3 LDA "linear discriminant analysis"

Logistic regression involves direction modeling $P(Y = k|X = x)$ using the logistic function for the case of $\underbrace{\text{two response classes}}$. We now consider a less direct approach.

Idea:

Model the distribution of the predictors X separately in each of the response classes (given Y) and then use $\underline{\text{Bayes theorems}}$ to flip these around and get estimates for $P(Y=k|X=x)$.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$

Why do we need another method when we have logistic regression?

* 1. We might have more than 2 response classes.

even with just 2 class in the response

2. If n is small and the distribution of the predictors is approximately normal in each class, LDA is more stable than logistic regression.

3. When classes are well-separated, the parameter estimates in logistic regression are suprisingly unstable.

3.1 Bayes' Theorem for Classification

notation

Suppose we wish to classify an observation into one of K classes, where $K \geq 2$.

categorical Y with K classes (possible distinct and unordered values).

π_k - overall or "prior" probability that a randomly chosen obervation falls into the k^{th} class.

→ could know this from domain knowledge
could estimate from training data

$f_k(x) = P(X = x \mid Y = k)$ ← only works in discrete case

probability that X falls into a small region around x given $Y = k$ (cts).

conditional density function of X for an observation that comes from class k.

$$P(\overset{A}{Y = k} \mid \overset{B}{X = x}) = \frac{\overset{P(\overset{A}{Y=k})}{\pi_k} \; \overset{P(\overset{B}{X=x} \mid \overset{A}{Y=k})}{f_k(x)}}{\underbrace{\sum_{l=1}^{K} \pi_l f_l(x)}_{\substack{P(X=x) \\ B}}}$$

Bayes theorem

Use the same abbreviation as before
$p_k(x) = P(Y = k \mid X = x)$
"posterior probability" that an observation $X = x$ comes from the k^{th} class.

In general, estimating π_k is easy if we have a random sample of Y's from the population.

could get from domain knowledge

computing the fraction of training observations that come from the k^{th} class.

Estimating $f_k(x)$ is more difficult unless we assume some particular forms.

If we can estimate $f_k(x)$ we can classifier that is close to the "best" classifier (more later).

" optimal" classifier: assuming we know $p_k(x) = P(Y=k \mid X=x)$
- assignment to class with the highest posterior probability $p_k(x)$.
- "Bayes classifier" and is known to be optimal in terms of overall error rate.
i.e. we can do no better than the Bayes classifier.

3.2 p = 1

Let's (for now) assume we only have 1 predictor. We would like to obtain an estimate for $f_k(x)$ that we can plug into our formula to estimate $p_k(x)$. We will then classify an observation to the class for which $\hat{p}_k(x)$ is greatest.

$\hookrightarrow \frac{\pi_k f_k(x)}{\sum_{\ell=1}^{K} \pi_\ell f_\ell(x)}$.

estimating the Bayes classifier!

Suppose we assume that $f_k(x)$ is normal. In the one-dimensional setting, the normal density takes the form

Gaussian

$$f_k(x) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{1}{2\sigma_k^2}(x-\mu_k)^2\right)$$

variance parameter for k^{th} class

mean parameter for k^{th} class

Let's also (for now) assume $\sigma_1^2 = \dots = \sigma_K^2 = \sigma^2$ (shared variance term).

Plugging this into our formula to estimate $p_k(x)$,

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu_k)^2\right)}{\sum_{\ell=1}^{K} \pi_\ell \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu_\ell)^2\right)}$$

not 3.14159..., this denotes the prior prob. that observation falls into ℓ^{th} class.

take log, rearrange.

We then assign an observation $X = x$ to the class which makes $p_k(x)$ tthe largest. This is equivalent to

assign obs. to class which makes

$$\delta_k(x) = x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

largest.

LDA decision criteria is linear in x

$\Rightarrow$ "Linear discriminant analysis"

Example 3.1 Let $K = 2$ and $\pi_1 = \pi_2$. When does the Bayes classifier assign an observation to class 1?

