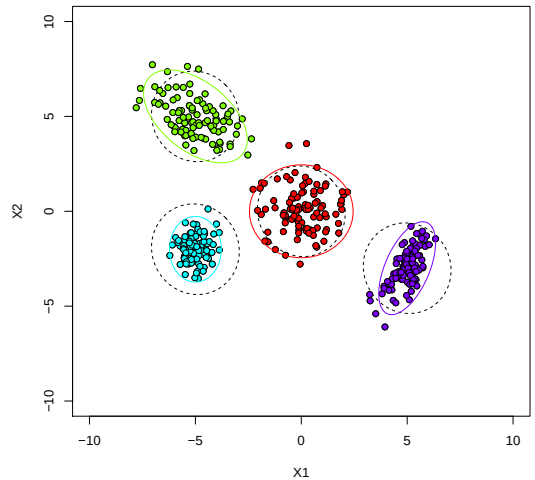
The assumption provides two benefits:

- Allows for estimation when n isn't large compared with $Gp(p+1)/2$
- Lowers the variance of the procedure (but produces bias)

  (This can be seen by the estimation of fewer parameters)



Different $\hat{\Sigma}_g$



All same $\hat{\Sigma}$

The LDA variance assumption

However, when

- n is large compared with $Gp(p+1)/2$
  (Say, $\min n_g \geq 40p(p+1)/2$)
- The induced bias outweighs the variance
  (This is hard to determine. Usually compare the prediction error on test set)

We relax the assumption and let $X|Y = g$ have

- mean μ_g
- variance Σ_k

These additional parameters make the decision boundary quadratic
(Instead of linear)

Quadratic Discriminant Analysis

Quadratic discriminant analysis (QDA)

The formulas for QDA are a bit more complicated, so I'll omit them

However, the motivation is the same: classify with the label of the closest group, taking into account:

- The covariance of every group (Σ_g)
- The relative probability of each group (π_g)

It has almost exactly the same R code:

```
library(MASS)
qda.fit  = qda(Y~.,data=X)

out = predict(qda.fit,X_0)
```

