IG.3510-Machine Learning
Lecture 2: Supervised learning: Classification
Part I

Dr. Patricia CONDE-CESPEDES

patricia.conde-cespedes@isep.fr

September 16th, 2024

Plan

1. Introduction
2. K-Nearest Neighbors
3. Logistic Regression
4. Linear Discriminant Analysis
5. Other forms of Discriminant Analysis
6. Evaluating the quality of the predictions
7. A Comparison of Classification Methods

Outline

1. Introduction
2. K-Nearest Neighbors
3. Logistic Regression
4. Linear Discriminant Analysis
5. Other forms of Discriminant Analysis
6. Evaluating the quality of the predictions
7. A Comparison of Classification Methods

Introduction to classification

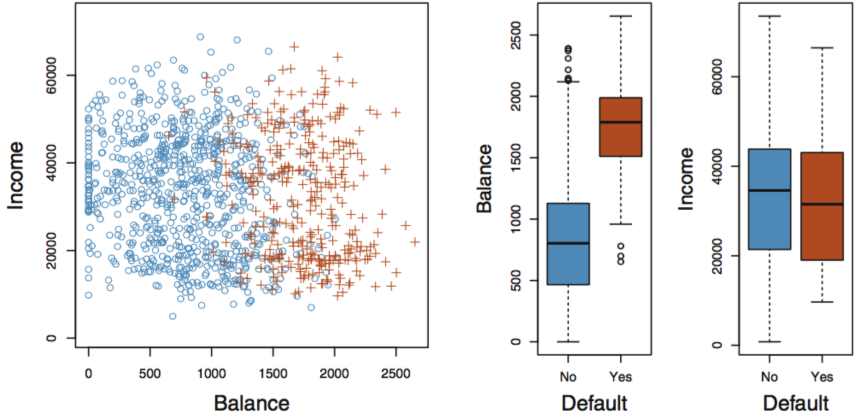
- In classification, the response Y is a **qualitative** or **categorical** variable. Some examples:
  - In the *Spam detection* problem the target can take only two values $\{"Spam", "mail"\}$.
  - If the variable is *"Origin"*, then, the target variable takes more than two labels $\{"American", "Asian", "African", "European"\}$.
- The goal is:

Classification goal

Given feature vectors $\mathbf{X}$ and a qualitative response Y, the goal is to build a classifier function $C(\mathbf{X})$ that takes as input $\mathbf{X}$ and predicts its value for Y.

Example

Goal: predict whether an individual will default on his/her credit card payment, on the basis of annual income and monthly credit card balance.



individuals who defaulted (orange) and those who did not (blue).

Y: default credit payment based on *balance* X_1 and income X_2.

Training error and *test error* in classification

- **Training error rate:** the proportion of misclassified observations in the training set:

$$\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}(y_i \neq \hat{y}_i)$$

where $\hat{y}_i$ is the predicted class by our classifier for the ith observation and y_i is the real value.

Training error and *test error* in classification

- **Training error rate:** the proportion of misclassified observations in the training set:

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(y_i \neq \hat{y}_i)$$

  where $\hat{y}_i$ is the predicted class by our classifier for the ith observation and y_i is the real value.

- **Test error rate:** . For a given observation (x_0, y_0), a good classifier will have minimum estimated error test:

$$\text{average}(\mathbb{1}(y_0 \neq \hat{y}_0))$$

The Bayes Classifier

In practice, we estimate the conditional probability of **Y** given **X**:

Suppose **Y** has κ categories numbered $\{1, 2, ..., \kappa\}$. Then, we want to estimate:

$$p_k(x) = P(Y = k | X = x), \qquad k = 1, 2, ..., \kappa.$$

$p_k(x)$ is the conditional probability of class k at value x.

The Bayes Classifier

In practice, we estimate the conditional probability of **Y** given **X**:

Suppose **Y** has κ categories numbered $\{1, 2, ..., \kappa\}$. Then, we want to estimate:

$$p_k(x) = P(Y = k|X = x), \qquad k = 1, 2, ..., \kappa.$$

$p_k(x)$ is the conditional probability of class k at value x.

The test error rate is minimized, on average, if given an observation, it is *assigned to the most likely class*.

Such a classifier is called the **Bayes classifier**:

$$C(x) = j \text{ if } p_j(x) = \max\{p_1(x), p_2(x), ..., p_\kappa(x)\}$$

Bayes error rate in a two-class problem

If there are only 2 classes, the Bayes classifier will choose the class j for which:

$$P(Y = j|X = x_0) > 0.5$$

Then, the **Bayes error rate** will be:

$$1 - E(\max_j P(Y = j|X = x_0))$$

In practical applications, we do not know the conditional distribution of Y given X. Then, we will have to estimate it!

Outline

1 Introduction

2 K-Nearest Neighbors

3 Logistic Regression

4 Linear Discriminant Analysis

5 Other forms of Discriminant Analysis

6 Evaluating the quality of the predictions

7 A Comparison of Classification Methods

The K-nearest neighbors (KNN) classifier

Given a positive integer K and a test observation x_0, the KNN classifier proceeds as follows:

1. first identifies the K points in the training data that are closest to x_0, represented by $\mathcal{N}_0$.

The K-nearest neighbors (KNN) classifier

Given a positive integer K and a test observation x_0, the *KNN* classifier proceeds as follows:

1. first identifies the K points in the training data that are closest to x_0, represented by $\mathcal{N}_0$.
2. It then estimates the conditional probability for class j as follows:

$$\hat{p}_j(x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} \mathbb{1}(y_i = j)$$

that is, the fraction of points in $\mathcal{N}_0$ whose response values are j.

The K-nearest neighbors (KNN) classifier

Given a positive integer K and a test observation x_0, the *KNN* classifier proceeds as follows:

1. first identifies the K points in the training data that are closest to x_0, represented by $\mathcal{N}_0$.
2. It then estimates the conditional probability for class j as follows:

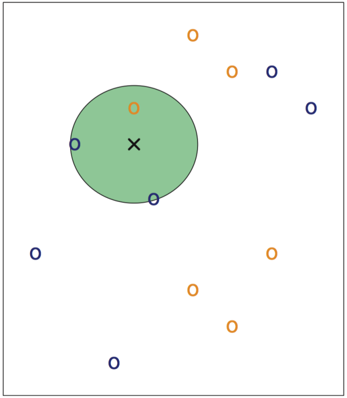
$$\hat{p}_j(x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} \mathbb{1}(y_i = j)$$

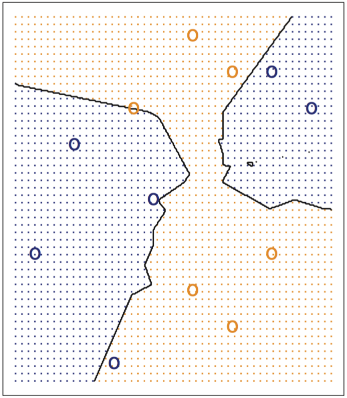
   that is, the fraction of points in $\mathcal{N}_0$ whose response values are j.

Finally, *KNN* applies the Bayes rule to classify the test observation x_0.

