Modelos Estadísticos para la Regresión y la Clasificación
Clase 4: Aprendizaje Estadístico - Nociones básicas

Mathias Bourel

Instituto de Matemática y Estadística Prof. Rafael Laguardia (IMERL)
Facultad de Ingeniería, Universidad de la República, Uruguay

30 de agosto de 2024

Plan

When you are working with data...

and you are thinking about a model to use, it is useful to remember that:

- Data Modeling Culture - Algorithm Modeling Culture
- Supervised - Unsupervised
- Supervised: Clasification - Regression
- Types of variables, Missing Values
- Accuracy of the method is important but it is even more important over *test data*.
- Multiplicity of good models: aggregation methods
- Occam's razor dilemna: simpler is better? Simplicity vs Accuracy?
- Curse of Dimensionality? Handicap or blessing?
- *(The focus)..is on solving the problem instead of asking what data model (they can create). The best solution could be an algorithmic model, or may be a data model, or may be a combination. But the trick to being a scientist is to be open to using a wide variety of tools*, Breiman, The two cultures.
- *All models are wrong, but some are useful*, Georges Box (1919-2013)

Describe → Explain → Predict

$$B = B_0 M^{3/4}$$
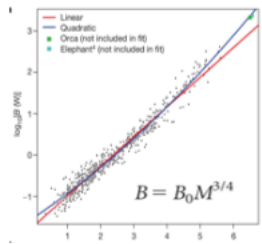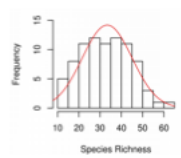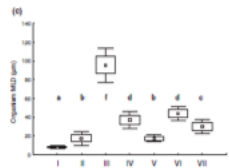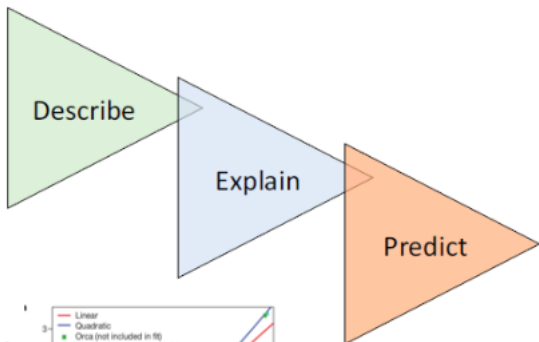
Kolokotrones et al 2010

- Predict whether an email is spam or not spam.
- Predict whether a patient is prone to heart disease.
- Estimate the ozone rate in a city taking into account climatic variables.
- Predict the absence or presence of a species in a given environment.
- Predicting customer leaks for a financial institution.
- Identify handwritten figures of postcards in envelopes.
- Split a population into several subgroups.
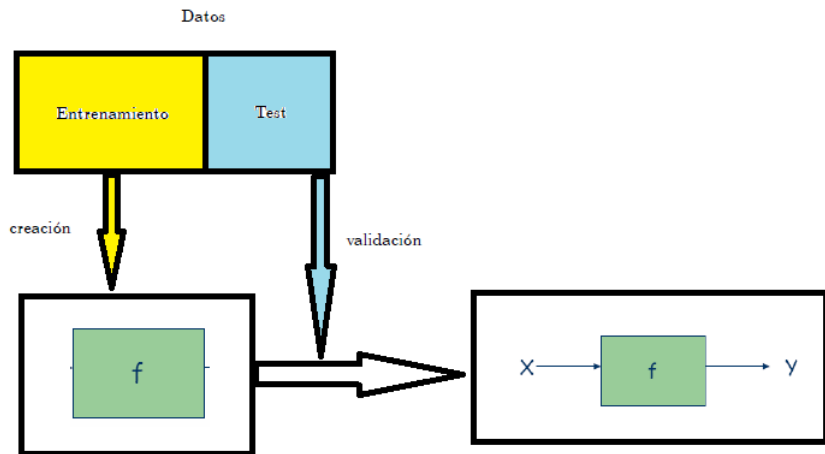
Example

