3 LDA "linear discriminant analysis"

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

Idea:

$P(X=x|Y=y)$: Model the distribution of the predictors X separately in each of the response classes (given Y) and then use Bayes theorem to flip these 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. When classes are well-separated, the parameter estimates for logistic regression are suprisingly unstable.

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

3. We might have more than 2 response classes.

3.1 Bayes' Theorem for Classification

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

Categorical Y can take on K possible distinct and unordered values.

notation

π_k - overall or "prior" probability that a randomly chosen observation comes from the k^{th} class.

$f_k(x) = P(X=x \mid Y=k)$ (discrete)

probability that X takes value x given Y of class k.

(CTS) density function of X for an observation that comes from class k.

probability that X falls in a small region around x given Y=k.

$$\overset{A}{P(Y=k}|\overset{B}{X=x)} = \frac{\overset{P(B|A)}{\overset{P(X=x|Y=k)}{f_k(x)}}\ \overset{P(A)}{\overset{P(Y=k)}{\pi_k}}}{\underbrace{\sum_{l=1}^{K} \underbrace{\pi_l f_l(x)}_{P(X=x,\,Y=k)}}_{P(X=x)\ P(B)}}$$ (Bayes Theorem)

We will use the abbreviation $p_k(x)$ as before.

> Called the "posterior probability" that an observation comes from k^{th} class given $X=x$.

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

Compute 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 develop a classifier that is close to the "best" classifier (more later).

3.2 p = 1

$p = 1$.

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.

$\hookrightarrow \frac{\pi_k f_k(x)}{\sum_{j=1}^{k} \pi_j f_j(x)}$

estimating the Bayes classifier.

assign to class w/ $p_k(x)$ highest is called the "Bayes classifier" and is known to be the optimal solution, i.e. we can do no better!

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

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

σ_k^2 and μ_k variance and mean parameters for kth 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 $p_k(x)$,

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

prior probability that an observation falls into ℓ^{th} class.

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

(log and rearrange)

assign obs. to class for which

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

is largest.

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

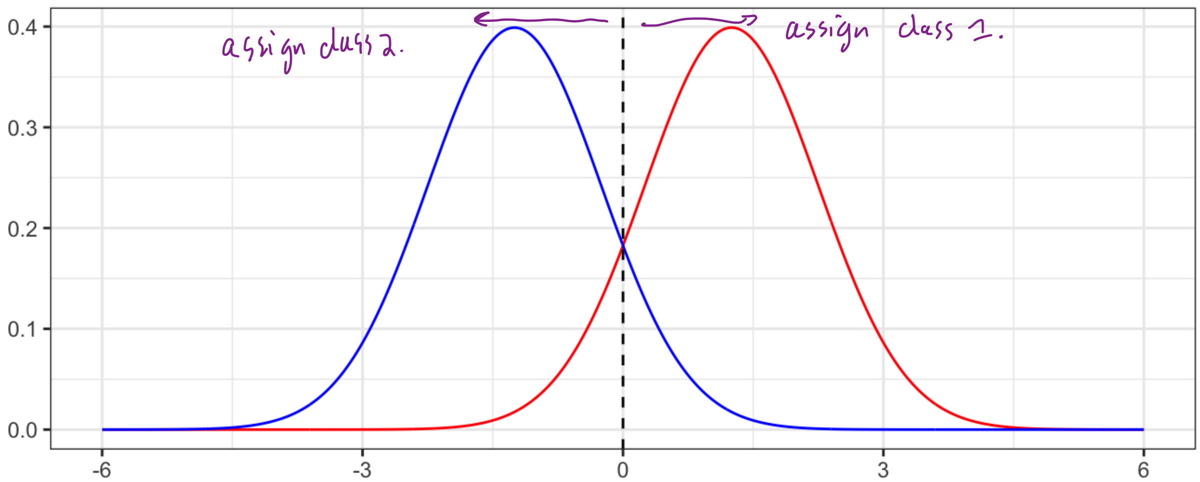
When $\delta_1(x) > \delta_2(x)$?