QDA: More flexibility than needed

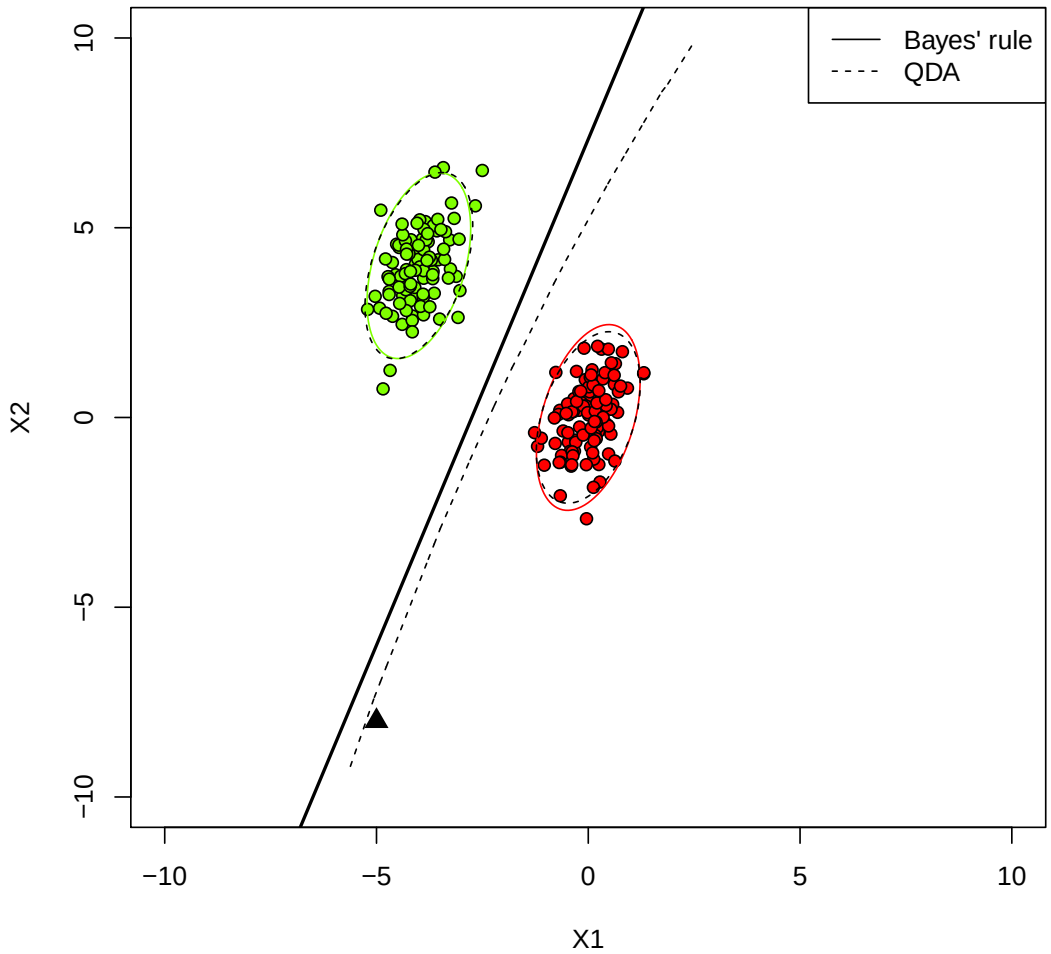


Figure: For $n_g = 100$. Note linear Bayes' rule, nonlinear QDA decision boundary

QDA: More flexibility than needed

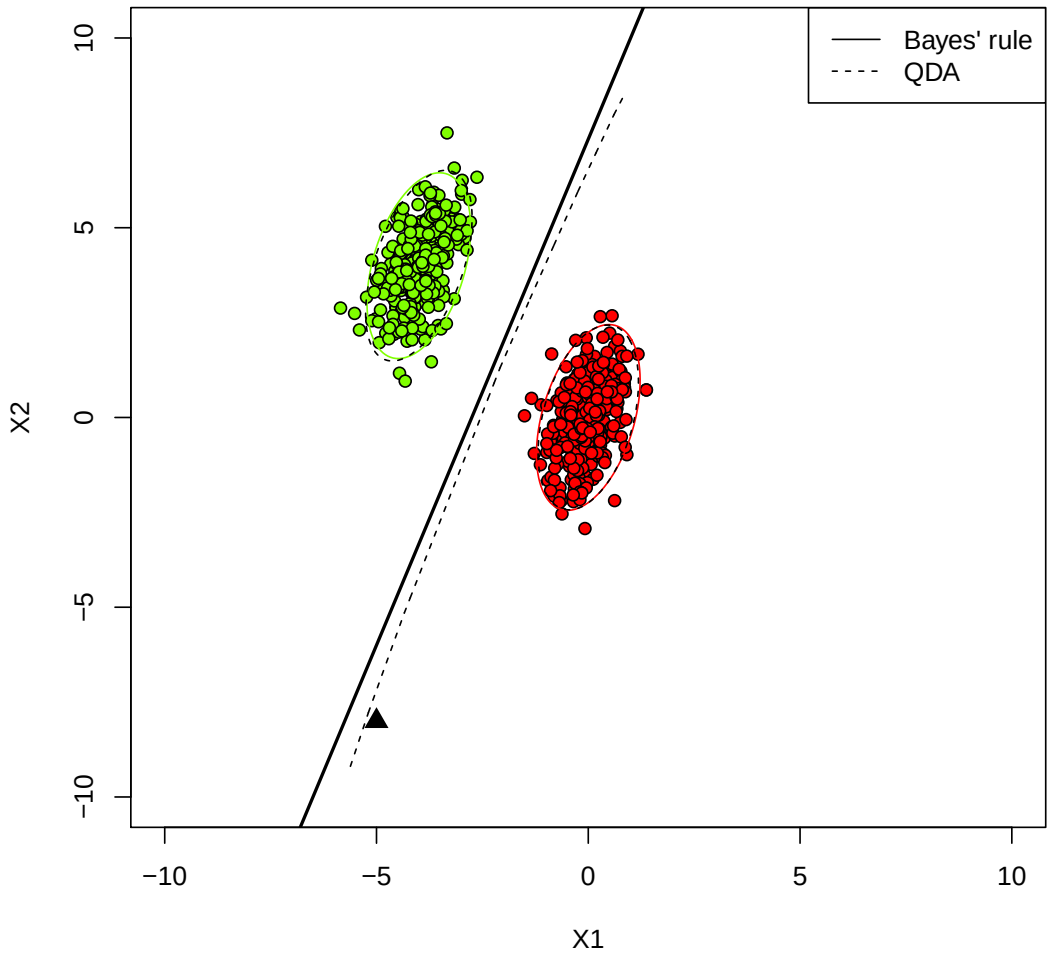


Figure: For $n_g = 300$. Note linear Bayes' rule, nonlinear QDA decision boundary