Dataset Advertising:

```
> datos=read.csv("Advertising.csv",header=T,sep=",")
> datos[,-1]
        TV Radio Newspaper Sales
1    230.1  37.8      69.2  22.1
2     44.5  39.3      45.1  10.4
3     17.2  45.9      69.3   9.3
4    151.5  41.3      58.5  18.5
5    180.8  10.8      58.4  12.9
```

In this case, each row of the dataset is an independent realization of the random multivariate variable (X, Y) where:

- $X = (X_1, X_2, X_3)$ is the *input* vector:
    - X_1 budget allocated to advertising by television (TV)
    - X_2 budget allocated to advertising by radio (Radio)
    - X_3 budget allocated to advertising by newspaper (Newspaper)
- Y (Sales) is the amount of sales made and is the output variable (response), dependent variable.

In general we will want models of the general form:

$$Y = f(X_1, \ldots, X_p) + \epsilon$$

where $X_1, X_2, \ldots, X_p$ are predictor variables e Y is the response variable,
ϵ is the error term, independent of X and with mean 0.

General framework:
$\mathcal{L}$ a data basis.

Framework of Machine Learning

General framework:

$\mathcal{L}$ a data basis. We search about $f : \mathcal{X} \to \mathcal{Y}$ a good predictor or a good explainer.

- Supervised Learning: $\mathcal{L} = \{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathcal{X} \times \mathcal{Y} \subset \mathbb{R}^d \times \mathbb{R}$
  X: input variable, independent variable, explanatory (real o multidimensional), continuous, categorical, binary, ordinal.
  Y: output variable, dependent variable, real o categorical.
  - Classification: $y \in \{-1, 1\}$ (binary) or $y \in \{1, \ldots, K\}$ (multiclass).
  - Regression: $y \in \mathbb{R}$.

- Unsupervised Learning $\mathcal{L} = \{x_1, \ldots, x_n\} \subset \mathcal{X} \subset \mathbb{R}^d$
  - Clustering
  - Density estimation

  In all cases, the sample $\mathcal{L}$ is a collection of n independents realization of a multivariate random variable (X, Y) or X

A little formality

Let see how three famous problem in Statistics can be viewed as statistical learning problem.

A little formality

Let see how three famous problem in Statistics can be viewed as statistical learning problem.

Consider a *loss function* L, i.e $L(y, u)$ which measures the cost of deciding $u = f(x)$ for the input x knowing that y is the true output.

A little formality

Let see how three famous problem in Statistics can be viewed as statistical learning problem.

Consider a *loss function* L, i.e $L(y, u)$ which measures the cost of deciding $u = f(x)$ for the input x knowing that y is the true output.

Ejemplos:

1. $L(y, u) = 1_{\{y \neq u\}}$ (classification)

A little formality

Let see how three famous problem in Statistics can be viewed as statistical learning problem.

Consider a *loss function* L, i.e $L(y, u)$ which measures the cost of deciding $u = f(x)$ for the input x knowing that y is the true output.

Ejemplos:

1. $L(y, u) = 1_{\{y \neq u\}}$ (classification)
2. $L(y, u) = (y - u)^2$ (regression)

A little formality

Let see how three famous problem in Statistics can be viewed as statistical learning problem.

Consider a *loss function* L, i.e $L(y, u)$ which measures the cost of deciding $u = f(x)$ for the input x knowing that y is the true output.

Ejemplos:

1. $L(y, u) = 1_{\{y \neq u\}}$ (classification)
2. $L(y, u) = (y - u)^2$ (regression)
3. $L(u) = -log(u)$ (density estimation)

A little formality

Let see how three famous problem in Statistics can be viewed as statistical learning problem.

Consider a *loss function L*, i.e $L(y, u)$ which measures the cost of deciding $u = f(x)$ for the input x knowing that y is the true output.