$$\iff x\frac{\mu_1}{\cancel{\sigma^2}} - \frac{\mu_1^2}{2\cancel{\sigma^2}} + \cancel{\log(\pi_1)} > x\frac{\mu_2}{\cancel{\sigma^2}} - \frac{\mu_2^2}{2\cancel{\sigma^2}} + \cancel{\log(\pi_2)}$$

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

$$\iff x > \frac{\mu_1+\mu_2}{2}$$ decision boundary

Bayes classifier.



example where $\pi_1 = \pi_2 = 0.5$

$\mu_2 = -1.25$
$\mu_1 = 1.25$ } decision boundary at 0
$\sigma = 1$

In this case, we know $f_k(x) \sim N(\mu_k, \sigma^2)$.

=> We can create the 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 $\mu_1, \dots, \mu_K, \pi_1, \dots, \pi_K, \sigma^2$. to estimate Bayes classifier!

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 = total # training obs

n_k = # training obs. in class k

from "science" or from the problem.

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.

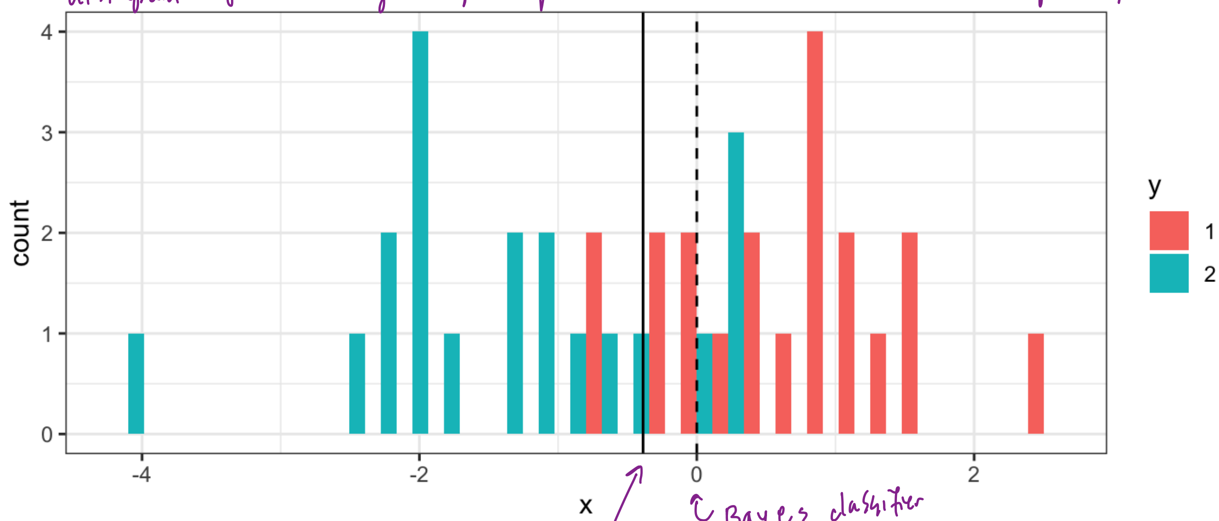
$$\hat{\pi}_k = \frac{n_k}{n}$$

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

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

$n_1 = n_2 = 20$

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



Bayes classifier decision boundary.

LDA decision boundary (based on data). $\frac{\hat{\mu}_1 + \hat{\mu}_2}{2}$.

predicted values

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

true values.

got wrong.

got right

"confusion matrix" on many many simulated test points (20k from each class).

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 with this problem! (we can only estimate it because this is a simulated example).

