

# Chapter 4: Classification

The linear model in Ch. 3 assumes the response variable  $Y$  is quantitative. But in many situations, the response is categorical.

e.g. eye color  
cancer diagnosis  
which product a customer would purchase.

In this chapter we will look at approaches for predicting categorical responses, a process known as classification.

Classification problems occur often, perhaps even more so than regression problems. Some examples include

1. A person arrives in the emergency room w/ set of symptoms that could be attributed to 3 root causes. predict which condition the person has.
2. An online banking service must be able to determine if a transaction is fraudulent or not on the basis of IP address, past transaction history, etc.
3. Something is in the street in front of a self-driving car you're riding in. The car must determine if it is a human or another car.

As with regression, in the classification setting we have a set of training observations  $(x_1, y_1), \dots, (x_n, y_n)$  that we can use to "build a classifier". We want our classifier to perform well on the training data and also on data not used to fit the model (**test data**).

fit a model.

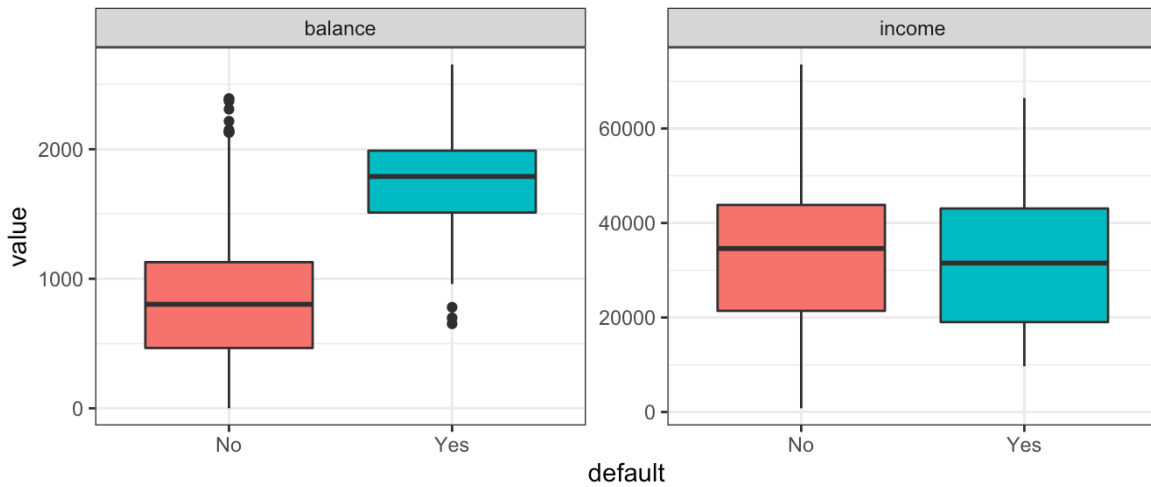
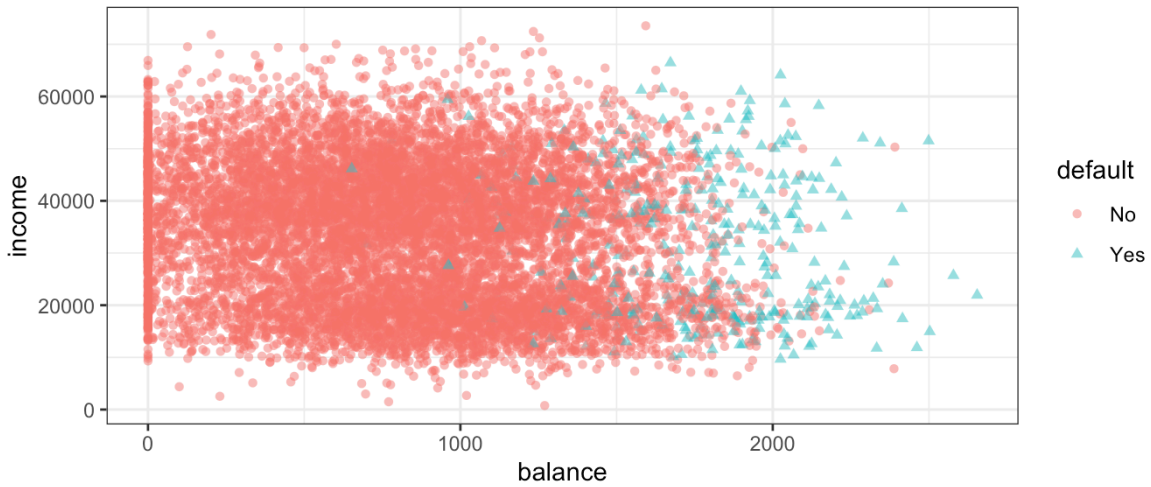
We will use the `Default` data set in the `ISLR` package for illustrative purposes. We are interested in predicting whether a person will default on their credit card payment on the basis of annual income and credit card balance.

↓  
yes or no  $\Rightarrow$  categorical.

head(Default).

|      | default | student | features. |           |
|------|---------|---------|-----------|-----------|
| ##   |         |         | balance   | income    |
| ## 1 | No      | No      | 729.5265  | 44361.625 |
| ## 2 | No      | Yes     | 817.1804  | 12106.135 |
| ## 3 | No      | No      | 1073.5492 | 31767.139 |
| ## 4 | No      | No      | 529.2506  | 35704.494 |
| ## 5 | No      | No      | 785.6559  | 38463.496 |
| ## 6 | No      | Yes     | 919.5885  | 7491.559  |

okay separation



relationship between balance and default

(in most real world problems the relationship is not so clear).

