Types of Errors

		True class		
		− or Null	+ or Non-null	Total
Predicted class	− or Null	True Neg. (TN)	False Neg. (FN)	N*
	+ or Non-null	False Pos. (FP)	True Pos. (TP)	P*
	Total	N	P	

Possible results when applying a classifier or diagnostic test to a population.

False positive rate (Type I error) $= \frac{FP}{N} = 1 - \frac{TN}{N} = 1 -$ specificity

(in our example = 0.2%)

specificity $= \frac{TN}{N}$ (in our example = 99.8%)

False negative rate (Type II error) $= \frac{FN}{P} = 1 -$ sensitivity

(in our example = 75.7%)

sensitivity $= \frac{TP}{P}$ (in our example = 24.3%)

Can we change the two error rates?

We found our previous results by classifying to class Yes if

$$\widehat{\Pr}(\text{default} = \text{Yes}|X = x) \geq 0.5$$

We can change the two error rates by changing the threshold from 0.5 to some other value in [0, 1]:

$$\widehat{\Pr}(\text{default} = \text{Yes}|X = x) \geq \textit{threshold}$$

and vary threshold.

For instance, we might label any customer with a posterior probability of default above 20% to the default class.

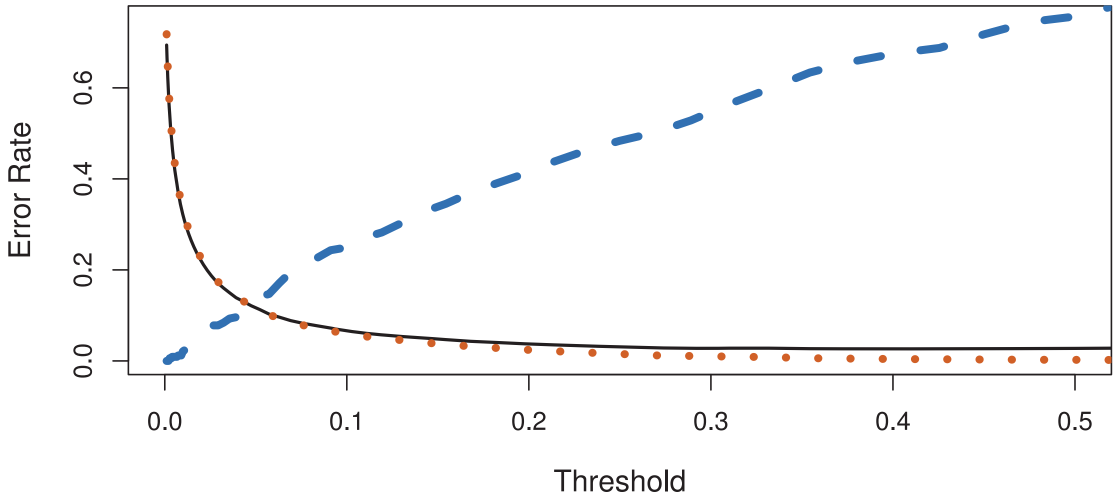
```
> lda_pred_def_2 <- rep("No", 10000)
> lda_pred_def_2[lda_pred_def_1$posterior[, 2] >= 0.2]  = "Yes"
> table(lda_pred_def_2, Default$default)

lda_pred_def_2   No  Yes
           No  9432  138
           Yes  235  195
```

		True default status		
		No	Yes	Total
Predicted	No	9432	138	9570
default status	Yes	235	195	430
	Total	9667	333	10000

A confusion matrix compares the LDA predictions to the true default statuses for the 10,000 training observations in the Default data set, using a modified threshold value that predicts default for any individuals whose posterior default probability exceeds 20%.

Threshold	Type I err.	Type II err.	overall err.
0.5	0.2%	75.7%	2.75%
0.2	2.43%	41.4%	3.73%



For the Default data set, error rates are shown as a function of the threshold value for the posterior probability that is used to perform the assignment. The black solid line displays the overall error rate. The blue dashed line represents the fraction of defaulting customers that are incorrectly classified, and the orange dotted line indicates the fraction of errors among the non-defaulting customers.