KNN small example

Training data set consisting of six blue and six orange observations.
Goal: make a prediction for the point labeled by the black cross.
KNN for $K = 3$,

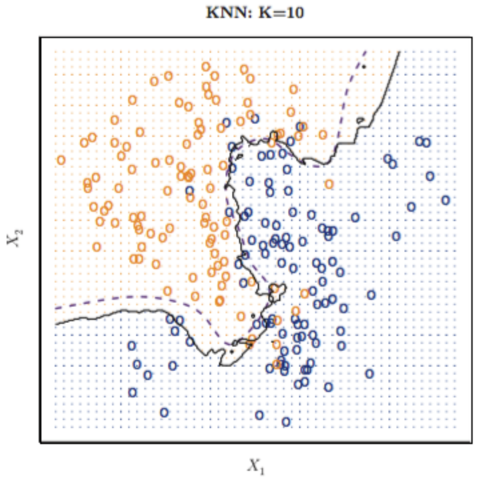




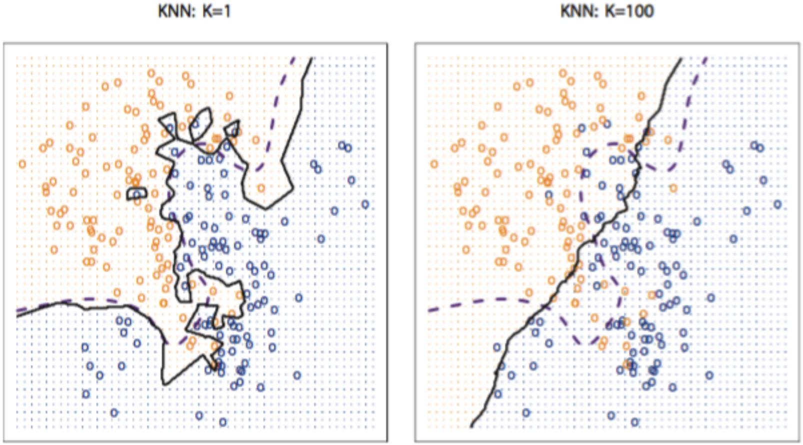
On the right, *KNN* for $\text{K} = 3$ applied at every point (the test set) and the corresponding *KNN decision boundary*.

KNN classifier: simple but good!

KNN can often produce classifiers that are surprisingly close to the optimal Bayes classifier. The purple dashed line represents the Bayes decision boundary.



KNN:The value of k and the flexibility of the model



KNN: The flexibility decreases with the value of K.

Outline

1 Introduction

2 K-Nearest Neighbors

3 Logistic Regression

4 Linear Discriminant Analysis

5 Other forms of Discriminant Analysis

6 Evaluating the quality of the predictions

7 A Comparison of Classification Methods

Motivation with binary classification

Let us suppose we want to predict the `Marital status`. Then, we have two levels and we can use the binary coding:

$$Y = \begin{cases} 0 : \texttt{No} \\ 1 : \texttt{Yes} \end{cases}$$

Motivation with binary classification

Let us suppose we want to predict the `Marital status`. Then, we have two levels and we can use the binary coding:

$$Y = \begin{cases} 0 : \texttt{No} \\ 1 : \texttt{Yes} \end{cases}$$

We want to estimate a **probability** $E(Y|X = x) = P(Y = 1|X = x)$ (because Y is an indicator variable).
What if we perform linear regression?

Linear regression with a two-level output

Let us suppose we have only one predictor X, then we want to estimate $p(X) = P(Y = 1|X)$ using a linear regression model:

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

and classify as "`1:Yes`" if $\hat{p} > 0.5$.

Linear regression with a two-level output

Let us suppose we have only one predictor X, then we want to estimate $p(X) = P(Y = 1|X)$ using a <u>linear regression</u> model:

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

and classify as "`1:Yes`" if $\hat{p} > 0.5$.
However, this model has some drawbacks:

- Why not reverse coding $\{0$:`Yes`, 1:`No`$\}$? And the fit would be different!

Linear regression with a two-level output

Let us suppose we have only one predictor X, then we want to estimate $p(X) = P(Y = 1|X)$ using a linear regression model:

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

and classify as "`1:Yes`" if $\hat{p} > 0.5$.
However, this model has some drawbacks:

- Why not reverse coding $\{0$:`Yes`, 1:`No`$\}$? And the fit would be different!
- Linear regression might produce probability estimates falling outside the interval $[0, 1]$.

Linear regression with a two-level output

Let us suppose we have only one predictor X, then we want to estimate $p(X) = P(Y = 1|X)$ using a linear regression model:

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

and classify as "`1:Yes`" if $\hat{p} > 0.5$.
However, this model has some drawbacks:

- Why not reverse coding {0:`Yes`, 1:`No`}? And the fit would be different!
- Linear regression might produce probability estimates falling outside the interval $[0, 1]$.
- The predictions provide an ordering.

Solution: the logistic function

The logistic function gives outputs between 0 and 1.

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

($e \approx 2.71828$ is the Euler's number.)

No matter what values β_0, β_1 or X take, $p(X)$ will always lie between 0 and 1.

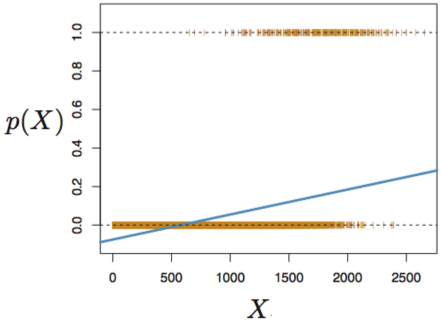
Linear versus Logistic Regression

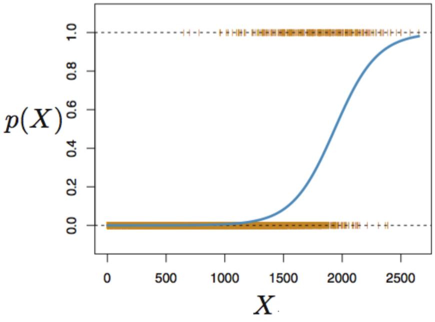
Linear regression

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

Logistic regression

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





In orange the observations, in blue, the fitted curve for each model.

For logistic regression, when $y = 0$, $p(X)$ takes low values, whereas for $y = 1$, $p(X)$ takes high values.

log odds or *logit* transformation of $p(X)$

Rewriting The logistic regression function is equivalent to:

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

The quantity $p(X)/(1 - p(X))$ is called the **odds**.

log odds or *logit* transformation of $p(X)$

Rewriting The logistic regression function is equivalent to:

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

The quantity $p(X)/(1 - p(X))$ is called the **odds**.
Interpretation:The odds is the ratio between $P(Y = 1/X)$ and $P(Y = 0/X)$

log odds or *logit* transformation of $p(X)$

Rewriting The logistic regression function is equivalent to:

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

The quantity $p(X)/(1-p(X))$ is called the **odds**.
Interpretation:The odds is the ratio between $P(Y=1/X)$ and $P(Y=0/X)$

By taking the logarithm of both sides, we get:

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

The left-hand side is called the **log-odds** or **logit**.

log odds or *logit* transformation of $p(X)$

Rewriting The logistic regression function is equivalent to:

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

The quantity $p(X)/(1-p(X))$ is called the **odds**.
Interpretation:The odds is the ratio between $P(Y=1/X)$ and $P(Y=0/X)$

By taking the logarithm of both sides, we get:

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

The left-hand side is called the **log-odds** or **logit**.
Interpretation increasing X by one unit changes the log odds by β_1, or equivalently it multiplies the odds by e^{β_1}.
if $\beta_1 > 0$ then increasing X will be associated with increasing $p(X)$, and
if $\beta_1 < 0$ then increasing X will be associated with decreasing $p(X)$.

Estimating the coefficients in logistic regression

Y is a Bernoulli random variable as it can take only two values:
$P(Y = 1|X) = p(x)$ and $P(Y = 0|X) = (1 - p(x))$.