# 1 Why not Linear Regression?

I have said that linear regression is not appropriate in the case of a categorical response. Why not?

Let's try it anyways. We could consider encoding the values of `default` in a quantitative response variable  $Y$

$$Y = \begin{cases} 1 & \text{if default} \\ 0 & \text{otherwise} \end{cases}$$

Using this coding, we could then fit a linear regression model to predict  $Y$  on the basis of `income` and `balance`. This implies an ordering on the outcome, not defaulting comes first before defaulting and insists the difference between these two outcomes is 1 unit. In practice, there is no reason for this to be true.

We could let  $Y = \begin{cases} 0 & \text{if default} \\ 1 & \text{otherwise} \end{cases}$

there is no natural reason to choose 0/1,

$$Y = \begin{cases} 1 & \text{if default} \\ 10 & \text{otherwise} \end{cases}$$

but it has an advantage:

Using the dummy encoding, we can get a rough estimate of  $P(\text{default}|X)$ , but it is not guaranteed to be scaled correctly.



predictions don't have to be between 0 and 1

but will provide an ordering.

Additional problem: this cannot be easily extended to more than 2 classes,

We can instead use methods specifically formulated for categorical responses.

# 2 Logistic Regression

Let's consider again the `default` variable which takes values `Yes` or `No`. Rather than modeling the response directly, logistic regression models the probability that  $Y$  belongs to a particular category.

e.g.  $P(\text{default} = \text{Yes} \mid \text{balance})$   
 which can be abbreviated as  $p(\text{balance}) \in [0, 1]$ .

For any given value of `balance`, a prediction can be made for `default`.

e.g. predict `default = Yes` if  $p(\text{balance}) > 0.5$

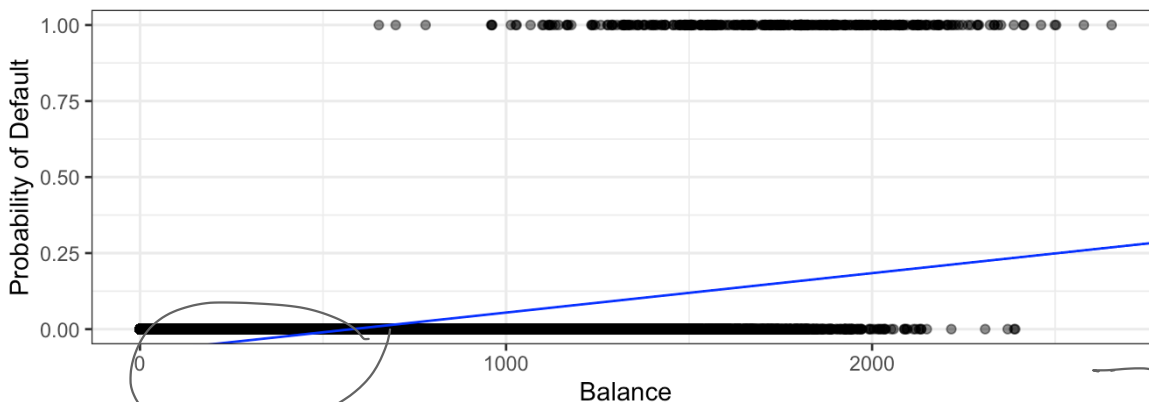
or the company could be more conservative and predict `default = Yes` if  $p(\text{balance}) > 0.1$   
threshold.

## 2.1 The Model

How should we model the relationship between  $p(X) = P(Y = 1 \mid X)$  and  $X$ ? We could use a linear regression model to represent those probabilities

using 0/1 encoding  
 - define as success (e.g. default = Yes).

$$p(x) = \beta_0 + \beta_1 x$$



problem:  
 for balances close to 0,  
 we predict negative  
 prob. of default.

neither of these  
 make sense!

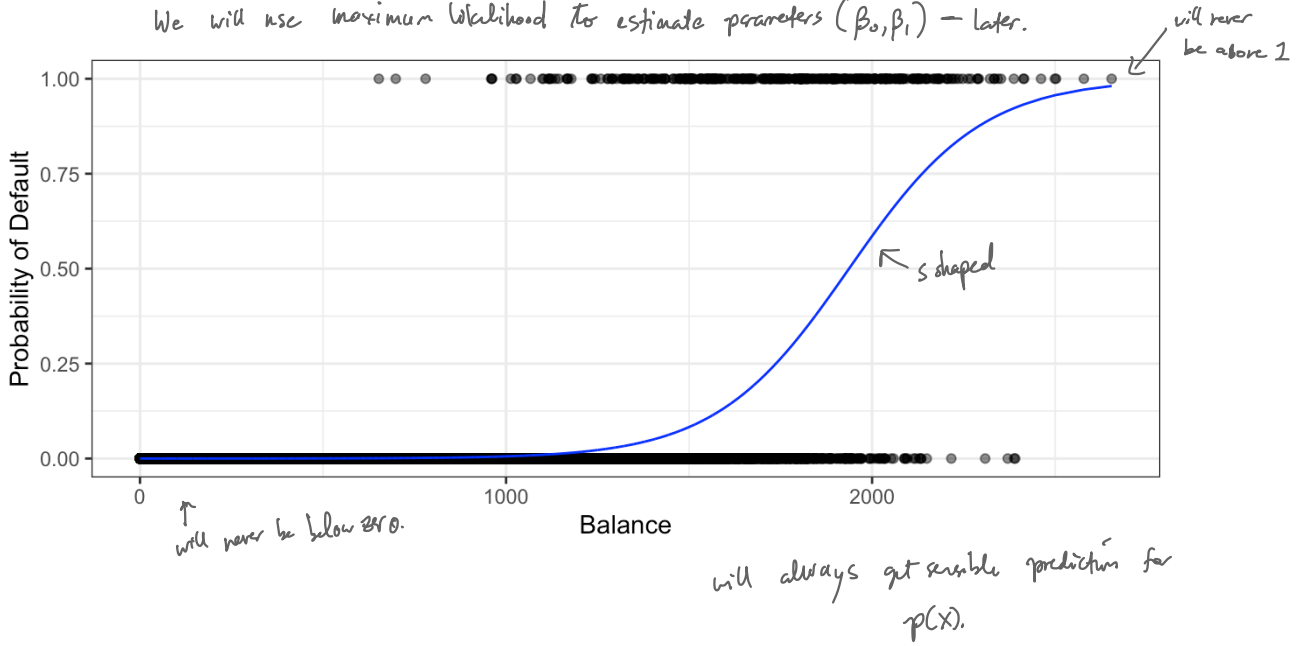
if we had a  
 very large balance,  
 would predict prob  
 > 1 of defaulting