When $\delta_1(x) > \delta_2(x)$?

what x values will make this happen?

$$\Leftrightarrow \quad x\frac{\mu_1}{\cancel{\sigma^2}} - \frac{\mu_1^2}{2\cancel{\sigma^2}} + \cancel{\log(\pi_1)} > x\frac{\mu_2}{\cancel{\sigma^2}} - \frac{\mu_2^2}{\cancel{\sigma^2}} + \cancel{\log(\underbrace{\pi_2}_{=\pi_1})}$$

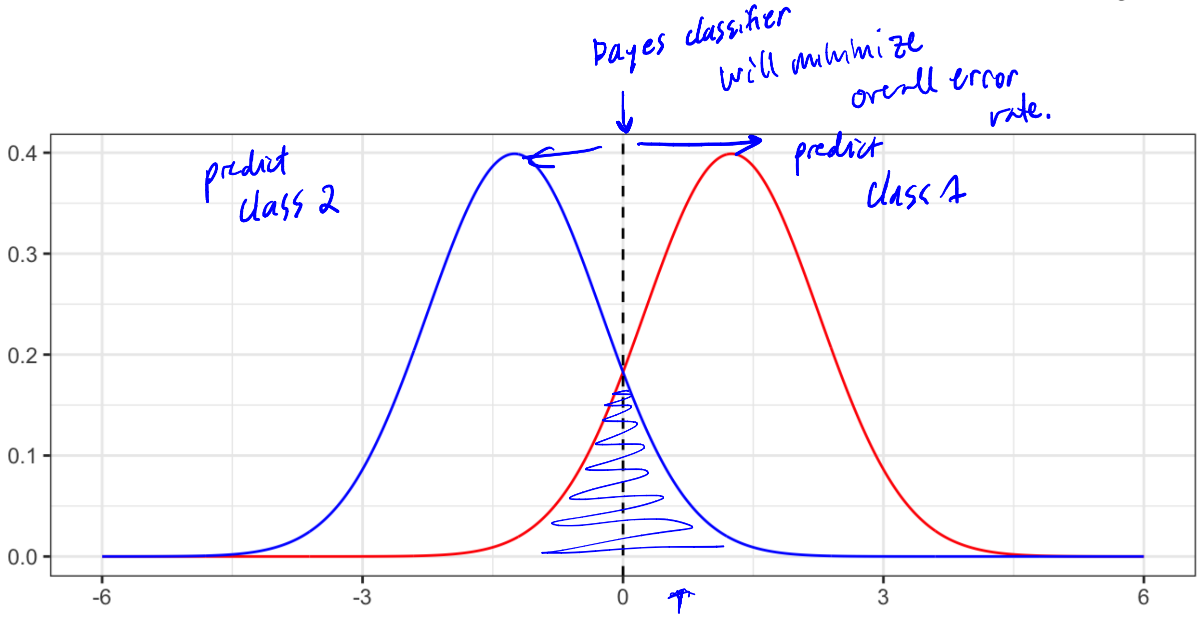
$$2x(\mu_1 - \mu_2) > \mu_1^2 - \mu_2^2$$

$(\mu_1-\mu_2)(\mu_1+\mu_2)$

$$x > \frac{\mu_1+\mu_2}{2}$$

Bayes decision boundary.

$\Rightarrow$ then we will predict class 1



example where $\pi_1 = \pi_2 = 0.5$

$\mu_2 = -1.25$, $\mu_1 = 1.25$, $\sigma = 1$ $\Rightarrow$ Bayes decision boundary would be at 0.

In this case we know $f_k(x) \sim N(\mu_k, \sigma^2)$ $\Rightarrow$ we can create this Bayes classifier!