The LDA approach 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 matrix.
↳ each component follows a Normal dsn and same covariance between components.

$$X \mid Y=k \sim N_p(\underset{p\times 1 \text{ vector}}{\underline{\mu}_k}, \underset{p\times p \text{ matrix}}{\Sigma}) \qquad E[X \mid Y=k] = \mu_k \qquad Cov[X \mid Y=k] = \Sigma$$

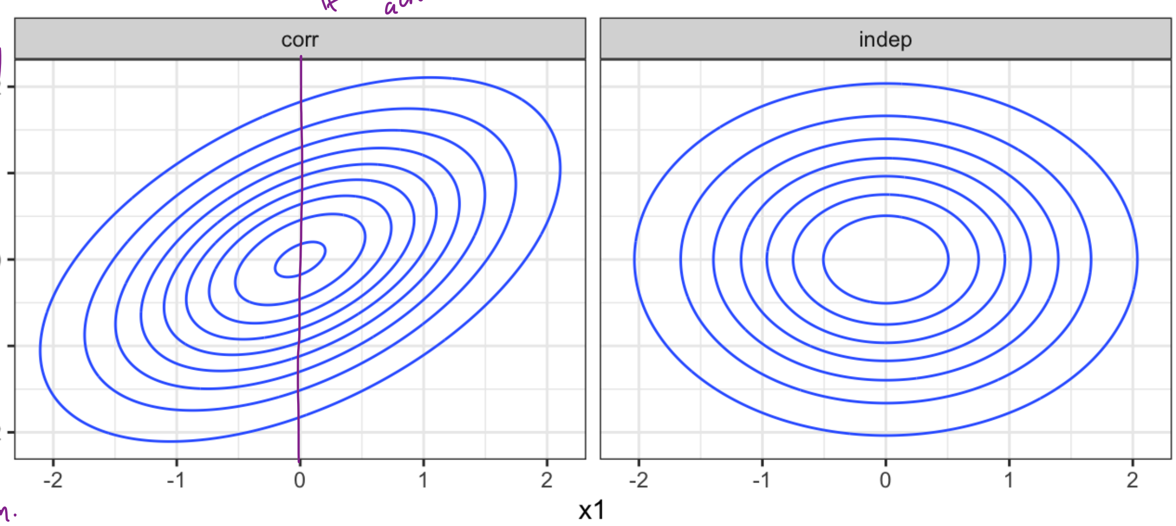
Formally the multivariate Gaussian density is defined as

$$f_k(\underline{x}) = \left[\frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}}\right] \exp\left(-\frac{1}{2}(x-\mu_k)^T \Sigma^{-1} (x-\mu_k)\right)$$

transpose

matrix inverse

"trace" = sum of diagonal elements of Σ.



if we sliced across the dsn ⇒ Normal

$Cov(X_1, X_2) = \frac{1}{2}$

results in more oval shape

If we marginalize out X_1 or X_2 we get Normal dsn.

independence ⇒ $Cov(X_1, X_2) = 0$

results in round/oval shape.

$p=2$ Gaussian density w/ $\mu = \binom{0}{0}$ and 2 Σs.

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 variance.

class specific mean vector

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

assign an observation $X = x$ to 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 linear in x. (hence the name LDA).

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

use similar formulas as $p=1$ case.

To classify a new value $X = x$, LDA plugs in estimates into $\delta_k(x)$ and chooses the class which maximized this value.

$\Rightarrow$ get $\hat{\delta}_k(x)$ choose class k maximizes this quantity (i.e. estimating 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.

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

specify formula just like lm & 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

average of each predictor within each class used to estimate μ_k

linear combinations of student & balance used to form the LDA decision boundary.

training or test data y value.

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

predict= true=

get LDA predictions of either training or test data

comparing predictions to actual values.

True

predicted

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

got wrong

got right

For individuals that did default, only got $\frac{81}{81+252} \approx 24\%$ right!

overall training error rate $\approx 2.75\%$

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

Only 3.33% of individuals in training data defaulted!

A simple (but useless) classifier and just predict default = No and get 3.33% overall training error!

LDA is trying to approximate the Bayes classifier => yields smallest possible overall error rate (irrespective of which class errors come from).

A CC company may want to avoid misclassifying default = Yes people, so can adjust how to select classes.

With 2 classes, usually we pick class w/ highest prob => pick class prob > 0.5.