To avoid this, we must model  $p(X)$  using a function that gives outputs between 0 and 1 for all values of  $X$ . Many functions meet this description, but in *logistic regression*, we use the logistic function,

Standard logistic function  
 $f(x) = \frac{e^x}{1+e^x}$

$$p(x) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}}$$

We will use maximum likelihood to estimate parameters  $(\beta_0, \beta_1)$  - later.



After a bit of manipulation,

$$\frac{p(x)}{1-p(x)} = e^{\beta_0 + \beta_1 x}$$

~~~~~

"odds"  $\rightarrow$  can take any value between 0 and  $\infty$

$\nearrow$  low prob of default

$\nearrow$  high prob of default.

ex:  $p(x) = 0.2$  (in 5 people default)  $\Rightarrow$  odds  $= \frac{0.2}{1-0.2} = \frac{1}{4}$

By taking the logarithm of both sides we see,

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$$

“log-odds”  
“logit”

log-odds is linear in  $X$ .

“log-odds”

Recall from Ch. 3 that  $\beta_1$  gives the “average change in  $Y$  associated with a one unit increase in  $X$ .” In contrast, in a logistic model,

increasing  $X$  by one-unit changes the log-odds by  $\beta_1$

$\Leftrightarrow$

increasing  $X$  by one unit multiplies the odds by  $e^{\beta_1}$

However, because the relationship between  $p(X)$  and  $X$  is not linear,  $\beta_1$  does **not** correspond to the change in  $p(X)$  associated with a one unit increase in  $X$ . The amount that  $p(X)$  changes due to a 1 unit increase in  $X$  depends on the current value of  $X$ .

Regardless of the value of  $X$ ,

if  $\beta_1$  is positive  $\Rightarrow$  increasing  $X$  increases  $p(X)$ .

if  $\beta_1$  is negative  $\Rightarrow$  increasing  $X$  reduces  $p(X)$ .

## 2.2 Estimating the Coefficients

The coefficients  $\beta_0$  and  $\beta_1$  are unknown and must be estimated based on the available training data. To find estimates, we will use the method of maximum likelihood.

The basic intuition is that we seek estimates for  $\beta_0$  and  $\beta_1$  such that the predicted probability  $\hat{p}(x_i)$  of default for each individual corresponds as closely as possible to the individual's observed default status.

To do this, use the likelihood  $L(\beta_0, \beta_1 | \{y_i, x_i\}_{i=1}^n) = \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1-p(x_i))$ .  
 $\hat{\beta}_0$  and  $\hat{\beta}_1$  chosen to maximize  $L(\beta_0, \beta_1)$ .  
 (use calculus: derivatives wrt  $\beta_0, \beta_1$ , set = 0, solve).  
 use numeric optimization

Aside: in linear regression model w/  $\epsilon_i \sim N(0, \sigma^2)$   
 $\Rightarrow \hat{\beta}_{MLE} = \hat{\beta}_{OLS}$

```
logistic_spec <- logistic_reg()
```

```
logistic_fit <- logistic_spec |>
  fit(default ~ balance, family = "binomial", data = Default)
```

```
logistic_fit |>
  pluck("fit") |>
  summary()
```

```
##
## Call:
## stats::glm(formula = default ~ balance, family = stats::binomial,
##   data = data)
##
## Deviance Residuals:
##   Min       1Q   Median       3Q      Max
## -2.2697  -0.1465  -0.0589  -0.0221   3.7589
##
## Coefficients:
##   Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.065e+01  3.612e-01  -29.49  <2e-16 ***
## balance      5.499e-03  2.204e-04   24.95  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##   Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1596.5  on 9998  degrees of freedom
## AIC: 1600.5
##
## Number of Fisher Scoring iterations: 8
```

$\hat{\beta}_1 = 0.0055 \Rightarrow$  increase in balance associated w/ increase in prob. of default.

$\Rightarrow$  one unit increase in balance associated w/ average increase of log-odds by .0055 units.

## 2.3 Predictions

Once the coefficients have been estimated, it is a simple matter to compute the probability of `default` for any given credit card balance. For example, we predict that the default probability for an individual with `balance` of \$1,000 is

$$\hat{p}(x) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x}}$$

$$\hat{p}(1000) = \frac{e^{-10.6513 + 0.0055 \times 1000}}{1 + e^{-10.6513 + 0.0055 \times 1000}} = 0.00576$$

In contrast, the predicted probability of default for an individual with a balance of \$2,000 is

$$\hat{p}(2000) = \frac{e^{-10.6513 + 0.0055 \times 2000}}{1 + e^{-10.6513 + 0.0055 \times 2000}} = 0.586 > 0.5$$

maybe we would predict default = yes based on a threshold of 0.5.

(tidy models: `augment`  
`predict`)



## 2.4 Multiple Logistic Regression

We now consider the problem of predicting a binary response using multiple predictors. By analogy with the extension from simple to multiple linear regression,

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$$

$$\Downarrow$$

$$p(x) = \frac{e^{\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p}}{1 + e^{\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p}}$$

Just as before, we can use maximum likelihood to estimate  $\beta_0, \beta_1, \dots, \beta_p$ .