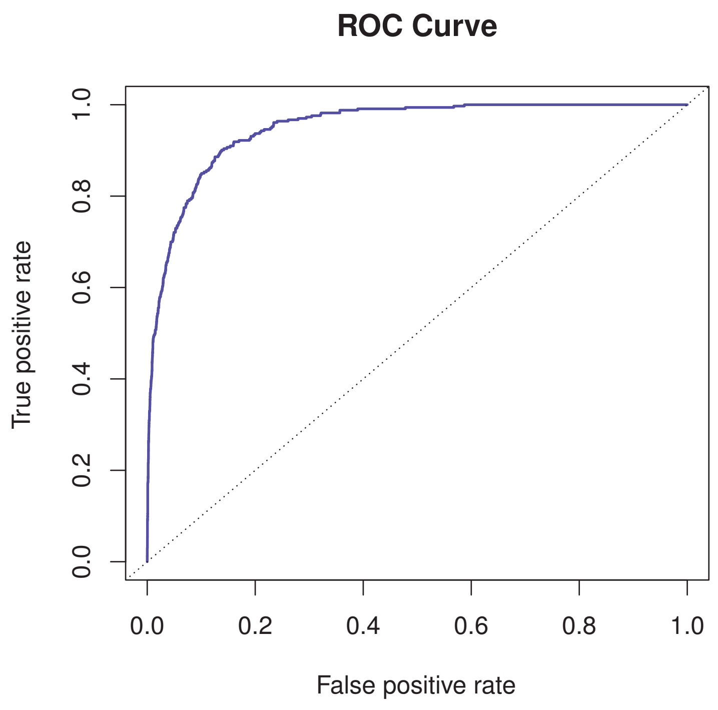
As the threshold is reduced, the error rate among individuals who default decreases steadily, but the error rate among the individuals who do not default increases.

How can we decide which threshold value is best?

such a decision must be based on domain knowledge,

such as detailed information about the costs associated with default.

The ROC plot displays both errors simultaneously.



A ROC curve for the LDA classifier on the Default data. It traces out two types of error as we vary the threshold value for the posterior probability of default. The ideal ROC curve hugs the top left corner, indicating a high true positive rate and a low false positive rate. The dotted line represents the "no information" classifier; this is what we would expect if student status and credit card balance are not associated with probability of default.

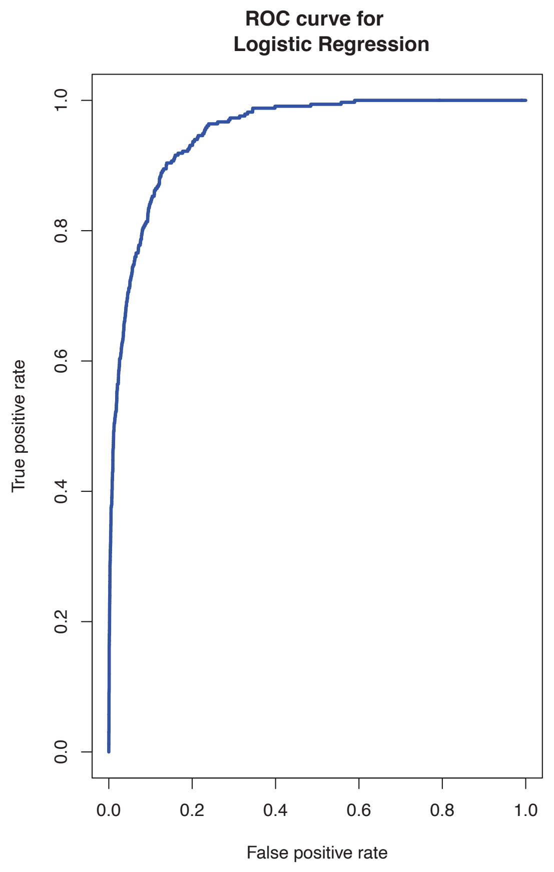
Sometimes we use the AUC or area under the curve to summarize the overall performance.