Estimating the coefficients in logistic regression

Y is a Bernoulli random variable as it can take only two values: $P(Y=1|X)=p(x)$ and $P(Y=0|X)=(1-p(x))$.

Suppose we have a random (train) sample of size n: $(y_1, x_1), \dots, (y_n, x_n)$, the joint probability of observing the n values of Y is given as:

$$\prod_{i=1}^{n} p(x_i)^{y_i}(1-p(x_i))^{1-y_i} \tag{1}$$

We suppose the observations are independently distributed.

The joint probability distribution is known in statistics as **likelihood** function and will be denoted $\ell(.)$:

$$\ell(\beta_0, \beta_1) = \prod_{y_i=1} p(x_i) \prod_{y_i=0}(1-p(x_i)) = \prod_{y_i=1} \frac{e^{\beta_0+\beta_1 X_i}}{1+e^{\beta_0+\beta_1 X_i}} \prod_{y_i=0}\left(1-\frac{e^{\beta_0+\beta_1 X_i}}{1+e^{\beta_0+\beta_1 X_i}}\right)$$

We estimate β_0 and β_1 that **maximize** the likelihood function.

Example with the `Credit` data

Considered the `Credit` dataset, we want to predict the `default` of a customer (pay or not) according to the `balance`.
The parameter estimates are $\hat{\beta}_0$ and $\hat{\beta}_1$.

	Coefficient	Std. Error	Z-statistic	P-value
`Intercept`	-10.6513	0.3612	-29.5	< 0.0001
`balance`	0.0055	0.0002	24.9	< 0.0001

- their standard errors measure the accuracy of the coefficient.
- The *z-statistic* plays the same role as the *t-statistic* in the linear regression output.

  A small *p-value* implies that there is an association between `balance` and probability of `default`.

Making predictions

What is our estimated probability of default for someone with a balance of \$1000?

Making predictions

What is our estimated probability of default for someone with a balance of \$1000?

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0+\hat{\beta}_1 X}}{1+e^{\hat{\beta}_0+\hat{\beta}_1 X}} = \frac{e^{-10.6513+0.0055\times 1{,}000}}{1+e^{-10.6513+0.0055\times 1{,}000}} = 0.006$$

Making predictions

What is our estimated probability of default for someone with a balance of \$1000?

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 X}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 X}} = \frac{e^{-10.6513 + 0.0055 \times 1{,}000}}{1 + e^{-10.6513 + 0.0055 \times 1{,}000}} = 0.006$$

With a balance of \$2000?

Making predictions

What is our estimated probability of default for someone with a balance of \$1000?

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 X}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 X}} = \frac{e^{-10.6513 + 0.0055 \times 1{,}000}}{1 + e^{-10.6513 + 0.0055 \times 1{,}000}} = 0.006$$

With a balance of \$2000?

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 X}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 X}} = \frac{e^{-10.6513 + 0.0055 \times 2{,}000}}{1 + e^{-10.6513 + 0.0055 \times 2{,}000}} = 0.586$$

Logistic regression with qualitative predictors

We can be interested in predicting `default` based on a categorical variable, for instance the `student` status.

	Coefficient	Std. error	Z-statistic	P-value
`Intercept`	-3.5041	0.0707	-49.55	<0.0001
`student[Yes]`	0.4049	0.1150	3.52	0.0004

What is the estimated probability of defaulting for a student (1: Yes, 0:Not)?

Logistic regression with qualitative predictors

We can be interested in predicting `default` based on a categorical variable, for instance the `student` status.

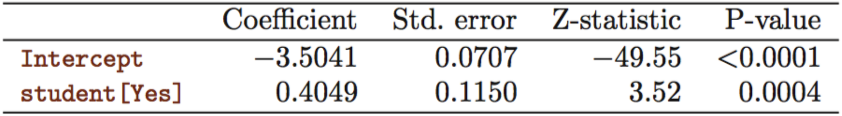
	Coefficient	Std. error	Z-statistic	P-value
`Intercept`	−3.5041	0.0707	−49.55	<0.0001
`student[Yes]`	0.4049	0.1150	3.52	0.0004

What is the estimated probability of defaulting for a student (1: Yes, 0:Not)?

$$\hat{p}(X) = \hat{p}(\texttt{default=Yes}|x = \texttt{student}) = \frac{e^{-3.5041+0.4049\times 1}}{1 + e^{-3.5041+0.4049\times 1}} = 0.0431$$

Logistic regression with qualitative predictors

We can be interested in predicting `default` based on a categorical variable, for instance the `student` status.



	Coefficient	Std. error	Z-statistic	P-value
`Intercept`	−3.5041	0.0707	−49.55	<0.0001
`student[Yes]`	0.4049	0.1150	3.52	0.0004

What is the estimated probability of defaulting for a student (1: Yes, 0:Not)?

$$\hat{p}(X) = \hat{p}(\texttt{default=Yes}|x = \texttt{student}) = \frac{e^{-3.5041+0.4049\times 1}}{1+e^{-3.5041+0.4049\times 1}} = 0.0431$$

What about for an individual who is not a student?

$$\hat{p}(X) = \hat{p}(\texttt{default=Yes}|x = \texttt{non student}) = \frac{e^{-3.5041+0.4049\times 0}}{1+e^{-3.5041+0.4049\times 0}} = 0.0292$$

Multiple Logistic Regression

Suppose there is more than one regressor. Analogously to the extension from simple to multiple linear regression, we have:

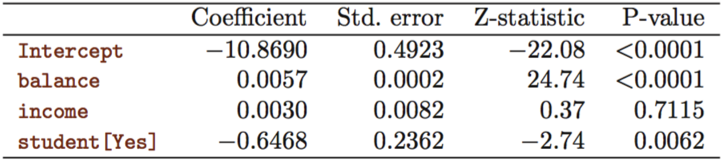
$$\log\left(\frac{p(X)}{(1-p(X))}\right) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$$

then,

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\ldots+\beta_p X_p}}{1 + e^{\beta_0+\beta_1 X_1+\ldots+\beta_p X_p}}$$

Multiple Logistic Regression example

Consider the `Credit` data, we want to predict `default` based on 3 predictors: `balance`, `income` and `student`.



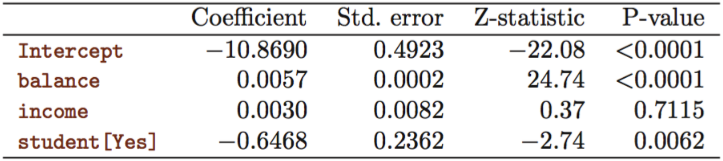
	Coefficient	Std. error	Z-statistic	P-value
`Intercept`	-10.8690	0.4923	-22.08	<0.0001
`balance`	0.0057	0.0002	24.74	<0.0001
`income`	0.0030	0.0082	0.37	0.7115
`student[Yes]`	-0.6468	0.2362	-2.74	0.0062

Then, we can make predictions:

For example, a student with a credit card balance of \$1, 500 and an income of 40 K\$ has an estimated probability of default of:

Multiple Logistic Regression example

Consider the `Credit` data, we want to predict `default` based on 3 predictors: `balance`, `income` and `student`.