```
logistic_fit2 <- logistic_spec |> logistic regression specification.
  fit(default ~ ., family = "binomial", data = Default)
logistic_fit2 |> Y is every other variable in data
  pluck("fit") |> Y is in {0,1}
  summary()
```

```
##
## Call:
## stats::glm(formula = default ~ ., family = stats::binomial, data =
## data)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.4691  -0.1418  -0.0557  -0.0203   3.7383
##
## Coefficients: β0, β1, β2, β3 SE(βi) H0: βi = 0 Ha: βi ≠ 0
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.087e+01  4.923e-01 -22.080 < 2e-16 ***
## studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 **
## balance      5.737e-03  2.319e-04  24.738 < 2e-16 ***
## income       3.033e-06  8.203e-06   0.370  0.71152 no significant relationship
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1571.5  on 9996  degrees of freedom
## AIC: 1579.5
##
## Number of Fisher Scoring iterations: 8
```

$\hat{\beta}_{\text{student(Yes)}} < 0 \Rightarrow$  if you are a student LESS likely to default holding balance & income constant.

Student confounded w/ income. (see "restricted regression" for more).

*dummy variable for categorical predictor*

By substituting estimates for the regression coefficients from the model summary, we can make predictions. For example, a student with a credit card balance of \$1,500 and an income of \$40,000 has an estimated probability of default of

$$\begin{aligned}\hat{p}(\underline{x} = \{\text{Yes}, 1500, 40000\}) &= \frac{e^{\text{exp}(-10.689 + (-0.6468) * 1 + 0.00574 * 1500 + 0.000003 * 40000)}}{1 + e^{\text{exp}(-10.689 + (-0.6468) * 1 + 0.00574 * 1500 + 0.000003 * 40000)}} \\ &= 0,058\end{aligned}$$

A non-student with the same balance and income has an estimated probability of default of

$$\hat{p}(\underline{x} = \{\text{No}, 1500, 40000\}) = 0,105$$

## 2.5 Logistic Regression for > 2 Classes

We sometimes wish to classify a response variable that has more than two classes. There are multi-class extensions to logistic regression ("multinomial regression"), but there are far more popular methods of performing this.

### 3 LDA "linear discriminant analysis"

Logistic regression involves 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:

Model the distribution of the predictors  $X$  separately in each of the response classes (given  $Y$ ). and then use Bayes theorem to flip base and get  $P(Y = k|X = x)$ .