QDA: More flexibility than needed

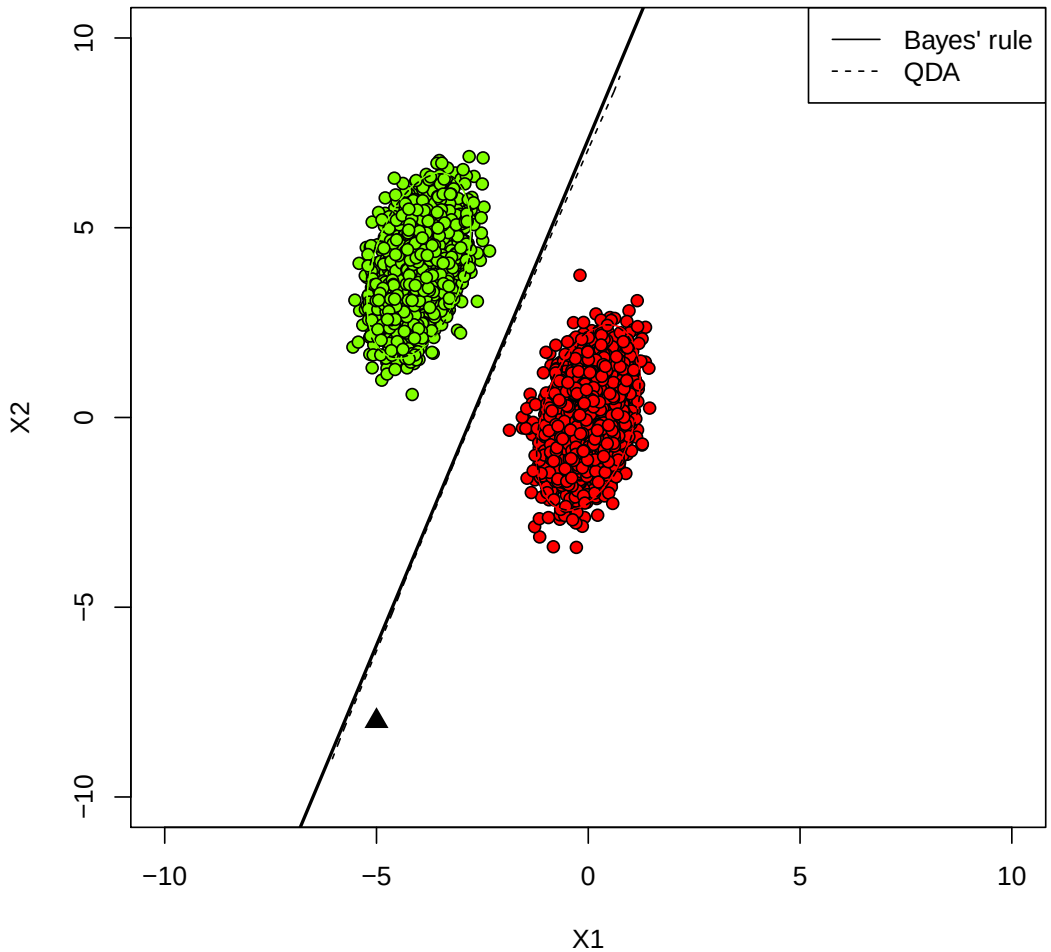


Figure: For $n_g = 2000$. Note linear Bayes' rule. The nonlinear QDA decision boundary has converged to Bayes' rule