In practice, even if we are certain of our assumption that X is drawn from a Gaussian distribution within each class, we still have to estimate the parameters

$\underbrace{\mu_1, \dots, \mu_K}_{\text{means of each class}}, \underbrace{\pi_1, \dots, \pi_K}_{\text{prior prob. of falling in each class}}, \sigma^2.$ ← shared variance

The *linear discriminant analysis* (LDA) method approximated the Bayes classifier by plugging estimates in for π_k, μ_k, σ^2.

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i: y_i = k} x_i$$ ← average of training observations in class k.

$$\hat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i: y_i = k} (x_i - \hat{\mu}_k)^2$$ ← weighted average of class variances

n_k = # training observations in class k $\qquad K$ = # classes.

n = total # training observation.

Sometimes we have knowledge of class membership probabilities $\pi_1, \dots, \pi_K$ that can be used directly. If we do not, LDA estimates π_k using the proportion of training observations that belong to the kth class.

from set up of experiment from science, directly from knowledge of the problem.

$$\hat{\pi}_k = \frac{n_k}{n}$$

The LDA classifier assignes an observation $X = x$ to the class with the highest value of

$$\hat{\delta}_k(x) = x \frac{\hat{\mu}_k}{\hat{\sigma}^2} - \frac{\hat{\mu}_k^2}{2\hat{\sigma}^2} + \log(\hat{\pi}_k)$$

linear in x

$\pi_1 = \pi_2 = .5$
$\sigma = 1$

$n_1 = n_2 = 20$

$\mu_1 = 1.25$
$\mu_2 = -1.25$

⇓

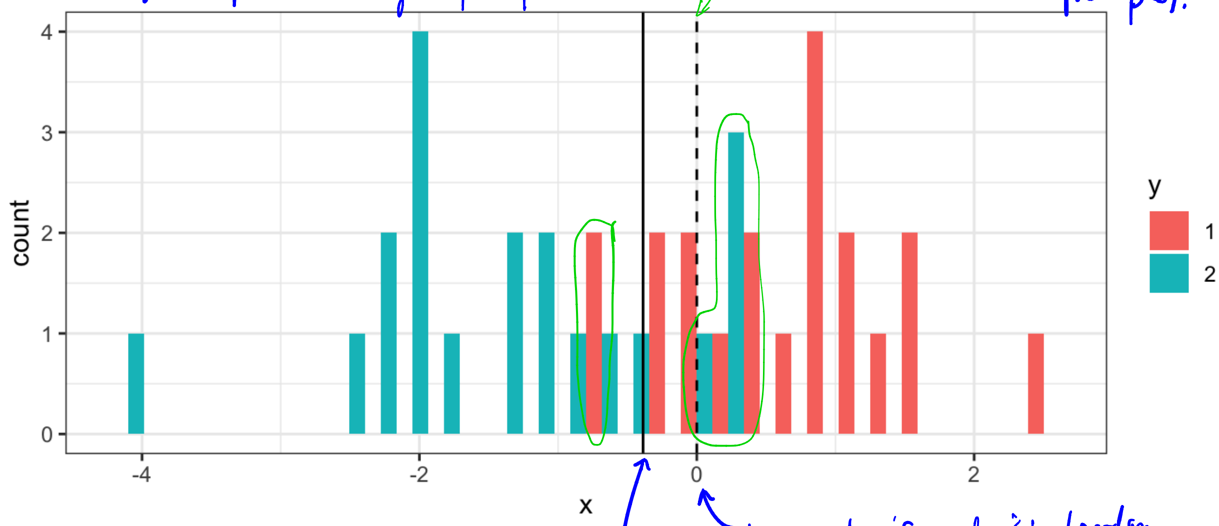
sampled (x, y)

⇓

calculated LDA decision boundary based on $\hat{\delta}$ (based on data).

histogram of randomly sampled points from class 1 and class 2 from prev. plot.

best I could possibly do.