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

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

1. Classes well separated, parameter estimates of predictors are surprisingly unstable
2.  $n$  is small ( $\frac{1}{2} X_i^{apx}$  Normal)  $\rightarrow$  LDA more stable
3. if we have  $> 2$  response classes

### 3.1 Bayes' Theorem for Classification

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

$Y$ -response ( $K$  distinct)

$\pi_k$   $\rightarrow$  prior  $\rightarrow$  overall probability that an observation is in class  $k$

$f_k(x)$   $\rightarrow$  "density of  $X$  in class  $k$ "

$\left\{ \begin{array}{l} P(X=x | Y=k) \rightarrow \text{discrete} \\ P(X \text{ in some small interval } | Y=k) \rightarrow \text{continuous} \end{array} \right.$

$$\frac{P(Y=k|X=x)}{P_k(x)} = \frac{P(X=x|Y=k) P(Y=k)}{P(X=x)} = \frac{f_k(x) \cdot \pi_k}{\sum_{l=1}^K \pi_l f_l(x)}$$

"posterior probability"

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

$\overline{\text{prior}}$

$$\pi_k = \frac{\# \text{ of obs. in class } k}{\text{Total } \# \text{ of observations}}$$

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

if we can estimate  $f_k(x) \rightarrow$  we can develop a classifier close to the "BEST" 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.

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

$$x \sim N(\mu_k, \sigma_k^2)$$

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

Assume for now  $\sigma_k^2 = \sigma^2$  for all  $k$

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

$$p_k(x) = P(Y=k|X=x) = \frac{\pi_k \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu_k)^2\right\}}{\sum_{l=1}^K \pi_l \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu_l)^2\right\}}$$

$\rightarrow f_k(x)$   
 prop of Numerator  
 (Total prob)  
 $\uparrow$  Forget to add this in obs

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

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

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

$\mu_1 > \mu_2$

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

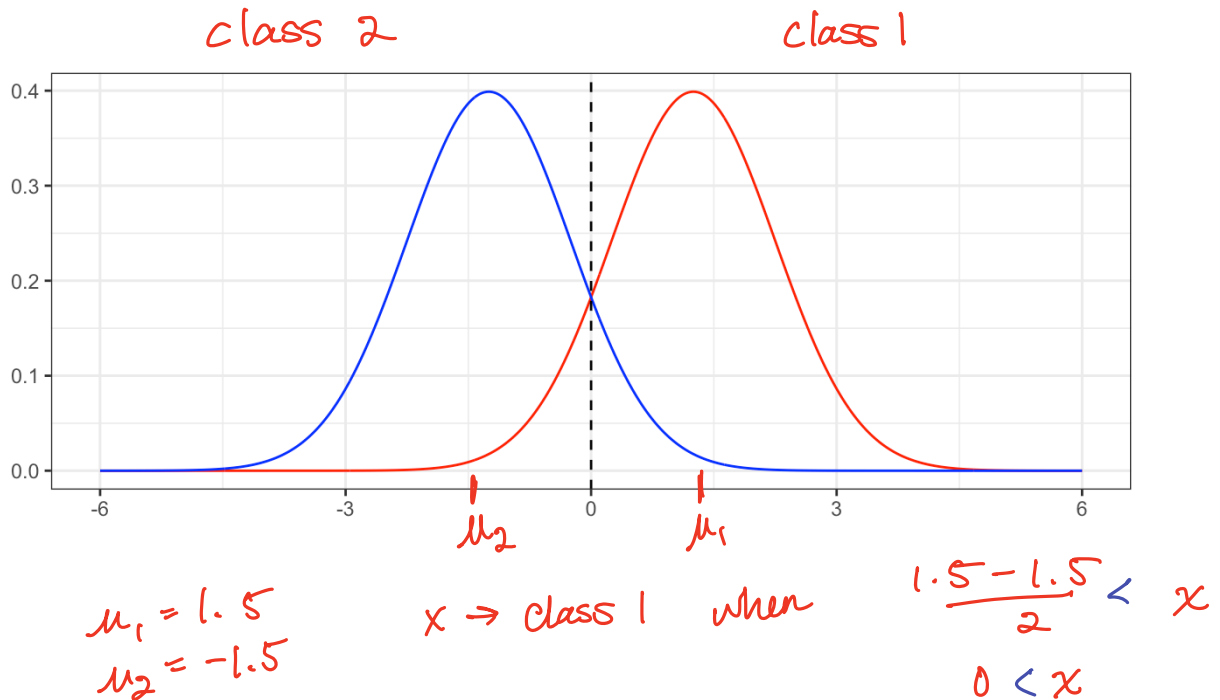
$$\cancel{x \frac{\mu_1}{\sigma^2} - \frac{\mu_1^2}{2\sigma^2} + \log(\pi_1)} > \cancel{x \frac{\mu_2}{\sigma^2} - \frac{\mu_2^2}{2\sigma^2} + \log(\pi_2)}$$

Note:  
if  $\mu_1 < \mu_2$   
 $x < \frac{\mu_1 + \mu_2}{2}$   
is the boundary

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

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