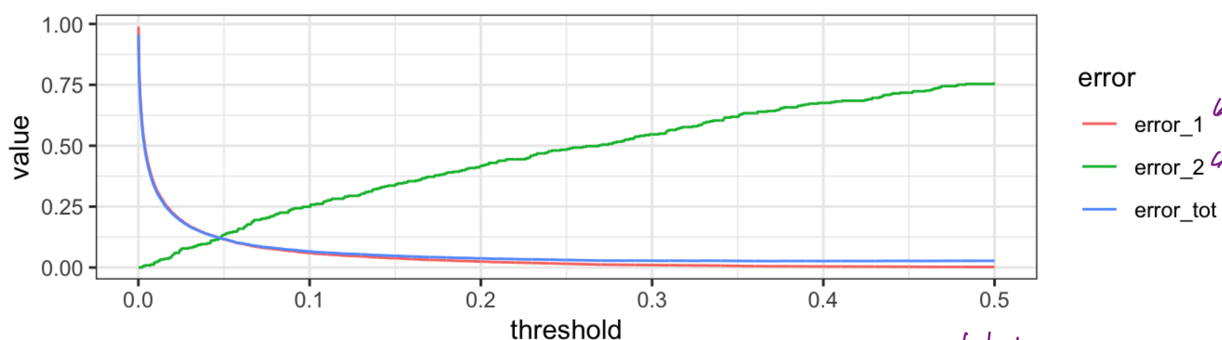
```
table(predict(lda_fit)$posterior[, "Yes"] > 0.2, Default$default)
```

can adjust using a threshold, but no longer approximating Bayes classifier.

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

now do worse w/ default = No people.

do better for default = Yes people $\frac{195}{195+138}$ right



default = No

default = Yes

total error rates.

as threshold ↑, error (default = NO) ↓ so does total error.

error (default = YES) ↑

How to choose? Domain knowledge, or pick 0.5 b/c theoretical justification, or ch. 5 stuff.

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

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

covariance matrix for k^{th} class

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

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

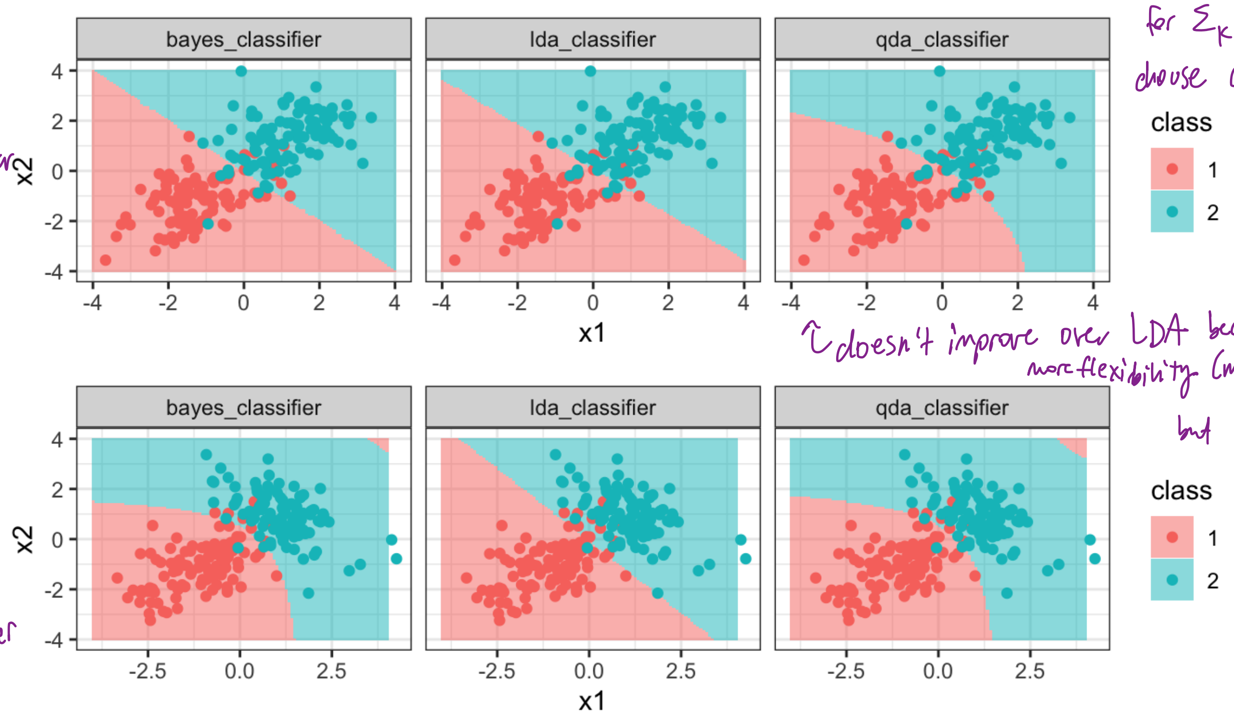
$$= -\frac{1}{2}x^T \Sigma_k^{-1} x + x^T \Sigma_k^{-1}\mu_k - \frac{1}{2}\mu_k^T \Sigma_k^{-1}\mu_k - \frac{1}{2}\log|\Sigma_k| + \log \pi_k.$$