QDA: Different Σ_g assumption needed

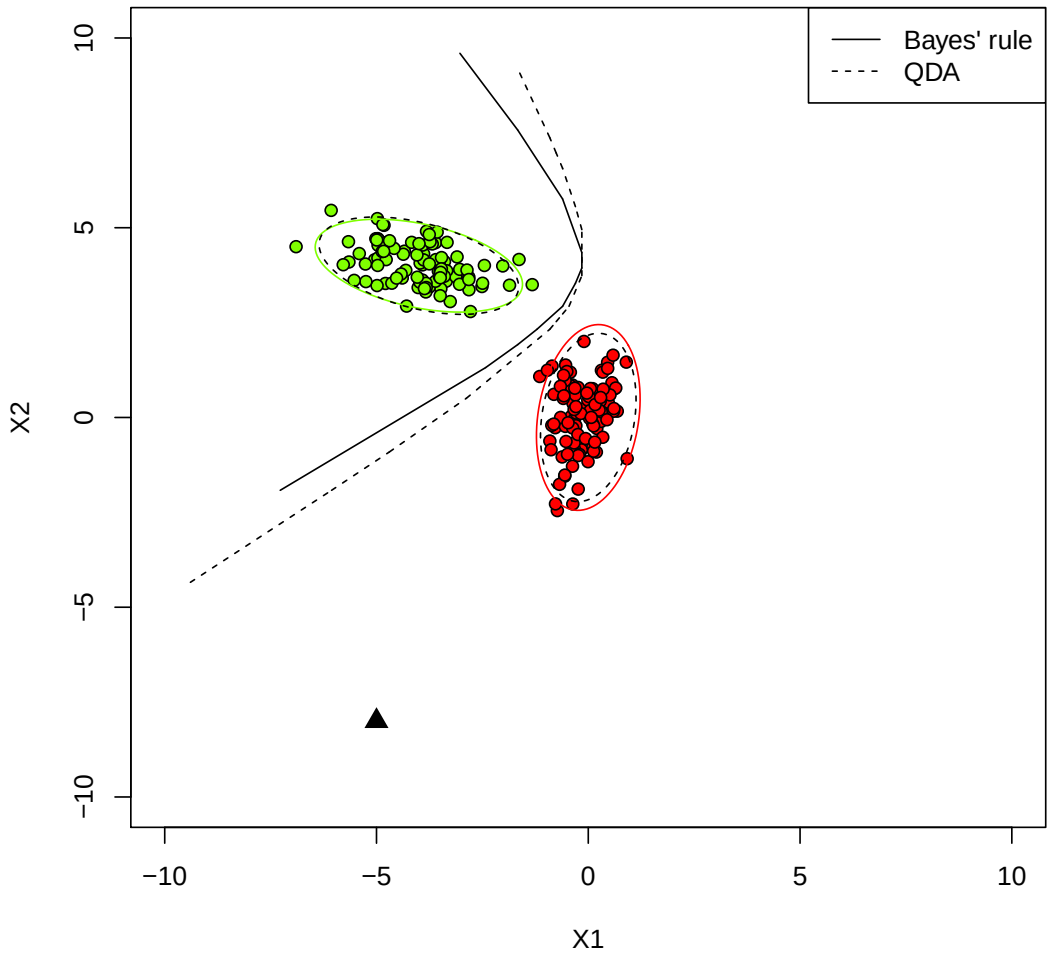


Figure: For $n_g = 100$. Note nonlinear Bayes' rule, nonlinear QDA decision boundary