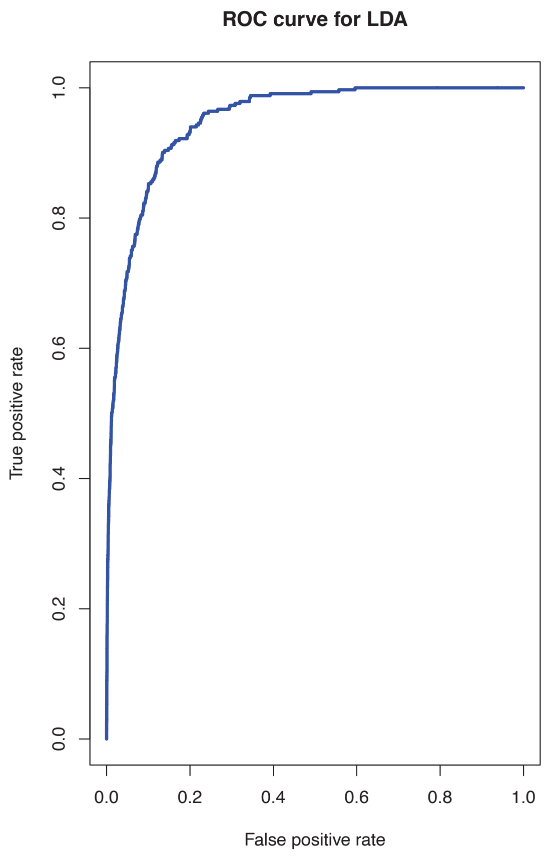
The larger the AUC the better the classifier.

The ROCR package can be used to produce ROC curves. Here we find the ROC curve and AUC for the Logistic Regression and the LDA classifiers on the Default data.

```
> library(ROCR)
> pred_glm <- prediction(glm_probs_def_4, Default$default)
> pred_lda <- prediction(lda_pred_def_1$posterior[, 2], Default$default)
> perf_glm <- performance(pred_glm,"tpr","fpr")
> perf_lda <- performance(pred_lda,"tpr","fpr")
> par(mfrow = c(1, 2))
> plot(perf_glm, col="blue", lwd = 3, main="ROC curve for
+      Logistic Regression")
> plot(perf_lda, col="blue", lwd = 3, main="ROC curve for LDA")
> auc_tmp_glm <- performance(pred_glm,"auc")
> auc_tmp_glm@y.values
[[1]]
[1] 0.9495581

> auc_tmp_lda <- performance(pred_lda,"auc")
> auc_tmp_lda@y.values
[[1]]
[1] 0.9495584
```





The ROC curves for the Logistic Regression classifier (left) and the LDA classifier (right) on the Default data.

The AUC for both models on the default data set is 0.95, which is close to the maximum of one so would be considered very good.

Quadratic Discriminant Analysis

- Like LDA, the QDA classifier results from assuming that the observations from each class are drawn from a Gaussian distribution, and plugging estimates for the parameters into Bayes' theorem in order to perform prediction.

- Unlike LDA, QDA assumes that each class has its own covariance matrix. That is, it assumes that an observation from the kth class is of the form $X \sim N(\mu_k, \mathbf{\Sigma}_k)$, where $\mathbf{\Sigma}_k$ is a covariance matrix for the kth class.

- In this case, the discriminant function has the following form:

a quadratic function in x $\longrightarrow$

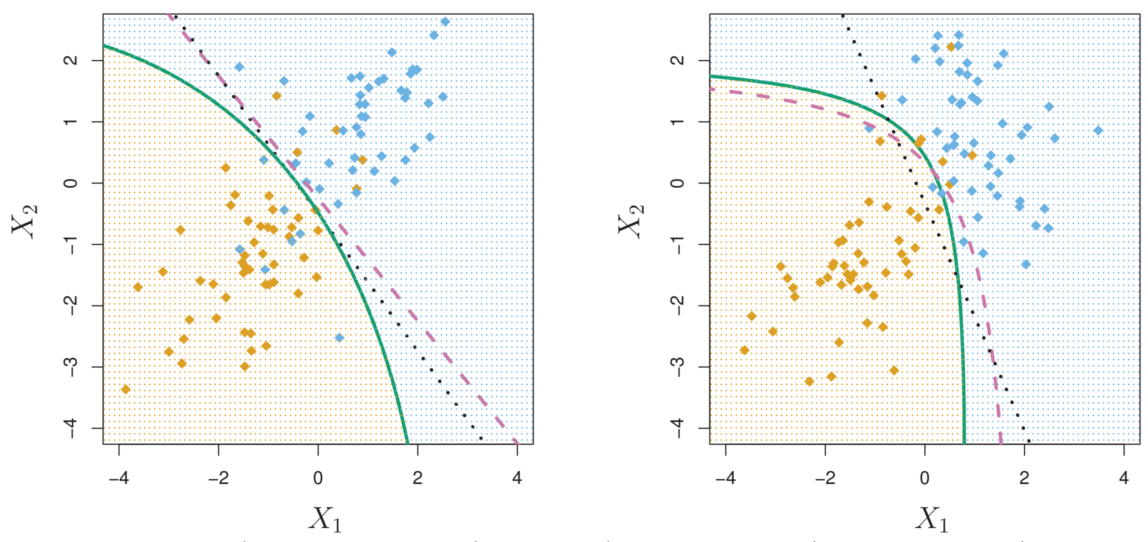
$$\delta_k(x) = -\frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}_k^{-1}(x-\mu_k) - \frac{1}{2}\log|\mathbf{\Sigma}_k| + \log \pi_k$$

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

and then x is assigned to the class with the largest discriminant score.

- Like before, we need to estimate the unknown parameters $\mu_1, \ldots, \mu_K$, $\pi_1, \ldots, \pi_K$, and $\mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_K$.

- To assign a new observation $X = x$, QDA plugs the estimates into $\delta_k(x)$ to obtain the quantity $\hat{\delta}_k(x)$, and classifies to the class for which $\hat{\delta}_k(x)$ is largest.



Left: The Bayes (purple dashed), LDA (black dotted), and QDA (green solid) decision boundaries for a two-class problem with $\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2$. The shading indicates the QDA decision rule. Since the Bayes decision boundary is linear, it is more accurately approximated by LDA than by QDA. Right: Details are as given in the left-hand panel, except that $\mathbf{\Sigma}_1 \neq \mathbf{\Sigma}_2$. Since the Bayes decision boundary is non-linear, it is more accurately approximated by QDA than by LDA.

Now we will perform QDA on the Default data in order to predict whether or not an individual will default on the basis of credit card balance and student status.

In R, we fit a QDA model using the qda() function, which is part of the MASS library. The syntax is identical to that of lda().

```
> library(ISLR)
> names(Default)
[1] "default" "student" "balance" "income"
> library(MASS)
> qda_fit_def_1 <- qda(default ~ balance + student, data = Default)
> qda_fit_def_1
Call:
qda(default ~ balance + student, data = Default)
```

```
Prior probabilities of groups:
    No    Yes
```
$\hat{\pi}_1 \rightarrow$ `0.9667 0.0333` $\leftarrow \hat{\pi}_2$

```
Group means:
      balance studentYes
No   803.9438  0.2914037
Yes 1747.8217  0.3813814

> qda_pred_def_1 <- predict(qda_fit_def_1)
> names(qda_pred_def_1)
[1] "class"     "posterior"

> table(qda_pred_def_1$class, Default$default)

        No  Yes
  No  9637  244
  Yes   30   89
> mean(qda_pred_def_1$class == Default$default)
[1] 0.9726
> mean(qda_pred_def_1$class != Default$default)
```
`[1] 0.0274` $\leftarrow$ overall err. rate or training err. rate

Naive Bayes

- The naive Bayes classifier takes a different tack for estimating $f_1(x), ..., f_K(x)$ in

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

- Instead of assuming that these functions belong to a particular family of distributions (e.g. multivariate normal), we instead make a single assumption:

Within the kth class, the p predictors are independent.