LDA decision boundary

$$\frac{\hat{\mu}_1 + \hat{\mu}_2}{2}$$

Bayes classifier decision boundary.

$$\frac{\mu_1 + \mu_2}{2} = 0.$$

predicted based on LDA decision boundary

confusion matrix (test data) 20k from each class.

```
##     pred
## y       1     2
##   1 18966  1034
##   2  3855 16145
```

true value of test data y

test cases got right

test cases I got wrong.

Best we can possibly do.

The LDA test error rate is approximately 12.22% while the Bayes classifier error rate is approximately 10.52%.

The Bayes error rate is the best we can possibly do in this problem!

the only reason we can get a good estimate of this is because this is simulated data.

The LDA did almost as well!

The LDA classifier results from assuming that the observations within each class come from a normal distribution with a class-specific mean vector and a common variance σ^2 and plugging estimates for these parameters into the Bayes classifier.

we will relax this assumption later

3.3 p > 1

We now extend the LDA classifier to the case of multiple predictors. We will assume

$X = (X_1, \dots, X_p)$ drawn from multivariate Gaussian dsn w/ class specific mean vector + common covariance.

↳ each individual component follows Normal distribution and some covariance between components.

p×1 vector p×p matrix

$\sim N_p(\mu, \Sigma)$

$EX = \mu$

$Cov(X) = \Sigma$

Formally the multivariate Gaussian density is defined as

$$f(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

transpose

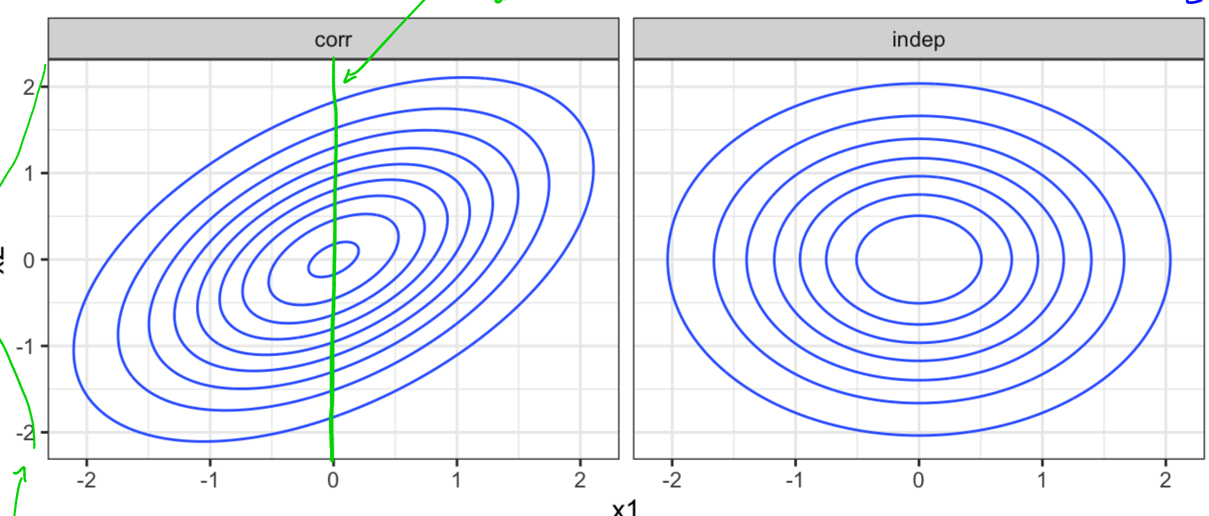
matrix inverse

trace = sum of diag. elements

if you slice across (condition) ⇒ Normal

$X_1, \dots, X_p$ independent ⇒ $cov(X_1, X_2) = 0$ results in "round" dsn.

$cor(X_1, X_2) = \frac{1}{2}$ results in more oval shaped dsn.



if you marginalize out X_1 or X_2, ⇒ Normal distribution.

In the case of $p > 1$ predictors, the LDA classifier assumes the observations in the kth class are drawn from a multivariate Gaussian distribution $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$. common covariance. class specific mean

Plugging in the density function for the kth class, results in a Bayes classifier

assign an observation $X=x$ the class which maximizes

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k$$

this decision rule is still linear in x.