Ejemplos:

1. $L(y, u) = 1_{\{y \neq u\}}$ (classification)
2. $L(y, u) = (y - u)^2$ (regression)
3. $L(u) = -log(u)$ (density estimation)

We look for a function f_C (the original), among all the functions of a certain class C, that minimizes the expected value of L (which we call *risk* or *Expected Predictive Error*), i.e:

$$f_C = \underset{f \in C}{\text{Argmin}} R_L(f) = \underset{f \in C}{\text{Argmin}} \mathbb{E}\big(L(Y, f(X))\big)$$

A little formality

Let see how three famous problem in Statistics can be viewed as statistical learning problem.

Consider a *loss function* L, i.e $L(y, u)$ which measures the cost of deciding $u = f(x)$ for the input x knowing that y is the true output.

Ejemplos:

1. $L(y, u) = 1_{\{y \neq u\}}$ (classification)
2. $L(y, u) = (y - u)^2$ (regression)
3. $L(u) = -log(u)$ (density estimation)

We look for a function f_C (the original), among all the functions of a certain class C, that minimizes the expected value of L (which we call *risk* or *Expected Predictive Error*), i.e:

$$f_C = \underset{f \in C}{\text{Argmin}} R_L(f) = \underset{f \in C}{\text{Argmin}} \mathbb{E}\big(L(Y, f(X))\big)$$

The choice of C depends on the nature of the phenomenon being modeled, the hypotheses and experience on the data available, the opinion of the experts, etc.

A little formality

In practice, this predictor is constructed from a data set $\mathcal{L} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ where $x_i \in \mathcal{X} \subset \mathbb{R}^d$ and $y_i \in \mathcal{Y} = \{1, \ldots, K\}$ or $y_i \in \mathcal{Y} \subset \mathbb{R}$ where it supposed that all the n labeled observations of $\mathcal{L}$ are independent realization of the variable (X, Y) with unknown distribution law.

As it is impossible to lead with the expected risk (as distribution of (X, Y) is unknown), the goal consists to minimize the empirical risk

$$R_{n,L}(f) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i))$$

That is to search a function $\widehat{f}_n \in \mathcal{C}$ such that:

$$\widehat{f}_n = \underset{f \in \mathcal{C}}{\text{Argmin}} R_{n,L}(f) = \underset{f \in \mathcal{C}}{\text{Argmin}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i))$$

For example, in a classification problem if $y \in \{1, \ldots, K\}$, we use as loss function $L(x, y, u) = \mathbb{1}_{\{u \neq y\}}$.

The classification problem

For example, in a classification problem if $y \in \{1, \ldots, K\}$, we use as loss function $L(x, y, u) = \mathbb{1}_{\{u \neq y\}}$.

The associated risk with L is:

$$R_L(f) = \mathbb{P}(Y \neq f(X))$$

and the empirical risk is

$$R_{L,n}(f) = \frac{1}{n} \#\{i : f(x_i) \neq y_i\}$$

The function that minimizes $R_L(f)$ is

$$f^*(x) = \underset{k \in \{1, \ldots, K\}}{\text{Argmax}} \ \mathbb{P}(Y = k | X = x)$$

and predicts the class k that maximizes the posterior probability of Y knowing X. This classifier is known as *Bayes classifier* and can be interpreted as follows:, the problem is reduced in looking for that function that minimizes the amount of errors committed on the sample.

The classification problem

Suppose our problem is binary and we want to classify the observations into two categories: 0 and 1. In this case the Bayes classifier is the function f^* that minimizes the probability of being wrong:

$$f^* = \underset{f:\mathcal{X}\rightarrow\{0,1\}}{\text{Argmin}} \ \mathbb{P}(f(X) \neq Y)$$

The classification problem

Suppose our problem is binary and we want to classify the observations into two categories: 0 and 1. In this case the Bayes classifier is the function f^* that minimizes the probability of being wrong:

$$f^* = \underset{f:\mathcal{X} \to \{0,1\}}{\operatorname{Argmin}} \ \mathbb{P}(f(X) \neq Y)$$