QDA: Different Σ_g assumption needed

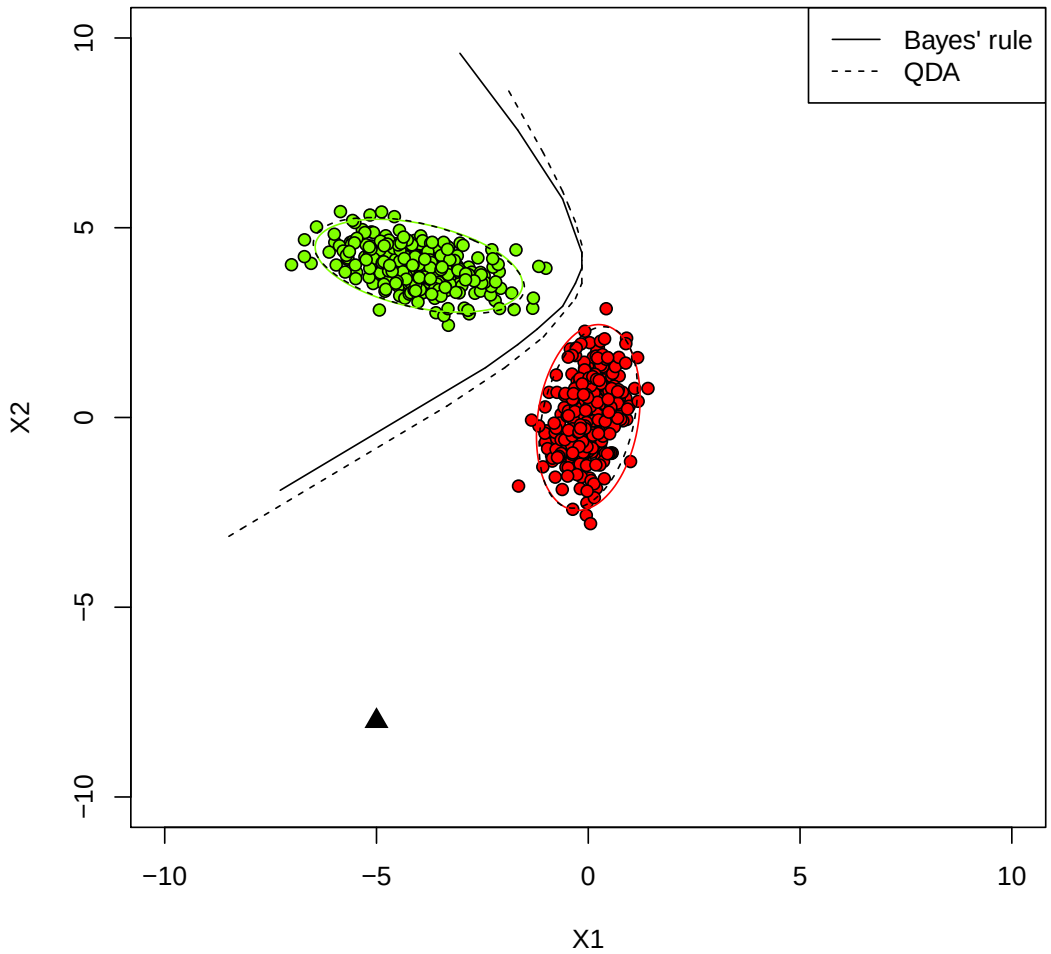


Figure: For $n_g = 300$. Note nonlinear Bayes' rule, nonlinear QDA decision boundary