Once again, we need to estimate the unknown parameters (LDA) $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K, \pi_1, \dots, \pi_K, \boldsymbol{\Sigma}$.

use similar formulas to estimate as in $p=1$ case.

To classify a new value $X = x$, LDA plugs in estimates into $\delta_k(x)$ and chooses the class which maximized this value. $\Rightarrow \hat{\delta}_k(x)$ choose k which maximizes (i.e. estimating the Bayes classifier).

Let's perform LDA on the `Default` data set to predict if an individual will default on their CC payment based on balance and student status. $p=2$.

```
library(MASS) # package containing lda function
lda_fit <- lda(default ~ student + balance, data = Default)
lda_fit
```

specify formula just like with lm and glm

```
## Call:
## lda(default ~ student + balance, data = Default)
##
## Prior probabilities of groups:
##     No    Yes
## 0.9667 0.0333
##
## Group means:
##     studentYes   balance
## No   0.2914037  803.9438
## Yes  0.3813814 1747.8217
##
## Coefficients of linear discriminants:
##                      LD1
## studentYes -0.249059498
## balance     0.002244397
```

estimates of π_k based on class membership in training data $\frac{n_k}{n}$

$\hat{\mu}_{No}$, $\hat{\mu}_{Yes}$: average of each predictor within each class (estimate μ_k)

linear combinations of student and balance used to form the LDA decision rule ($\hat{\delta}$).

compare predictions to actual values → confusion matrix

```r
# training data confusion matrix
table(predict(lda_fit)$class, Default$default)
```

get LDA predictions
→ no new data ⇒ training predictions $\hat{y}$
→ new data ⇒ test predictions.

```
##
##        No   Yes
##   No  9644  252
##   Yes   23   81
```

true; predictions; get wrong; get right

For Default = Yes, only get $\frac{81}{252+81}$ right $\approx 24\%$

overall training error $= 2.75\%$

Why does the LDA classifier do such a poor job of classifying the customers who default?

Only 3.33% of individuals in the training set defaulted!

A simple (but useless) classifier could just predict default = NO and only get 3.33% overall training error.

LDA is trying to approximate Bayes classifier ⇒ yield smallest possible overall error rate (irrespective of which class errors come from).

A CC company may want to avoid misclassifying default = Yes. So we could adjust how we select classes.

with 2 classes, Bayes rule says pick class w/ highest prob ⇒ pick class prob > 0.5.

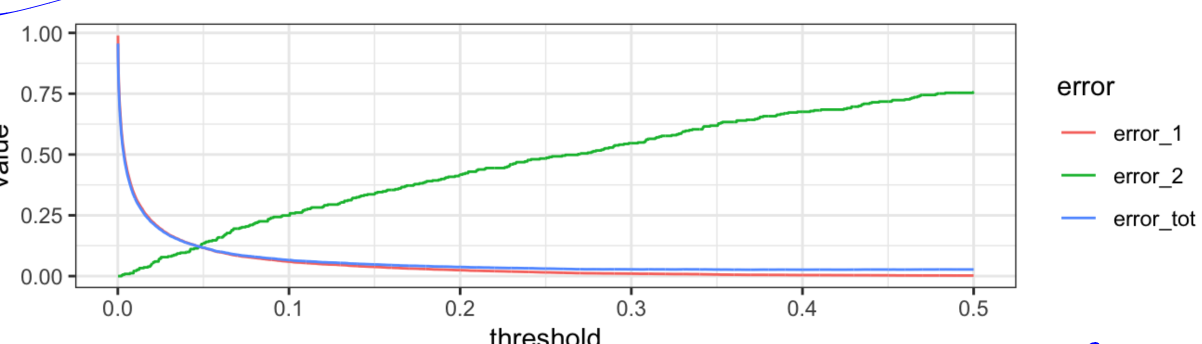
```r
table(predict(lda_fit)$posterior[, "Yes"] > 0.2, Default$default)
```

can adjust using threshold but now we are not approximated Bayes classifier!

```
##
##           No  Yes
##   FALSE 9432  138
##   TRUE   235  195
```

do better at predicting default = Yes

do worse w/ default = No people!