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



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

$$\mu_1, \dots, \mu_K, \pi_1, \dots, \pi_K, \sigma^2.$$

The *linear discriminant analysis* (LDA) method approximated the Bayes classifier by plugging estimates in for  $\pi_k, \mu_k, \sigma^2$ .

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i: y_i = k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-k} \sum_{i: y_i = k} \sum (x_i - \hat{\mu}_k)^2$$

weighted  
avg. of class variances

$n$  = Total # of observations  
 $n_k$  = # obs. in class  $k$

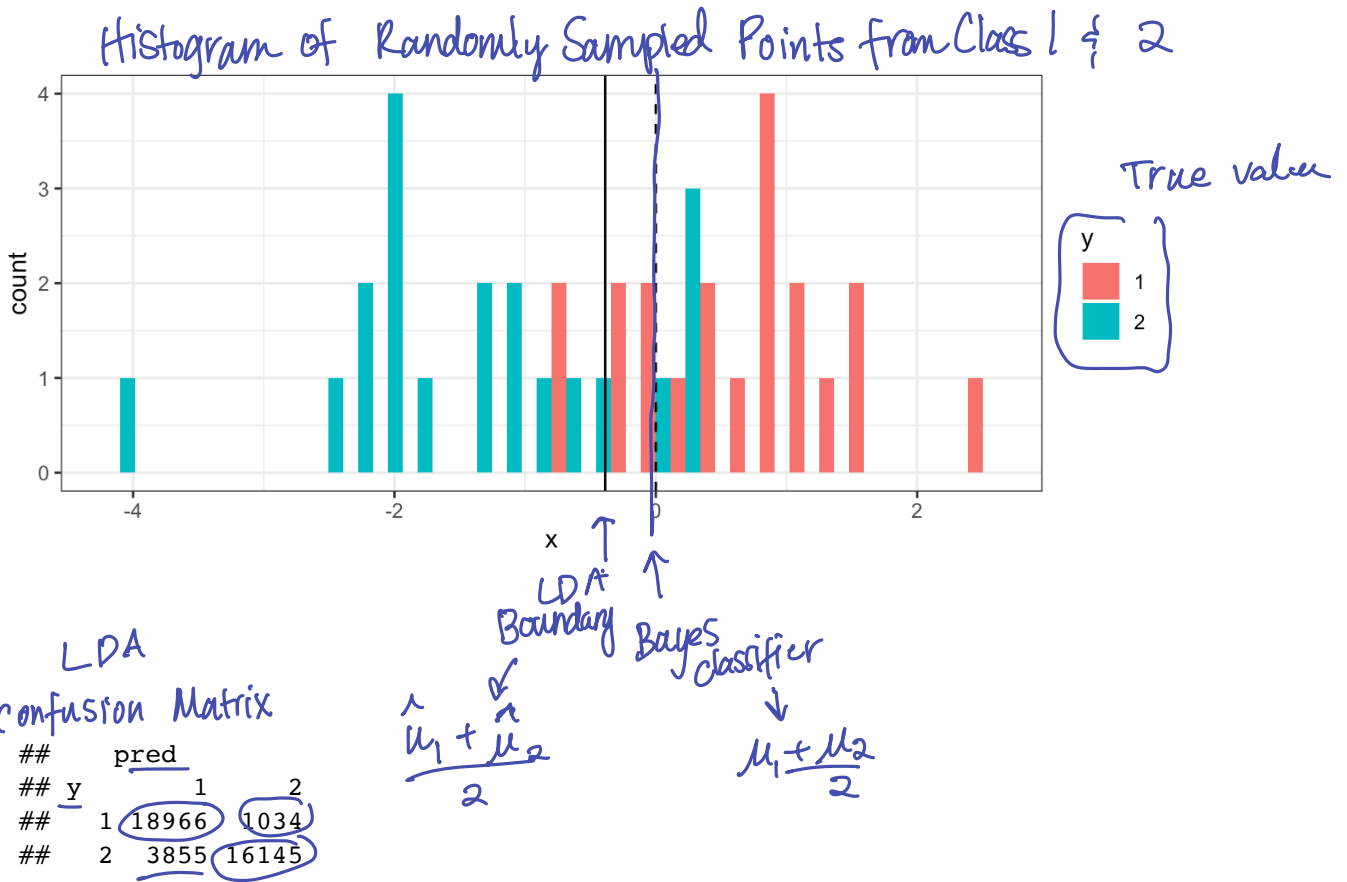
Sometimes we have knowledge of class membership probabilities  $\pi_1, \dots, \pi_K$  that can be used directly. If we do not, LDA estimates  $\pi_k$  using the proportion of training observations that belong to the  $k$ th class.

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

The LDA classifier assigns 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



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

$$\frac{1034 + 3855}{18966 + 1034 + 3855 + 16145} = \text{error}$$

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  $\sigma^2$  and plugging estimates for these parameters into the Bayes classifier.

↑  
we will relax this.

### 3.3 $p > 1$

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

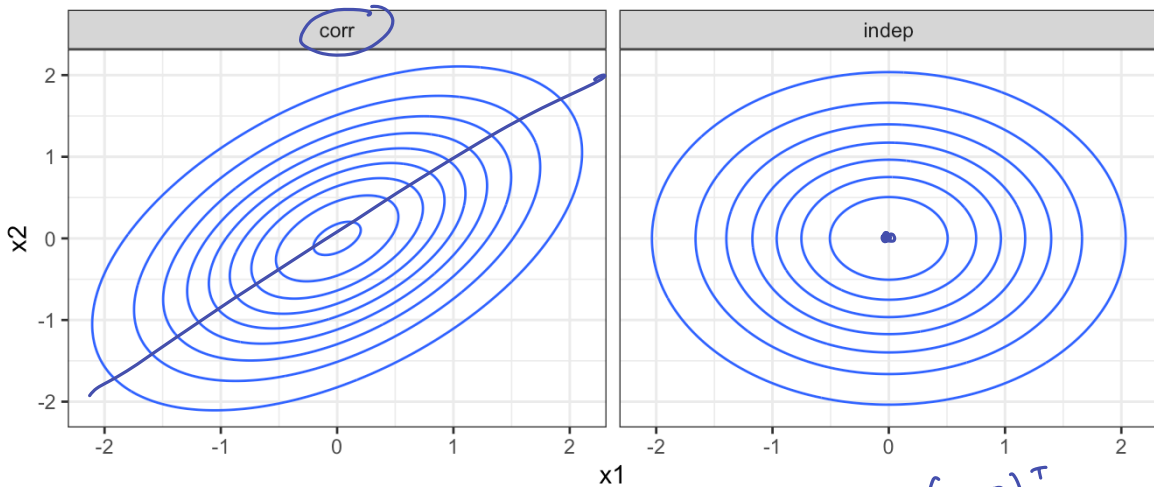
$$\underline{x} = (x_1, \dots, x_p) \sim \text{MVN}(\underline{\mu}, \Sigma)$$

Formally the multivariate Gaussian density is defined as

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

↑ "trace"  
↳ sum of elements on diagonal

$p=2$



$$\underline{\mu} = (0, 0)^T$$

$$\Sigma = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix}$$



In the case of  $p > 1$  predictors, the LDA classifier assumes the observations in the  $k$ th class are drawn from a multivariate Gaussian distribution  $N(\mu_k, \Sigma)$ .

Plugging in the density function for the  $k$ th class, results in a Bayes classifier

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

Once again, we need to estimate the unknown parameters  $\mu_1, \dots, \mu_K, \pi_1, \dots, \pi_K, \Sigma$ .

$$\hat{\delta}_k(x) = x^T \hat{\Sigma}^{-1} \hat{\mu}_k - \frac{1}{2} \hat{\mu}_k^T \hat{\Sigma}^{-1} \hat{\mu}_k + \log \hat{\pi}_k$$

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

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.