QDA: Different Σ_g assumption needed

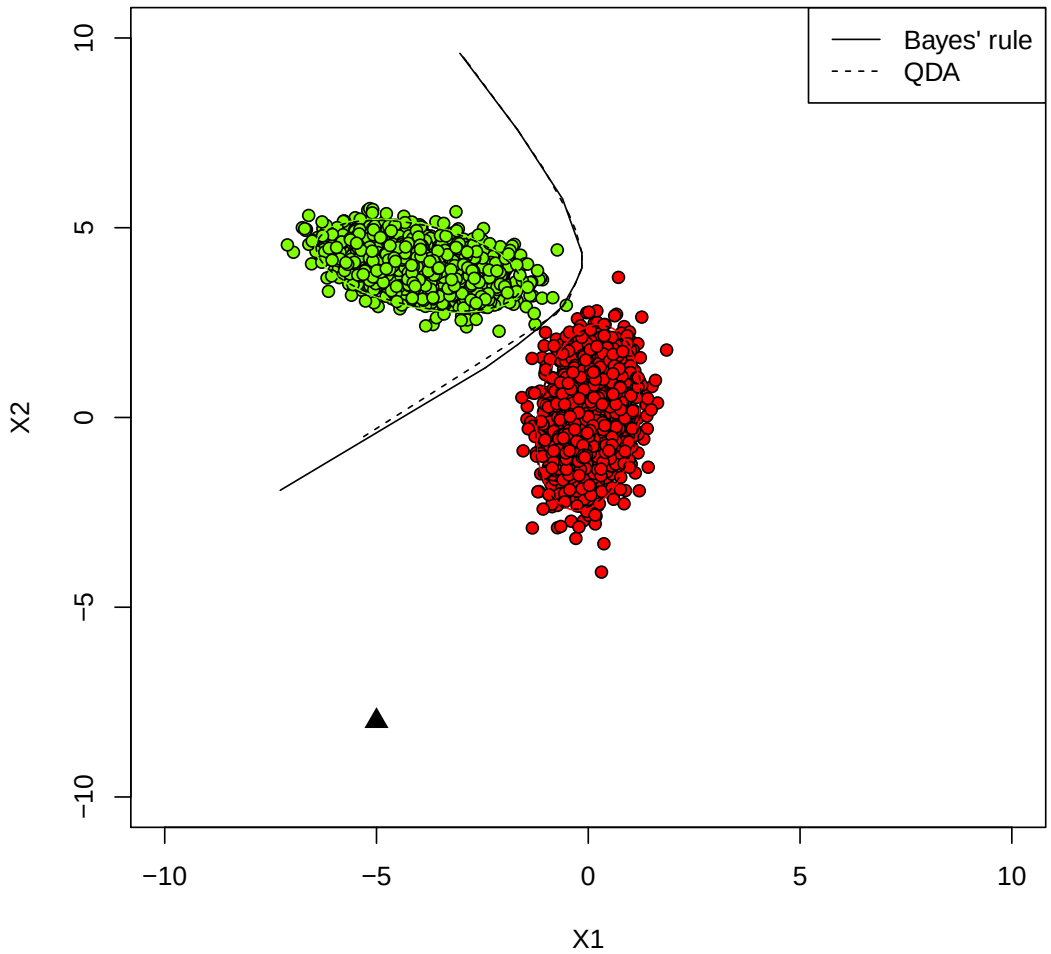


Figure: For $n_g = 2000$. Note nonlinear Bayes' rule, nonlinear QDA decision boundary