	Coefficient	Std. error	Z-statistic	P-value
`Intercept`	−10.8690	0.4923	−22.08	<0.0001
`balance`	0.0057	0.0002	24.74	<0.0001
`income`	0.0030	0.0082	0.37	0.7115
`student[Yes]`	−0.6468	0.2362	−2.74	0.0062

Then, we can make predictions:

For example, a student with a credit card balance of \$1, 500 and an income of 40 K\$ has an estimated probability of default of:

$$\hat{p}(X) = \frac{e^{-10.869+0.0057\times1{,}500+0.003\times40-0.6468\times1}}{1+e^{-10.869+0.0057\times1{,}500+0.003\times40-0.6468\times1}} = 0.058$$

Multiple Logistic Regression example

Consider the `Credit` data, we want to predict `default` based on 3 predictors: `balance`, `income` and `student`.

	Coefficient	Std. error	Z-statistic	P-value
`Intercept`	−10.8690	0.4923	−22.08	<0.0001
`balance`	0.0057	0.0002	24.74	<0.0001
`income`	0.0030	0.0082	0.37	0.7115
`student[Yes]`	−0.6468	0.2362	−2.74	0.0062

Then, we can make predictions:
For example, a student with a credit card balance of \$1, 500 and an income of 40 K\$ has an estimated probability of default of:

$$\hat{p}(X) = \frac{e^{-10.869+0.0057\times 1{,}500+0.003\times 40-0.6468\times 1}}{1+e^{-10.869+0.0057\times 1{,}500+0.003\times 40-0.6468\times 1}} = 0.058$$

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

$$\hat{p}(X) = \frac{e^{-10.869+0.00574\times 1{,}500+0.003\times 40-0.6468\times 0}}{1+e^{-10.869+0.00574\times 1{,}500+0.003\times 40-0.6468\times 0}} = 0.105$$

Logistic regression with more than two classes

What happens if the target variable has more than 2 categories?

We can generalize logistic regression to a κ-level output variable as follows:

$$P(Y = k|X) = \frac{e^{\beta_{0k}+\beta_{1k}X_1+\ldots+\beta_{pk}X_p}}{\sum_{\ell=1}^{\kappa} e^{\beta_{0\ell}+\beta_{1\ell}X_1+\ldots+\beta_{p\ell}X_p}} \quad k \in 1, \ldots, \kappa$$

This is also called **softmax function**.

In this case there is a linear function for **each** class, except the last one since all the probabilities sum up to 1.
So, only $\kappa - 1$ linear functions are fitted.

Logistic regression with more than two classes is also referred to as **multinomial logistic regression**.

Outline

1 Introduction

2 K-Nearest Neighbors

3 Logistic Regression

4 Linear Discriminant Analysis

5 Other forms of Discriminant Analysis

6 Evaluating the quality of the predictions

7 A Comparison of Classification Methods

Introduction to Discriminant Analysis

In Discriminant Analysis:

- We treat the predictors X as random continuous variables and model the distribution of X in each of the classes separately.

Introduction to Discriminant Analysis

In Discriminant Analysis:

- We treat the predictors X as random continuous variables and model the distribution of X in each of the classes separately.
- Next, use **Bayes theorem** to obtain $P(Y = k|X = x)$.

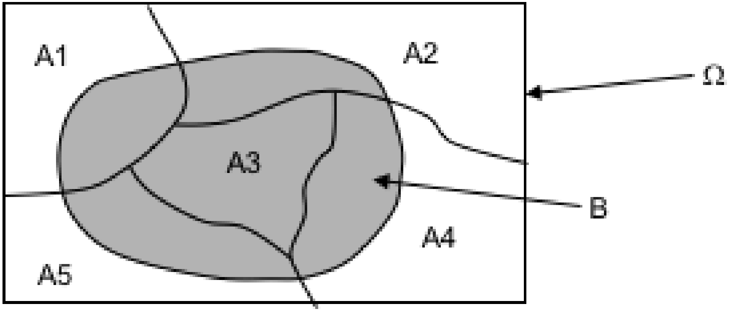
Bayes' theorem

Let $A_1, A_2, \dots, A_\kappa$ be a collection of κ mutually exclusive and exhaustive events with *prior* probabilities $P(A_k)\,\forall k \in \{1, \dots, \kappa\}$. Then, given an event B for which $P(B) > 0$, the *posterior* probability of A_k given that B has occurred is:

$$P(A_k|B) = \frac{P(A_k \cap B)}{P(B)} = \frac{P(B|A_k)P(A_k)}{\sum_{\ell=1}^{\kappa} P(B|A_\ell)P(A_\ell)}$$

Bayes' theorem scheme

$$P(A_k|B) = \frac{P(A_k \cap B)}{P(B)} = \frac{P(B|A_k)P(A_k)}{\sum_{\ell=1}^{5} P(B|A_\ell)P(A_\ell)}$$



Bayes theorem for LDA (Linear Discriminant Analysis)

Using the Bayes' theorem for the classification problem, the probability of class k given an observation x is:

$$P(Y = k|X = x) = \frac{P(X = x|Y = k)P(Y = k)}{P(X = x)}$$

Bayes theorem for LDA (Linear Discriminant Analysis)

Using the Bayes' theorem for the classification problem, the probability of class k given an observation x is:

$$P(Y=k|X=x) = \frac{P(X=x|Y=k)P(Y=k)}{P(X=x)}$$

We will use the following notation:

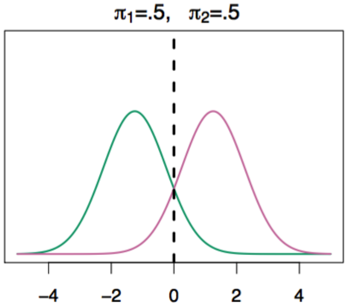
$$P(Y=k|X=x) = p_k(X=x) = \frac{\pi_k f_k(x)}{\sum_{\ell=1}^{K} \pi_\ell f_\ell(x)}$$

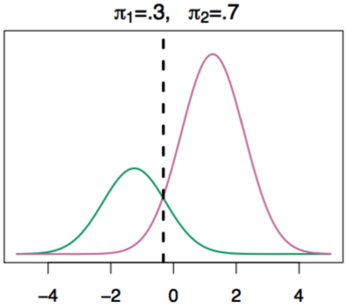
where:

- $\pi_k = P(Y=k)$ represent the overall or prior probability that a randomly chosen observation comes from the kth class;
- $f_k(x) = P(X=x|Y=k)$ is the density of X for an observation that belongs to class k.

Visual example: LDA with $\kappa = 2$ and $p = 1$

To simplify, we assume $f_k(X)$ is a normal distribution.
Example: In the case of 2 classes, we classify a new point according to which density is higher and one explanatory variable X.





On the left, $\pi_1 = \pi_2$, then compare $f_1(x)$ and $f_2(x)$.
On the right, $\pi_1 \neq \pi_2$, then compare $\pi_1 f_1(x)$ and $\pi_2 f_2(x)$.

Linear Discriminant Analysis when $p = 1$

We assume The density of X in class k follows a Gaussian density $\mathcal{N}(\mu_k, \sigma_k^2)$:

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

where:

- μ_k is the mean of X in class k and
- σ_k^2 is the variance of X in class k. For now, we assume $\sigma_1 = \ldots \sigma_K = \sigma$ are the same among all the classes.

Plugging this into Bayes formula, we get for $p_k(x) = P(Y = k|X = x)$

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

Discriminant functions

The Bayes classifier involves assigning an observation $X = x$ to the class for which $p_k(x)$ is **largest**. Taking logs, and discarding terms that do not depend on k, this is equivalent to assigning x to the class with the largest **discriminant score**:

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

Remark that δ_k is a **linear** function of x. That is where the name *Linear Discriminant Analysis (LDA)* comes from.