```
lda_spec <- discrim_linear(engine = "MASS")
```

```
lda_fit <- lda_spec |> y ~ x
  fit(default ~ student + balance, data = Default)
```

```
lda_fit |>
  pluck("fit")
```

*specify formula just like linear regression or logistic regression*

```
## Call:
## lda(default ~ student + balance, data = data)
```

```
## Prior probabilities of groups:
```

```
##      No      Yes
## 0.9667 0.0333
```

*← estimates of  $\pi_k$  based on class membership in training data.*

```
## Group means:
```

```
##      studentYes  balance
## No (0.2914037, 803.9438)
## Yes (0.3813814, 1747.8217)
```

*average of each predictor  
within each class, use the estimate  $\mu_k$*

```
## Coefficients of linear discriminants:
```

```
##              LD1
## studentYes -0.249059498
## balance    0.002244397
```

\*

*← linear combinations of student and balance  
used in the LDA decision rule.*

Confusion matrix for training data.

```
# training data confusion matrix
lda_fit |>
  augment(new_data = Default) |>
  conf_mat(truth = default, estimate = .pred_class)
```

| ## |            | Truth |     |
|----|------------|-------|-----|
| ## | Prediction | No    | Yes |
| ## | No         | 9644  | 252 |
| ## | Yes        | 23    | 81  |

could instead put test data here.

For Default = Yes, only got  $\frac{81}{252+81}$  right

overall training accuracy  $\frac{9644+81}{9644+81+252+23} = 1 - .0275 = 24\%$

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

only 3.33% of individuals in the training data set defaulted!

A simple (but use less) classifier could just predict default = NO.  $\uparrow$  get only 3.33% error rate.

LDA is trying to approximate the Bayes classifier  $\Rightarrow$  yields smallest possible overall error rate.

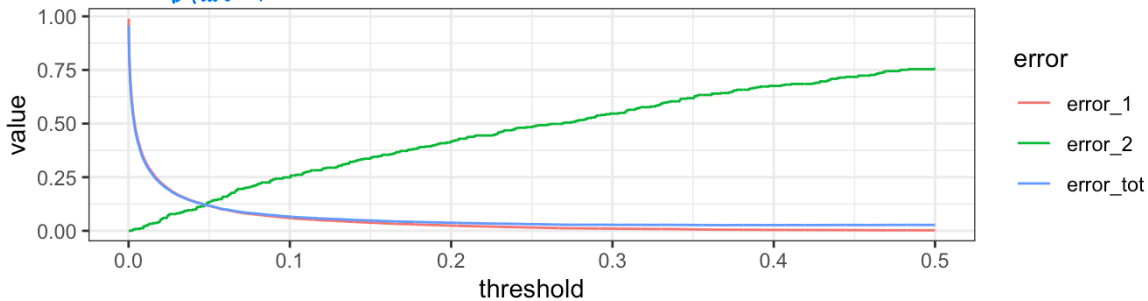
A CC company may want to avoid misclassifying default = YES, so we can adjust how we select classes.