quadratic in x $\Rightarrow$ "quadratic discriminant analysis"

When would we prefer QDA over LDA?

plug in estimates for Σ_k, μ_k, π_k and choose $\arg\max_k \delta_k(x)$.

common covariance $\Rightarrow$ LDA similar to Bayes classifier



doesn't improve over LDA because more flexibility (more variance terms) but didn't need.

different Σ_k $\Rightarrow$ QDA similar to Bayes classifier

When there are p predictors, estimating Σ_k requires estimating $\frac{p(p+1)}{2}$ parameters $\Rightarrow K\frac{p(p+1)}{2}$ parameters

not flexible enough

LDA is linear in x $\Rightarrow$ $K \cdot p$ parameters to estimate. (can still give good predictions).

$\Rightarrow$ LDA much less flexible than QDA, but if assumption of global variance is bad, LDA predictions can be wildly off but LDA > QDA if not many training observations

4 KNN

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.

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

nearest

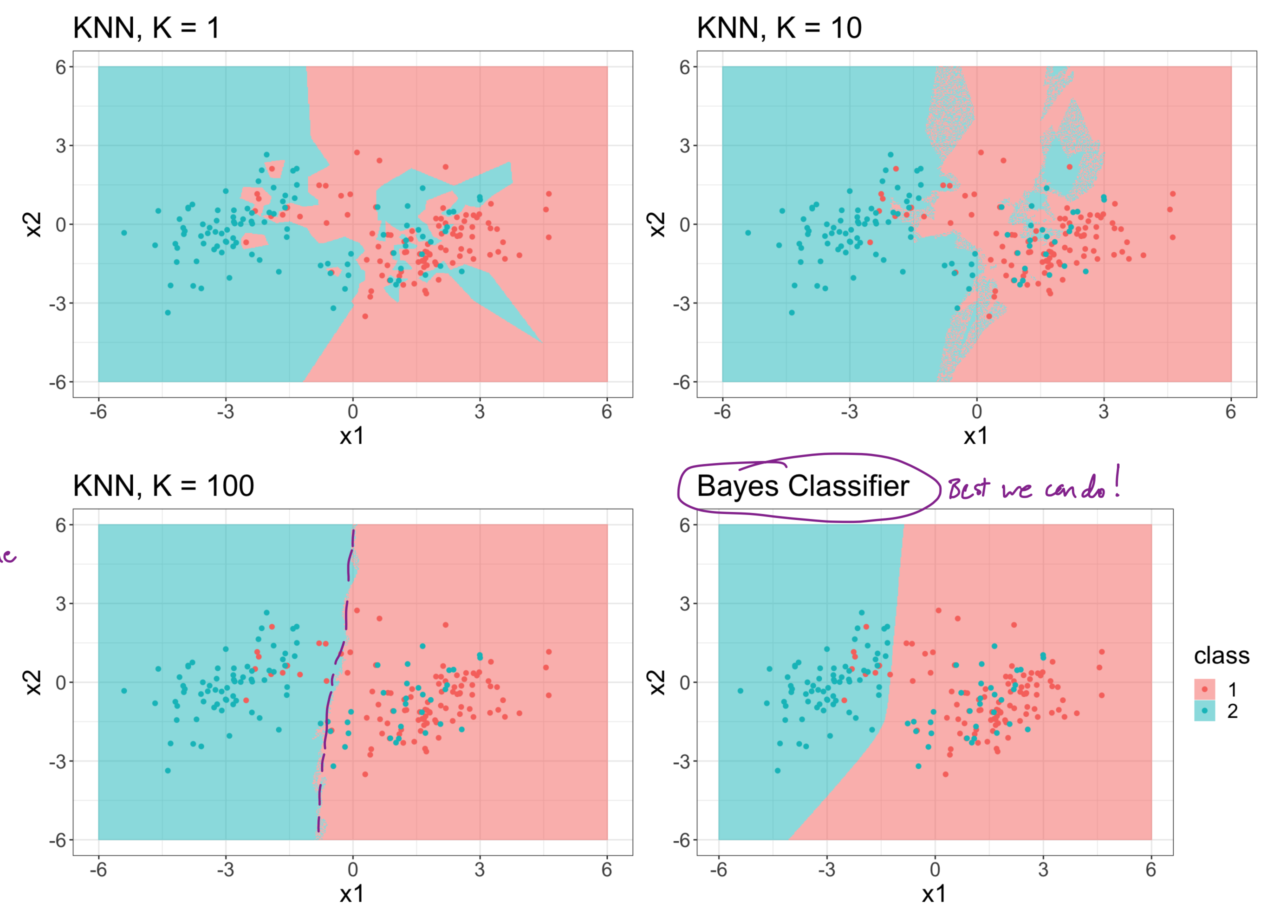
Then we can estimate $P(Y=k \mid X=x)$ as

$$\underbrace{\frac{1}{K}}_{\text{\# of points in neighborhood}} \sum_{i \in N(x)} \mathbb{I}(y_i = k)$$

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

overly flexible boundary (overfitted)

less flexible more linear boundary



- choosing correct level of flexibility is critical to success for any method (KNN, logistic regression, LDA vs. QDA).

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