Discriminant functions, example (1/2)

If there are $\kappa = 2$ classes and $\pi_1 = \pi_2 = 0.5$, then the decision boundary is at

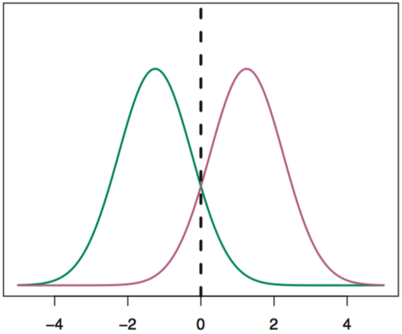
$$x = \frac{\mu_1 + \mu_2}{2}$$

That is $\delta_1(x) = \delta_2(x) \Leftrightarrow x = \frac{\mu_1^2 - \mu_2^2}{2(\mu_1 - \mu_2)} = \frac{\mu_1 + \mu_2}{2}$ Then, a test observation

x_0 will be classified in class 1 if $x_0 > \frac{\mu_1 + \mu_2}{2}$ (here we suppose $\mu_1 > \mu_2$) and to class 2 otherwise.

Discriminant functions, example (2/2)

In this example an observation is equally likely to come from either class, that is, $\pi_1 = \pi_2 = 0.5$.



The mean and variance parameters for the two density functions are $\mu_1 = -1.25$, $\mu_2 = 1.25$, and $\sigma_1^2 = \sigma_2^2 = 1$

The Bayes classifier assigns the observation to class 1 if $x < 0$ and class 2 otherwise.

Estimation of the parameters

In practice, even if we know X is drawn from a Gaussian distribution, the parameters are unknown, therefore we have to estimate them:

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

Estimation of the parameters

In practice, even if we know X is drawn from a Gaussian distribution, the parameters are unknown, therefore we have to estimate them:

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

Estimation of the parameters

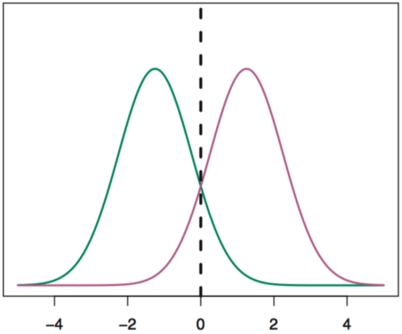
In practice, even if we know X is drawn from a Gaussian distribution, the parameters are unknown, therefore we have to estimate them:

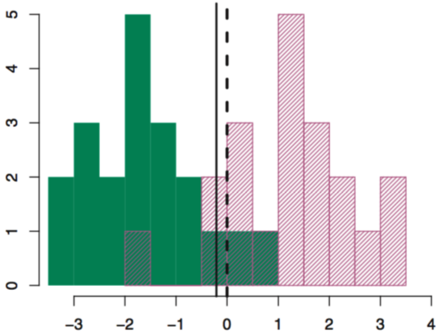
- $\hat{\pi}_k = \dfrac{n_k}{n}$
- $\hat{\mu}_k = \dfrac{1}{n_k} \displaystyle\sum_{i:y_i=k} x_i$
- $\hat{\sigma}^2 = \dfrac{1}{n-\kappa} \displaystyle\sum_{k=1}^{\kappa} \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2 = \sum_{k=1}^{\kappa} \dfrac{n_k - 1}{n-\kappa} \hat{\sigma}_k^2$
  where: $\hat{\sigma}_k^2 = \frac{1}{n_k-1} \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2$ is the usual formula for the estimated variance within the kth class.

where n is the total number of training observations, n_k is the number of training observations in the kth class. $\hat{\sigma}^2$ can be seen as a weighted average of the sample variances for each of the κ classes.

Discriminant functions, example with estimated parameters

Simulated data and the corresponding histogram for 20 observations from each class.



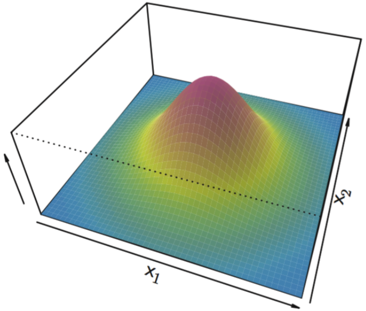


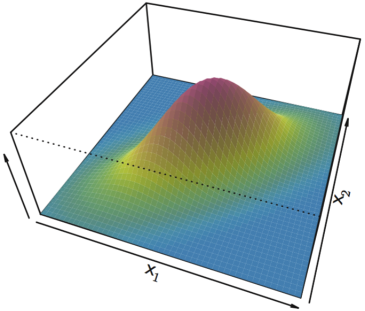
On the left the theoretical Bayes boundary (dashed line), on the right the decision boundary was calculated with the estimates (black solid line)

Since $\hat{\pi}_1 = \hat{\pi}_2$, the decision boundary corresponds to the midpoint between the sample means for the two classes, $(\mu_1 + \mu_2)/2$.

Linear Discriminant Analysis for $p > 1$

We assume that $X = (X_1, X2, ..., Xp)$ is drawn from a **multivariate Gaussian** or **multinormal** distribution





Left: Equal variance and zero correlation, Right: different variances and existing correlation.

where the density can be written: $f(x) = \dfrac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$

Where $\mu \in \mathbb{R}^p$ is the mean vector and Σ is the covariance matrix.

LDA with $p > 1$ predictors

The LDA classifier assumes that the observations in the kth class are drawn from a multivariate Gaussian distribution $\mathcal{N}(\mu_k, \Sigma)$, where:

- μ_k is mean vector of X specific to class k, and
- Σ is a covariance matrix that is supposed common to all κ classes.

LDA with $p > 1$ predictors

The LDA classifier assumes that the observations in the kth class are drawn from a multivariate Gaussian distribution $\mathcal{N}(\mu_k, \Sigma)$, where:

- μ_k is mean vector of X specific to class k, and
- Σ is a covariance matrix that is supposed common to all κ classes.

Plugging the density function for the kth class, $f_k(X = x)$, into the Bayes formula and a little of algebra reveals that the Bayes classifier assigns an observation $X = x$ to the class for which:

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

is largest.

LDA with $p > 1$ predictors

The LDA classifier assumes that the observations in the kth class are drawn from a multivariate Gaussian distribution $\mathcal{N}(\mu_k, \Sigma)$, where:

- μ_k is mean vector of X specific to class k, and
- Σ is a covariance matrix that is supposed common to all κ classes.

Plugging the density function for the kth class, $f_k(X = x)$, into the Bayes formula and a little of algebra reveals that the Bayes classifier assigns an observation $X = x$ to the class for which:

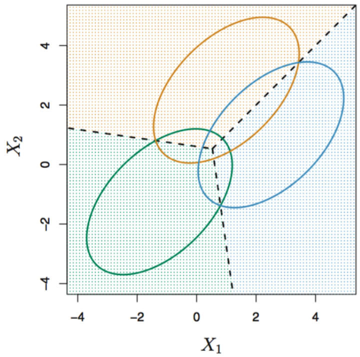
$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log(\pi_k)$$

is largest.

Notice that $\delta_k(x) = c_{k_0} + c_{k_1} x_1 + c_{k_2} x_2 + \ldots + c_{k_p} x_p$ is a linear function. That is the reason of the name *LDA (Linear Discriminant Analysis)*.

Example for $p = 2$ and $\kappa = 3$

Three equally-sized Gaussian classes are shown with class-specific mean vectors and a common covariance matrix.