as threshold ↑ 0.5 error (default = No) ↓
error (default = Yes) ↑

How to choose? Domain knowledge. Or method Cross validation (ch. 5) or pick 0.5 because has theoretical justification

3.4 QDA

LDA assumes that the observations within each class are drawn from a multivariate Gaussian distribution with a class-specific mean vector and a common covariance matrix across all K classes.

Quadratic Discriminant Analysis (QDA) also assumes the observations within each class are drawn from a multivariate Gaussian distribution with a class-specific mean vector but now each class has its own covariance matrix.

an observation from k^{th} class

$$X \sim N(\mu_k, \Sigma_k)$$

Under this assumption, the Bayes classifier assignes observation $X = x$ to class k for whichever k maximizes

$$\delta_k(x) = -\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k) - \frac{1}{2}\log|\Sigma_k| + \log \pi_k$$

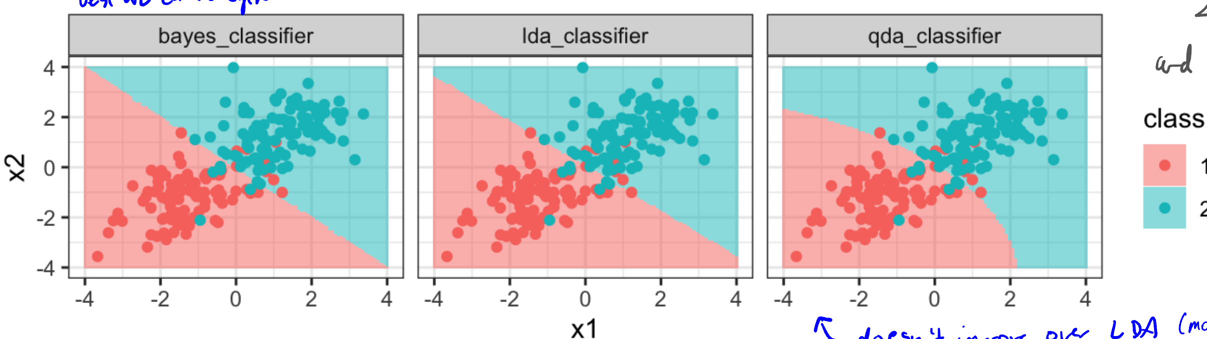
$$= -\frac{1}{2}x^T\Sigma_k^{-1}x + x^T\Sigma_k^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma_k^{-1}\mu_k - \frac{1}{2}\log|\Sigma_k| + \log \pi_k$$

quadratic in x $\Rightarrow$ "quadratic discriminant analysis" $\Rightarrow$ plug in estimates for μ_k, Σ_k, π_k and choose $\arg\max_k \delta_k(x)$.

When would we prefer QDA over LDA?

Best we can do (given I know where the data comes from).

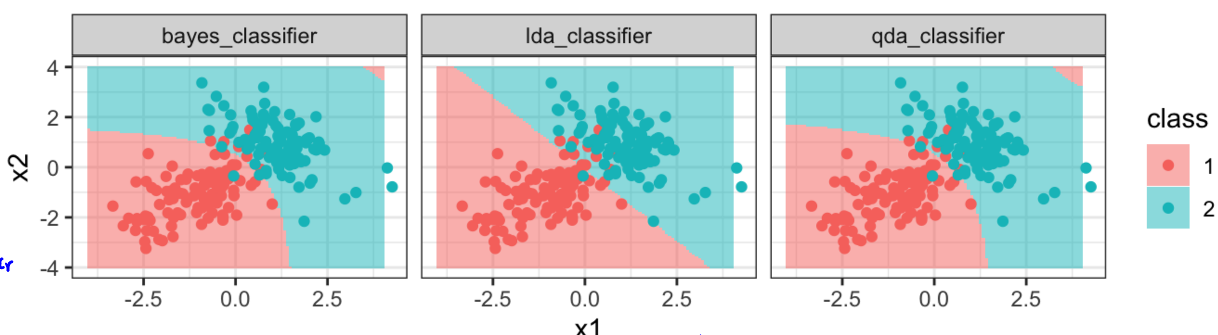
common covariance across groups (classes)
simulated data are points