```
lda_fit |>
  augment(new_data = Default) |>
  mutate(pred_lower_cutoff = factor(ifelse(.pred_Yes > 0.2, "Yes", "No"))) |>
  conf_mat(truth = default, estimate = pred_lower_cutoff)
```

predicted probability of defaulting.

| ## |            | Truth |     |
|----|------------|-------|-----|
| ## | Prediction | No    | Yes |
| ## | No         | 9432  | 138 |
| ## | Yes        | 235   | 195 |

do worse for Default = NO  $\leftarrow$  do better at Default = YES



As I increase threshold, error (default = YES)  $\uparrow$  (green line)  
 error (default = NO)  $\downarrow$  (red line)

How to choose? Domain knowledge (CV next week), 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  $k^{\text{th}}$  class  $X \sim \text{MVN}(\mu_k, \Sigma_k)$   
 $\Sigma_k$  ~ cov matrix for  $k^{\text{th}}$  class.

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

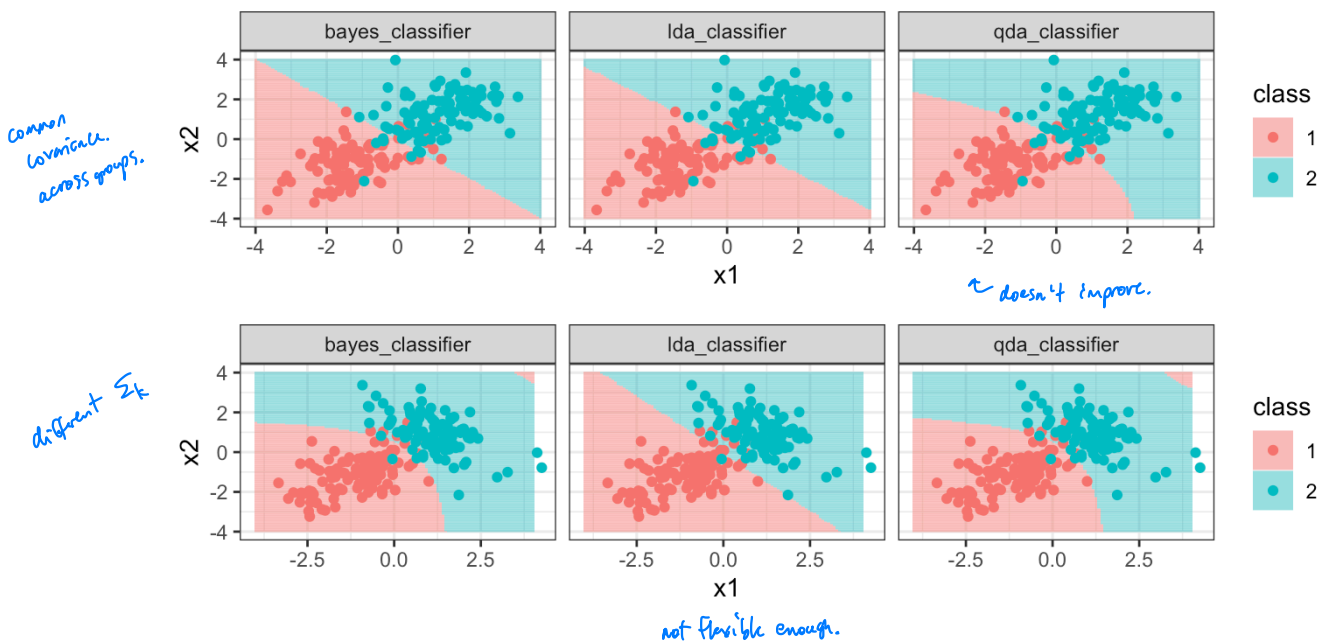
$$\delta_k(x) = -\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) - \frac{1}{2} \log |\Sigma_k| + \log \pi_k$$

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

quadratic in  $x \Rightarrow$  quadratic discriminant analysis

plug in estimates for  $\Sigma_k, \mu_k, \pi_k$   
 and choose  $\underset{k}{\text{argmax}} \delta_k(x)$

When would we prefer QDA over LDA?



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

LDA is linear in  $x \Rightarrow$  only  $K \cdot p$  parameters to estimate.

$\Rightarrow$  LDA is much less flexible than QDA, but if global variance assumption is bad, LDA might be wildly off  $\Rightarrow$  QDA?

$\Rightarrow$  LDA  $>$  QDA if not many training obs.

# 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  $P(Y=k|X=x)$  as

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

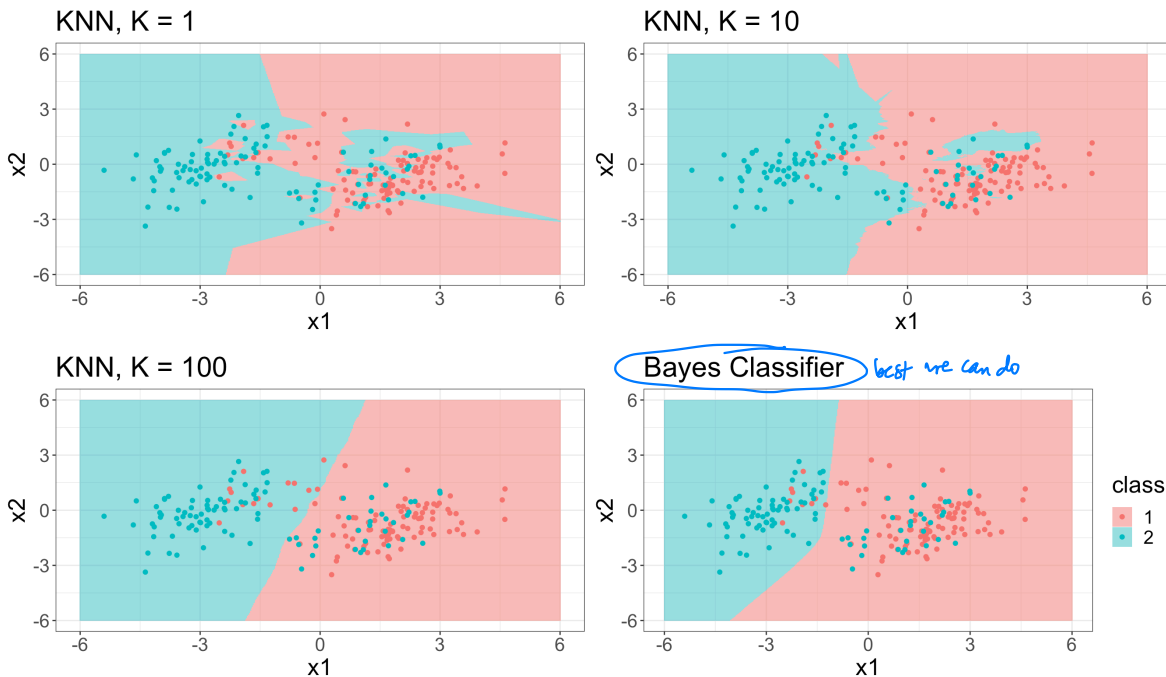
# points in neighborhood.      class k

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

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  
close to linear boundary.

- choosing the correct level of flexibility is critical to success of any method (KNN + LDA vs. QDA).
- = How to choose? ch. 5!

## 5 Comparison

LDA vs. Logistic Regression

LDA and Logistic Regression are closely related. Consider  $k=2$ ,  $p=1$ , and  $p_1(x)$ ,  $p_2(x) = 1 - p_1(x)$  are probability observation  $X=x$  belongs to class 1 or 2.

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

LDA:  $\log\left(\frac{p_1(x)}{1-p_1(x)}\right) = \log\left(\frac{p_1(x)}{p_2(x)}\right) = \log\left[\frac{\pi_1}{\pi_2} \exp\left(-\frac{1}{2\sigma^2}\{(x-\mu_1)^2 - (x-\mu_2)^2\}\right)\right]$   
 $= \log \pi_1 - \log \pi_2 - \frac{1}{2\sigma^2} [\cancel{x^2} - 2x\mu_1 + \mu_1^2 - \cancel{x^2} + 2x\mu_2 - \mu_2^2] = C_0 + C_1 x$  (linear function in  $x$ ).  
fit using plug-in estimates of  $\mu_1, \mu_2, \sigma$

should give similar results, but LDA assume Gaussian dist w/ common variance } whichever holds  
 Logistic regression does not. } should be "better".

(LDA & Logistic Regression) vs. KNN

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

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

KNN doesn't tell us which parameters are important (have relationship w/ the predictor).

QDA

Compromise between non-parametric KNN & linear LDA, logistic regression

Quadratic decision boundary  $\Rightarrow$  can accurately model a wider range of problems than linear methods.

Not as flexible as KNN  $\Rightarrow$  for problems w/ less training data can have improvements for prediction on test data.