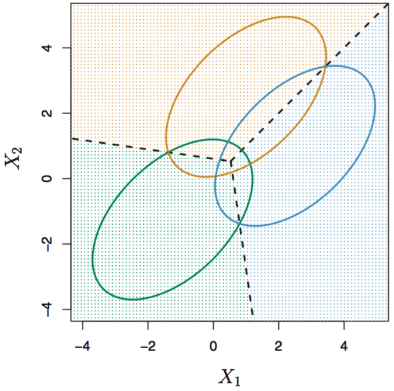


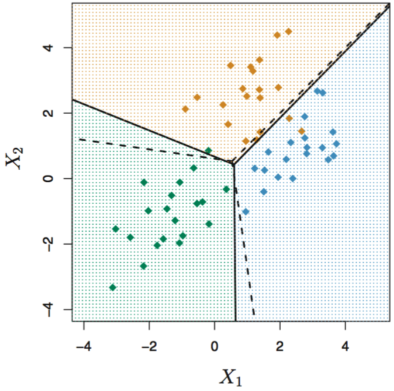
- The dashed lines represent the theoretical **Bayes decision boundaries**. So, they represent the set of values x for which $\delta_k(x) = \delta_\ell(x)$ for $k \neq \ell$. There is one line per each $\underline{\text{pair}}$ of classes.

These three Bayes decision boundaries divide the predictor space into three regions. The Bayes classifier will classify an observation according to the region in which it is located.

Example of estimation for $p = 2$ and $\kappa = 3$

Once more we estimate the unknown parameters $\mu_1, \ldots, \mu_\kappa$, $\pi_1, \ldots, \pi_\kappa$, and Σ. Given a new observation $X = x$, LDA calculates $\hat{\delta}(x)$ and classifies to the class for which it is largest.





On the right, the estimated LDA decision boundaries are shown as solid black lines.

Here, $n = 60$ observations, 20 per class.

From $\delta_k(x)$ to probabilities

Once we have the estimates $\hat{\delta}_k(x)$, we can turn these into estimates for class probabilities:

$$\hat{p}(Y = k|X = x) = \frac{\hat{\pi}_k e^{\hat{\delta}_k(x)}}{\sum_{\ell=1}^{\kappa} \hat{\pi}_\ell e^{\hat{\delta}_\ell(x)}}.$$

So classifying to the largest $\hat{\delta}_k(x)$ amounts to classifying to the class for which $P(Y = k|X = x)$ is the largest.

From $\delta_k(x)$ to probabilities

Once we have the estimates $\hat{\delta}_k(x)$, we can turn these into estimates for class probabilities:

$$\hat{p}(Y = k|X = x) = \frac{\hat{\pi}_k e^{\hat{\delta}_k(x)}}{\sum_{\ell=1}^{\kappa} \hat{\pi}_\ell e^{\hat{\delta}_\ell(x)}}.$$

So classifying to the largest $\hat{\delta}_k(x)$ amounts to classifying to the class for which $P(Y = k|X = x)$ is the <u>largest</u>.

When $\kappa = 2$, classify to class 2 if $P(Y = 2|X = x) > 0.5$, else to class 1.

Outline

1. Introduction
2. K-Nearest Neighbors
3. Logistic Regression
4. Linear Discriminant Analysis
5. Other forms of Discriminant Analysis
6. Evaluating the quality of the predictions
7. A Comparison of Classification Methods

Quadratic Discriminant Analysis (QDA)

QDA, like *LDA* assumes the X are drawn from a multivariate Gaussian distribution. However, unlike LDA, QDA assumes that
<u>each class has its own covariance matrix</u>.

If X comes from the kth class, then $X \sim \mathcal{N}(\mu_k, \Sigma_k)$

According to bayes classifier, an observation x will be assigned to class k if:

$$\begin{aligned} \delta(x) &= -\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k) + \log \pi_k - \frac{1}{2}\log|\sigma_k| \qquad (3) \\ &= -\frac{1}{2}x^T \Sigma_k^{-1} x + x^T \Sigma_k^{-1}\mu_k - \frac{1}{2}\mu_k^T \Sigma_k^{-1}\mu_k + \log \pi_k - \frac{1}{2}\log|\sigma_k| \end{aligned}$$

is largest.

Notice that this is a **quadratic** function of x. That is where the name QDA comes from!

Why to use QDA instead of LDA?

The answer lies in the **bias-variance trade-off**:

- Estimating a covariance matrix implies estimating $p(p+1)/2$ parameters for each class. Whereas *LDA* implies estimating only one covariance matrix.

Why to use QDA instead of LDA?

The answer lies in the **bias-variance trade-off**:

- Estimating a covariance matrix implies estimating $p(p+1)/2$ parameters for each class. Whereas *LDA* implies estimating only one covariance matrix.
- *LDA* is a much less flexible classifier than *QDA*, and so has substantially lower variance.

Why to use QDA instead of LDA?

The answer lies in the **bias-variance trade-off**:

- Estimating a covariance matrix implies estimating $p(p+1)/2$ parameters for each class. Whereas *LDA* implies estimating only one covariance matrix.
- *LDA* is a much less flexible classifier than *QDA*, and so has substantially lower variance.
- However, if *LDA*'s assumption that the κ classes share a common covariance matrix is wrong, then *LDA* can suffer from high bias. So, *QDA* would be a better choice.

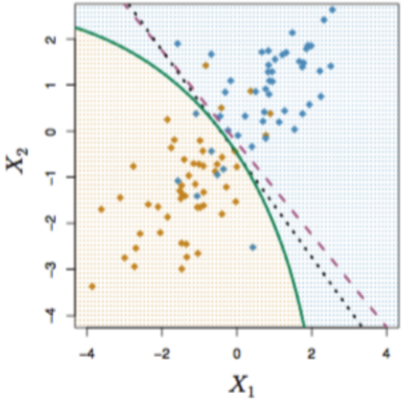
Why to use QDA instead of LDA?

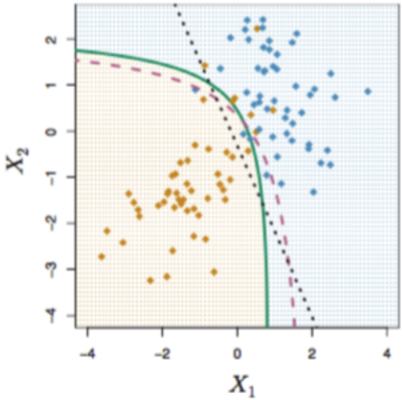
The answer lies in the **bias-variance trade-off**:

- Estimating a covariance matrix implies estimating $p(p+1)/2$ parameters for each class. Whereas *LDA* implies estimating only one covariance matrix.
- *LDA* is a much less flexible classifier than *QDA*, and so has substantially lower variance.
- However, if *LDA*'s assumption that the κ classes share a common covariance matrix is wrong, then *LDA* can suffer from high bias. So, *QDA* would be a better choice.
- If n is small and so reducing variance is crucial *LDA* tends to be better than *QDA*. Incontrast, *QDA* is recommended if the training set is very large, so the variance of the classifier is not a major concern.

QDA, example

Observations drawn from two Gaussian variables.





Bayes' classifier (purple dashed), *LDA* (black dotted), and *QDA* (green solid).
Left: common correlation of 0.7 among the two classes.
Right: Orange class has 0.7 correlation, whereas blue class has -0.7 correlation.