doesn't improve over LDA (more flexible $\Rightarrow$ more variability but didn't need it.

different Σ_k across classes.

QDA is more similar to Bayes classifier



not flexible enough.

When there are p predictors, estimate Σ_k requires $\frac{p(p+1)}{2}$ parameters $\Rightarrow$ $\frac{Kp(p+1)}{2}$ parameters. (QDA)
vs. LDA linear in x, $\Rightarrow$ only need Kp parameters to estimate (can give good test predictions)
LDA less flexible than QDA but if assumption of global variance is bad, LDA can be wildly off.
LDA > QDA if not many training observations (Ch. 5).

(nonparametric)

4 KNN

K-Nearest neighbors Classification.

Another method we can use to estimate $P(Y = k|X = x)$ (and thus estimate the Bayes classifier) is through the use of K-nearest neighbors.

class k

The KNN classifier first identifies the K points in the training data that are closest to the test data point $X = x$, called $\mathcal{N}(x)$. ← neighborhood.

neighborhood size

Then we estimate $P(Y=k \mid X=x)$ as

$$\frac{1}{K} \sum_{i \in \mathcal{N}(x)} \mathbb{I}(y_i = k)$$

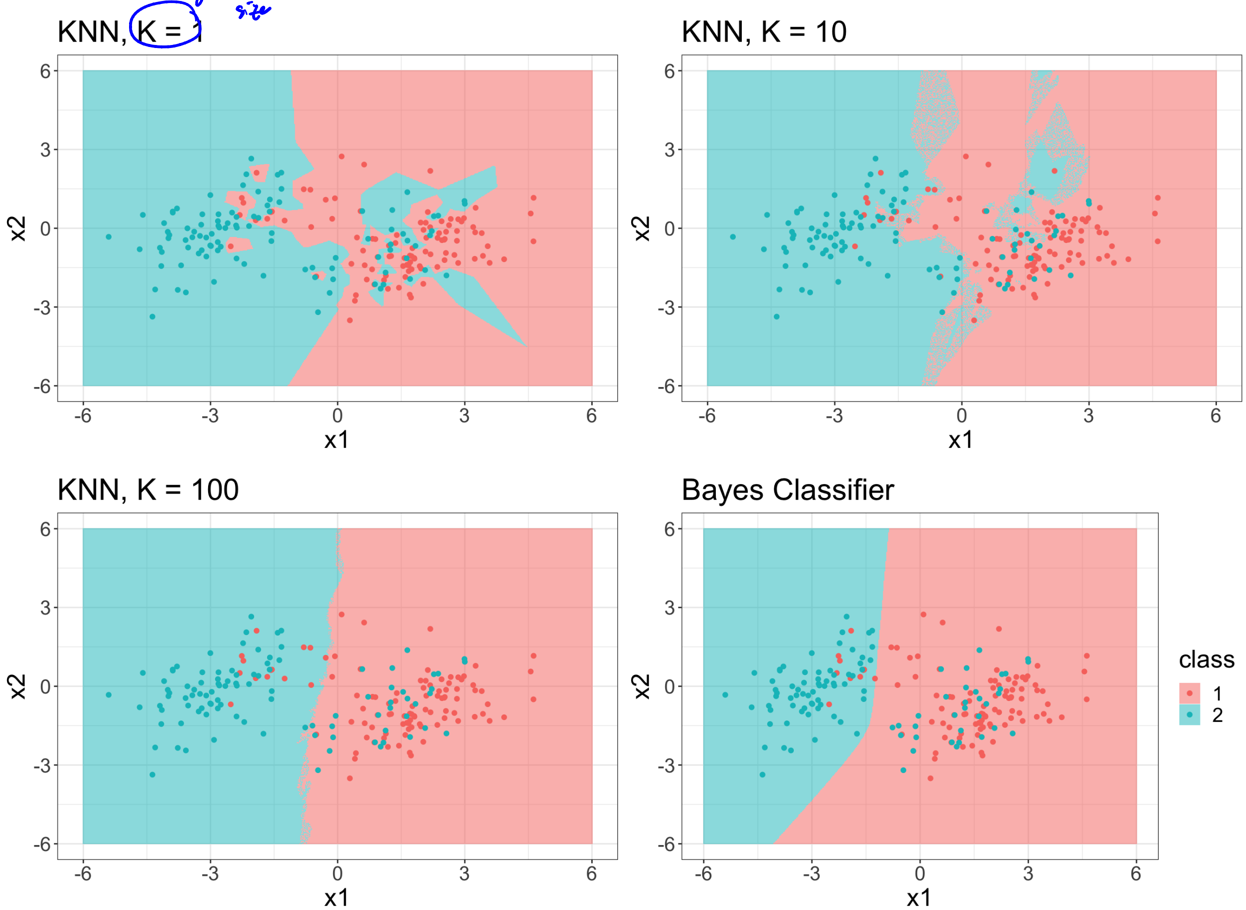
class k

neighborhood size

and then classify x to the class k w/ highest $\hat{P}(Y=k \mid X=x)$.

Just as with regression tasks, the choice of K (neighborhood size) has a drastic effect on the KNN classifier obtained.

overly flexible boundary



Less flexible boundary

approximately linear

— choosing the correct level of flexibility is critical to success of any method (KNN, LDA vs. QDA).