look at package "class" for performing.

5 Comparison

LDA vs. Logistic Regression

LDA and Logistic Regression are closely related. Consider $K=2$, $p=1$ and $p_1(x)$, $p_2(x)$ are prob. of observing $X=x$ belongs to class 1 and 2 respectively. $p_2(x) = 1-p_1(x)$.

Logistic Regression: $\log\left(\frac{p_1}{1-p_1}\right) = \underbrace{\beta_0+\beta_1}_{\text{fit using MLE}} x$ linear function of x

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

$$= \log\pi_1 - \log\pi_2 - \frac{1}{2\sigma^2}\left[\cancel{x^2}-2x\mu_1+\mu_1^2-\cancel{x^2}+2x\mu_2-\mu_2^2\right] = c_0+c_1x$$

linear function of x

(LDA & Logistic Regression) vs. KNN

should get similar results between 2 methods.

LDA assumes Gaussian dsn w/ common variance. and logistic regression does not. $\Rightarrow$ whichever assumption holds should be better method.

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

$\Rightarrow$ should outperform LDA & logistic regression if decision boundary is highly non linear.

KNN doesnot tell us which parameters are important (have relationships w/ response).

QDA

Compromise between KNN and LDA/logistic regression.

Quadratic decision boundary $\Rightarrow$ can accurately model non linear decision boundaries (wider range of problems).

Not as flexible as KNN $\Rightarrow$ for problems w/ less training data than we need for KNN can have improvement for test predictions.