Naive Bayes

The Bayes theorem implies:

$$P(Y = k|X = x) = p_k(X = x) = \frac{\pi_k f_k(x)}{\sum_{\ell=1}^{\kappa} \pi_\ell f_\ell(x)}$$

Naive Bayes classifier assumes conditional independence between the feature variables, $f_k(x) = \prod_{j=1}^{p} f_{jk}(x_j)$. For a Gaussian distribution, this means that Σ_k are diagonal.

$$\delta_k(x) \propto \log\left(\pi_k \prod_{j=1}^{p} f_{kj}(x_j)\right) = -\frac{1}{2}\sum_{j=1}^{p} \frac{(x_j - \mu_{kj})^2}{\sigma_{kj}^2} + \log \pi_k$$

It is useful when p is large. Despite strong assumptions, naive Bayes often produces good classification results.

Outline

1 Introduction

2 K-Nearest Neighbors

3 Logistic Regression

4 Linear Discriminant Analysis

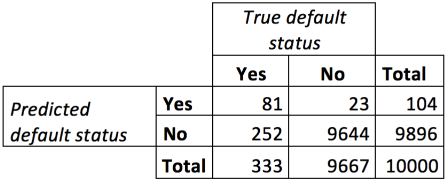
5 Other forms of Discriminant Analysis

6 Evaluating the quality of the predictions

7 A Comparison of Classification Methods

LDA on the `Credit` data, the confusion matrix (1/2)

We want to predict whether or not an individual will default on the basis of credit card `balance` and `income`. The **confusion matrix**

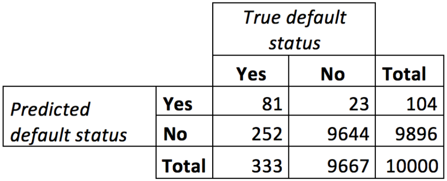


		True default status		
		Yes	**No**	**Total**
Predicted default status	**Yes**	81	23	104
	No	252	9644	9896
	Total	333	9667	10000

$(23 + 252)/10000$ errors, so a 2.75% training error rate. In contrast the quantity $(81 + 9644)/10000 = 97.25\%$ is called **accuracy**!

LDA on the `Credit` data, the confusion matrix (1/2)

We want to predict whether or not an individual will default on the basis of credit card `balance` and `income`. The **confusion matrix**

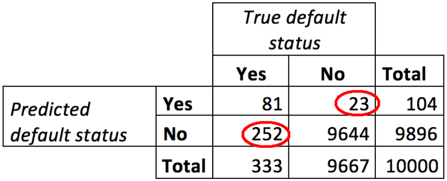


		True default status		
		Yes	**No**	**Total**
Predicted default status	**Yes**	81	23	104
	No	252	9644	9896
	Total	333	9667	10000

$(23 + 252)/10000$ errors, so a 2.75% training error rate. In contrast the quantity $(81 + 9644)/10000 = 97.25\%$ is called **accuracy**!
However:

- Only 3.33% of the individuals defaulted. So, a trivial classifier that always predicts `not default`, will result in an error rate of $\underline{3.33\%}$.

LDA on the `Credit` data, the confusion matrix (2/2)



		True default status		
		Yes	**No**	**Total**
Predicted default status	**Yes**	81	23	104
	No	252	9644	9896
	Total	333	9667	10000

- Of the true `No`'s, we make $23/9667 = 0.2\%$ errors; of the true `Yes`'s, we make $252/333 =$ 75.7% errors!

Types of errors

A binary classifier can make two types of errors:

- **False positive rate**: The fraction of negative examples that are classified as positive, 0.2% in example.
- **False negative rate**: The fraction of positive examples that are classified as negative, 75.7% in example.

Types of errors

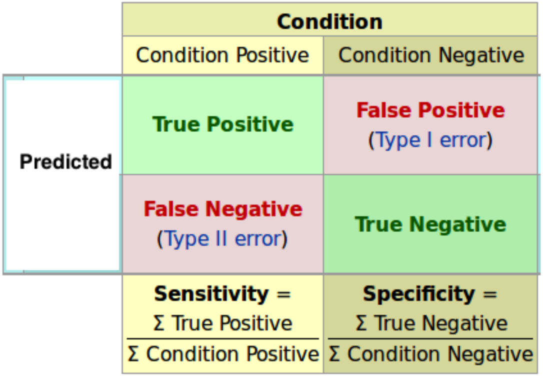
A binary classifier can make two types of errors:

- **False positive rate**: The fraction of negative examples that are classified as positive, 0.2% in example.
- **False negative rate**: The fraction of positive examples that are classified as negative, 75.7% in example.

It is often of interest to evaluate class-specific performance.

The bank can be more interested in detecting people who default than people that do not default.

Confusion matrix



	Condition	
	Condition Positive	Condition Negative
Predicted	**True Positive**	**False Positive** (Type I error)
	False Negative (Type II error)	**True Negative**
	Sensitivity = $\frac{\Sigma \text{ True Positive}}{\Sigma \text{ Condition Positive}}$	**Specificity** = $\frac{\Sigma \text{ True Negative}}{\Sigma \text{ Condition Negative}}$

- **sensitivity** or **recall** (True positive rate): percentage of true defaulters that are identified, 24.3% in the example.
- **specificity** (True negative rate): percentage of non-defaulters that are correctly identified, this case, 99.8% in the example.

Changing the *threshold*

The LDA produces a low *sensitivity* because it tries to approximate the Bayes classifier. The Bayes classifier will yield the smallest possible total number of misclassified observations, irrespective of which class the errors come from.
LDA assigns an observation to class `Yes` if:

$$\hat{p}(Y = \texttt{Yes} | \text{Balance,Income}) \geq 0.5$$

Changing the *threshold*

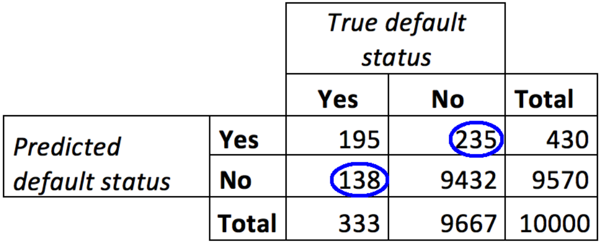
The LDA produces a low *sensitivity* because it tries to approximate the Bayes classifier. The Bayes classifier will yield the smallest possible total number of misclassified observations, irrespective of which class the errors come from.
LDA assigns an observation to class `Yes` if:

$$\hat{p}(Y = \texttt{Yes}|\text{Balance,Income}) \geq 0.5$$

In contrast, the bank might particularly wish to avoid incorrectly classifying an individual who will default. Why not to change this *threshold* and classify any customer with a posterior probability of default above 20% to the default class?

$$\hat{p}(Y = \texttt{Yes}|\text{Balance,Income}) \geq 0.2$$

LDA for `Credit` data with threshold 0.2



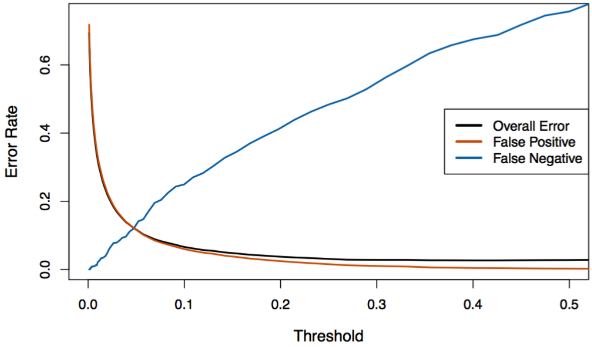
		True default status		
		Yes	**No**	**Total**
Predicted default status	**Yes**	195	235	430
	No	138	9432	9570
	Total	333	9667	10000

- Now the false negative rate decreased to 41.4%.
- However, the false positive rate has increased. As a result the overall error rate has increased slightly to 3.73%.