LDA vs. QDA: under correct assumptions

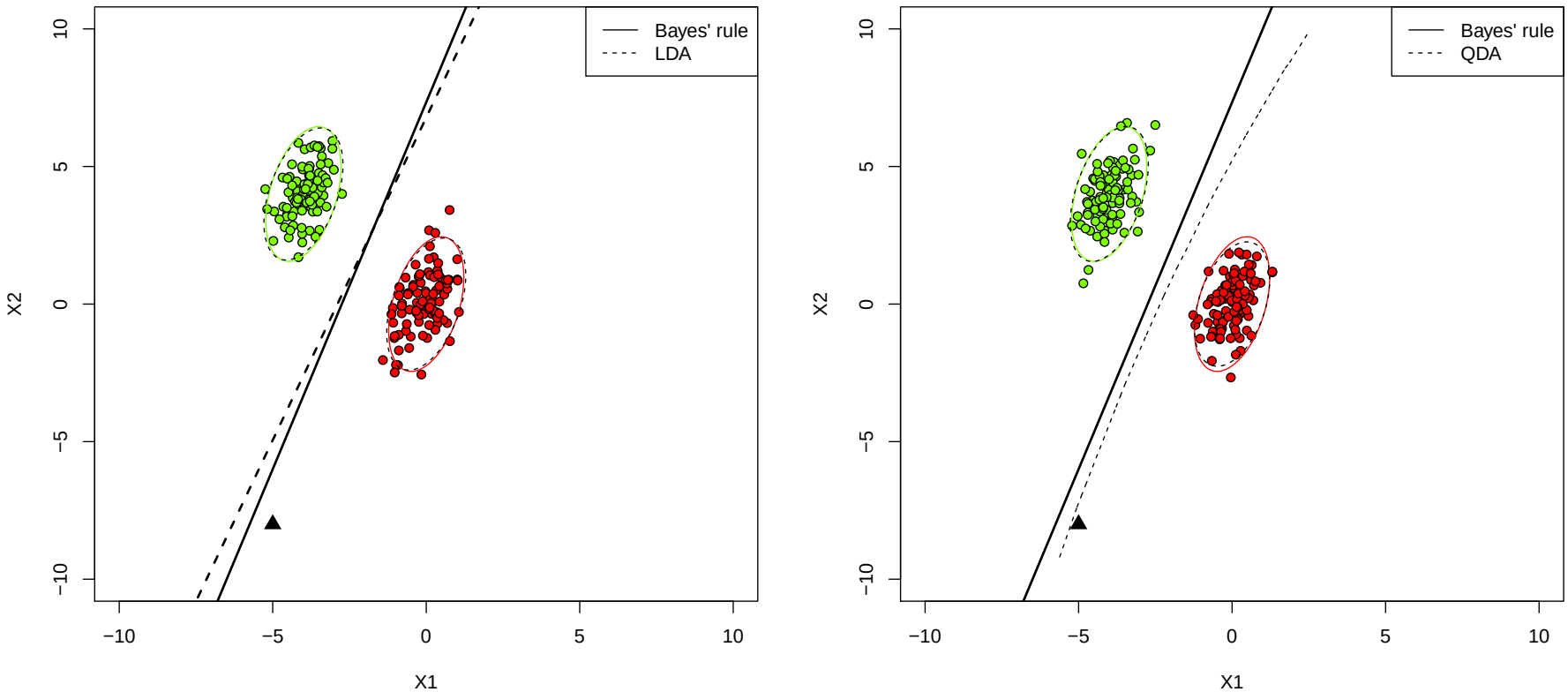


FIGURE: For $n_g = 100$

LDA vs. QDA: very incorrect assumptions

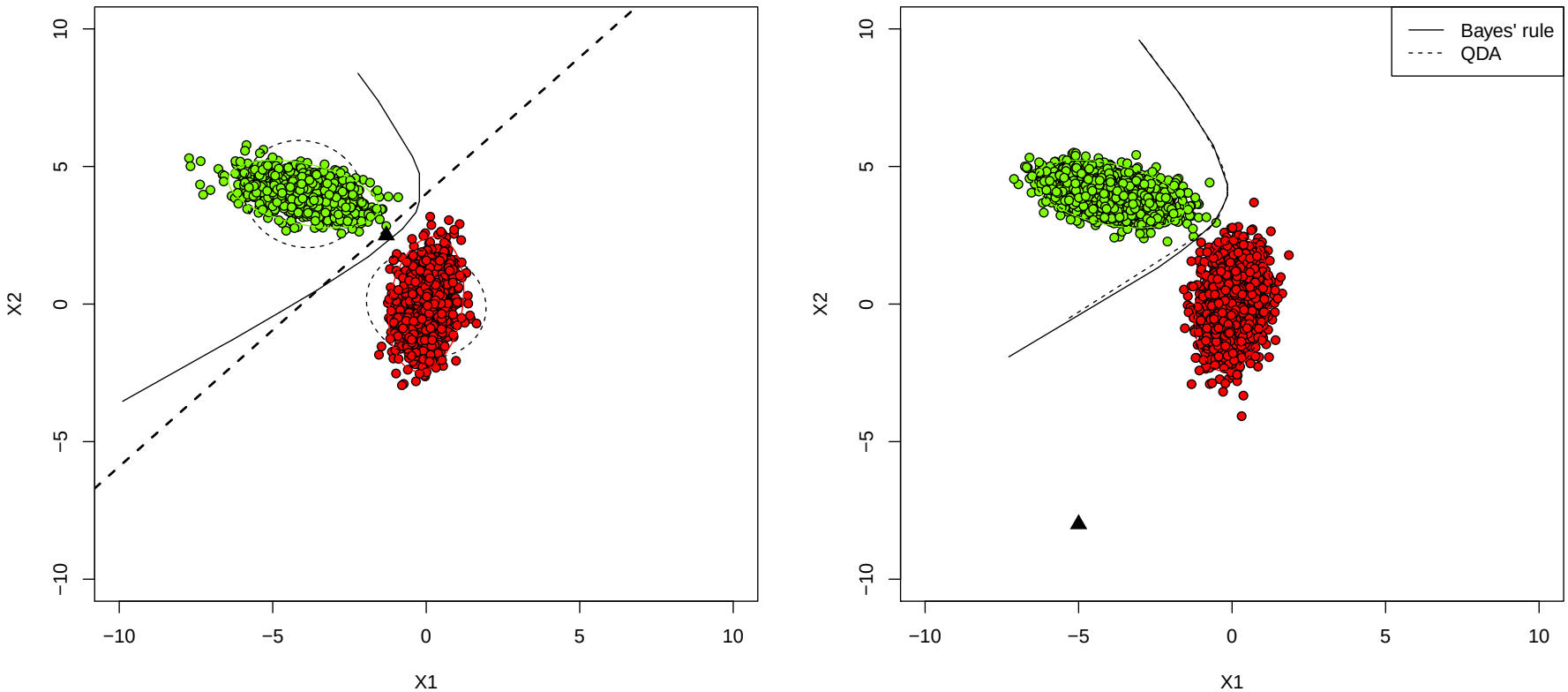


Figure: LDA $n_g = 1000$, QDA $n_g = 2000$