— How to choose? Ch. 5 (coming up next).

5 Comparison

LDA vs. Logistic Regression

LDA & Logistic Regression are closely related.

Consider $\overset{\text{classes}}{K=2}$, $p=1$, and $p_1(x)$, $p_2(x)=1-p_1(x)$

LDA $\log\left(\frac{p_1(x)}{1-p_1(x)}\right) = \log\left(\frac{p_1(x)}{p_2(x)}\right) = \log\left(\frac{\pi_1}{\pi_2}\exp\left[-\frac{1}{2\sigma^2}\left\{(x-\mu_1)^2-(x-\mu_2)^2\right\}\right]\right) = \log\pi_1 - \log\pi_2 - \frac{1}{2\sigma^2}\left[x^2-2x\mu_1+\mu_1^2-x^2+2x\mu_2-\mu_2^2\right]$

$= c_0 + c_1 x$ linear in x

plug in estimates based on $\hat{\mu}_1, \hat{\mu}_2, \hat{\sigma}^2, \hat{\pi}_1, \hat{\pi}_2$

Logistic Regression $\log\left(\frac{p_1}{1-p_1}\right) = \beta_0 + \beta_1 x$ ← linear in x

fit using MLE. ← can be unstable.

Should get similar results, but LDA assumes Gaussian distribution and Logistic regression does not ⟹ whichever assumption holds, should be better.

(LDA & Logistic Regression) vs. KNN

KNN is non parametric, no assumptions made about shape of decision boundary.

⟹ should outperform LDA & Logistic regression when decision boundary is highly non-linear.

KNN doesn't tell us which parameters are important (relationships w/ predictor) no inference.

QDA

Compromise between KNN & linear (LDA & Logistic regression).

Quadratic decision boundary ⟹ can accurately model a wider range of problems.

Not as flexible as KNN ⟹ for problems w/ less training data could have an improvement in prediction over KNN.