We can try different values of *threshold*.

Varying the *threshold*

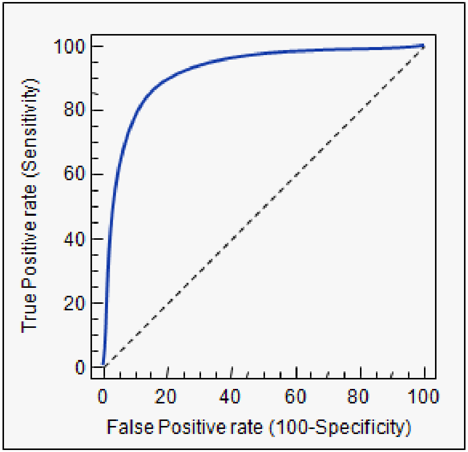
The trade-off that results from modifying the threshold value for the posterior probability of `default`.



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

ROC (Receiver Operating Characteristics) curve

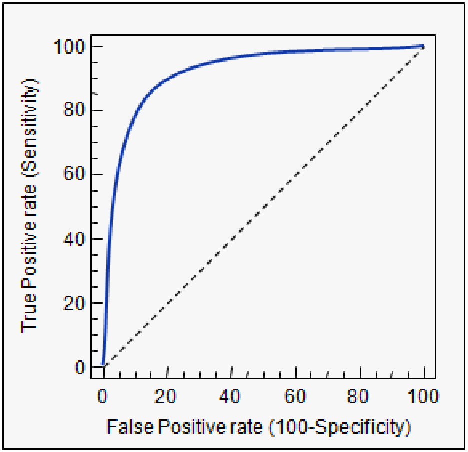
The overall performance of a classifier, summarized over all possible thresholds, is given by the area under the (ROC) curve (AUC).
The *ROC* curve plots the *sensitivity* versus (1-*specificity*)



- Ideally $AUC = 1$.
- A classifier that performs no better than chance has an $AUC = 0.5$.

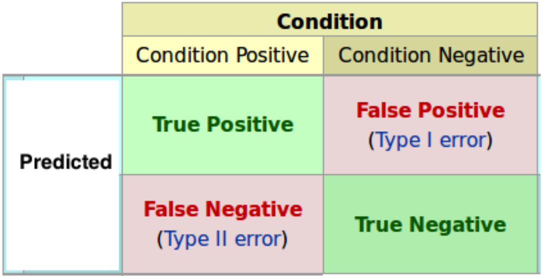
ROC (Receiver Operating Characteristics) curve

The overall performance of a classifier, summarized over all possible thresholds, is given by the area under the (ROC) curve (AUC).
The *ROC* curve plots the *sensitivity* versus (1-*specificity*)



- Ideally $AUC = 1$.
- A classifier that performs no better than chance has an $AUC = 0.5$.
- ROC curves are useful for comparing different classifiers, since they take into account all possible thresholds.

Confusion matrix: F_1 score



	Condition	
	Condition Positive	Condition Negative
Predicted	**True Positive**	**False Positive** (Type I error)
	False Negative (Type II error)	**True Negative**

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

$$\text{Sensitivity/ recall} = \frac{\Sigma\ \text{True Positive}}{\Sigma\ \text{Condition Positive}}$$

$$F_1\ \text{score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Outline

1 Introduction

2 K-Nearest Neighbors

3 Logistic Regression

4 Linear Discriminant Analysis

5 Other forms of Discriminant Analysis

6 Evaluating the quality of the predictions

7 A Comparison of Classification Methods

Logistic Regression versus LDA

For a two-class problem, one can show that for LDA:

$$\log\left(\frac{p_1(x)}{1-p_1(x)}\right) = \log\left(\frac{p_1(x)}{p_2(x)}\right) = c_0 + c_1x_1 + \ldots + c_px_p$$

Hence, both, *LDA* and logistic regression have a linear boundary.

- The difference is in how the parameters are estimated.
- *LDA* assumes that the observations are drawn from a Gaussian distribution with a common covariance matrix in each class, it is preferable over logistic regression when this assumption approximately holds. Conversely, logistic regression can outperform *LDA* if these Gaussian assumptions are not met.

Logistic Regression versus LDA

For a two-class problem, one can show that for LDA:

$$\log\left(\frac{p_1(x)}{1-p_1(x)}\right) = \log\left(\frac{p_1(x)}{p_2(x)}\right) = c_0 + c_1 x_1 + \ldots + c_p x_p$$

Hence, both, *LDA* and logistic regression have a linear boundary.

- The difference is in how the parameters are estimated.
- *LDA* assumes that the observations are drawn from a Gaussian distribution with a common covariance matrix in each class, it is preferable over logistic regression when this assumption approximately holds. Conversely, logistic regression can outperform *LDA* if these Gaussian assumptions are not met.
- logistic regression can also fit quadratic boundaries like QDA, by explicitly including quadratic terms in the model.

Summary

- When the true decision boundaries are linear, then the *LDA* and logistic regression approaches will tend to perform well.

Summary

- When the true decision boundaries are linear, then the *LDA* and logistic regression approaches will tend to perform well.
- When the boundaries are moderately non-linear, *QDA* may give better results.

Summary

- When the true decision boundaries are linear, then the *LDA* and logistic regression approaches will tend to perform well.
- When the boundaries are moderately non-linear, *QDA* may give better results.
- For much more complicated decision boundaries, a non-parametric approach such as *KNN* can be superior. But the level of smoothness for a non-parametric approach must be chosen carefully.

Summary

- When the true decision boundaries are linear, then the *LDA* and logistic regression approaches will tend to perform well.
- When the boundaries are moderately non-linear, *QDA* may give better results.
- For much more complicated decision boundaries, a non-parametric approach such as *KNN* can be superior. But the level of smoothness for a non-parametric approach must be chosen carefully.
- *LDA* is useful when n is small, or the classes are well separated, and Gaussian assumptions are reasonable. Also when $\kappa > 2$.

Summary

- When the true decision boundaries are linear, then the *LDA* and logistic regression approaches will tend to perform well.
- When the boundaries are moderately non-linear, *QDA* may give better results.
- For much more complicated decision boundaries, a non-parametric approach such as *KNN* can be superior. But the level of smoothness for a non-parametric approach must be chosen carefully.
- *LDA* is useful when n is small, or the classes are well separated, and Gaussian assumptions are reasonable. Also when $\kappa > 2$.
- Naive Bayes is useful when p is very large.

Outline

1 Introduction

2 K-Nearest Neighbors

3 Logistic Regression

4 Linear Discriminant Analysis

5 Other forms of Discriminant Analysis

6 Evaluating the quality of the predictions

7 A Comparison of Classification Methods

References

- James, Gareth; Witten, Daniela; Hastie, Trevor and Tibshirani, Robert. "An Introduction to Statistical Learning with Applications in R", 2nd edition, New York : "Springer texts in statistics", 2021. Site web: `https://hastie.su.domains/ISLR2/ISLRv2_website.pdf`
- Hastie, Trevor; Tibshirani, Robert and Friedman, Jerome (2009). "The Elements of Statistical Learning (Data Mining, Inference, and Prediction), 2nd edition". New York: "Springer texts in statistics". Site web : `http://statweb.stanford.edu/~tibs/ElemStatLearn/`