The classification problem

Suppose our problem is binary and we want to classify the observations into two categories: 0 and 1. In this case the Bayes classifier is the function f^* that minimizes the probability of being wrong:

$$f^* = \underset{f:\mathcal{X}\to\{0,1\}}{\text{Argmin}} \; \mathbb{P}(f(X) \neq Y)$$

We will prove that

$$g(x) = \underset{y\in\{0,1\}}{\text{Argmax}} \mathbb{P}(Y = y|X = x) = \left\{ \begin{array}{lll} 1 & \text{si} & \mathbb{P}(Y = 1|X = x) \geq \frac{1}{2} \\ 0 & \text{si} & \mathbb{P}(Y = 1|X = x) < \frac{1}{2} \end{array} \right. = f^*(x)$$

The classification problem

Suppose our problem is binary and we want to classify the observations into two categories: 0 and 1. In this case the Bayes classifier is the function f^* that minimizes the probability of being wrong:

$$f^* = \underset{f:\mathcal{X} \to \{0,1\}}{\text{Argmin}} \ \mathbb{P}(f(X) \neq Y)$$

We will prove that

$$g(x) = \underset{y \in \{0,1\}}{\text{Argmax}} \mathbb{P}(Y = y|X = x) = \left\{ \begin{array}{ll} 1 & \text{si} \quad \mathbb{P}(Y = 1|X = x) \geq \frac{1}{2} \\ 0 & \text{si} \quad \mathbb{P}(Y = 1|X = x) < \frac{1}{2} \end{array} \right. = f^*(x)$$

Demostración.

If $g(x) = \underset{y \in \{0,1\}}{\text{Argmax}} \ \mathbb{P}(Y = y|X = x)$ and let considerer $f : \mathcal{X} \to \{0, 1\}$ another classifier and $x \in \mathcal{X}$. We have that

$$\mathbb{P}(f(X) \neq Y|X = x) = 1 - \mathbb{P}(f(X) = Y|X = x).$$

The classification problem

Suppose our problem is binary and we want to classify the observations into two categories: 0 and 1. In this case the Bayes classifier is the function f^* that minimizes the probability of being wrong:

$$f^* = \operatorname*{Argmin}_{f:\mathcal{X}\to\{0,1\}} \mathbb{P}(f(X) \neq Y)$$

We will prove that

$$g(x) = \operatorname*{Argmax}_{y\in\{0,1\}} \mathbb{P}(Y = y|X = x) = \left\{ \begin{array}{lll} 1 & \text{si} & \mathbb{P}(Y = 1|X = x) \geq \frac{1}{2} \\ 0 & \text{si} & \mathbb{P}(Y = 1|X = x) < \frac{1}{2} \end{array} \right. = f^*(x)$$

Demostración.

If $g(x) = \operatorname*{Argmax}_{y\in\{0,1\}} \mathbb{P}(Y = y|X = x)$ and let considerer $f : \mathcal{X} \to \{0, 1\}$ another classifier and $x \in \mathcal{X}$. We have that

$$\mathbb{P}(f(X) \neq Y|X = x) = 1 - \mathbb{P}(f(X) = Y|X = x).$$

As $\mathbb{P}(Y = g(X)|X = x) = \text{máx}\{\mathbb{P}(Y = 0|X = x), \mathbb{P}(Y = 1|X = x)\}$ we have :

$$\mathbb{P}(f(X) \neq Y|X = x) - \mathbb{P}(g(X) \neq Y|X = x)$$

The classification problem

Suppose our problem is binary and we want to classify the observations into two categories: 0 and 1. In this case the Bayes classifier is the function f^* that minimizes the probability of being wrong:

$$f^* = \underset{f:\mathcal{X} \to \{0,1\}}{\text{Argmin}} \; \mathbb{P}(f(X) \neq Y)$$

We will prove that

$$g(x) = \underset{y \in \{0,1\}}{\text{Argmax}} \mathbb{P}(Y = y | X = x) = \left\{ \begin{array}{lll} 1 & \text{si} & \mathbb{P}(Y = 1 | X = x) \geq \frac{1}{2} \\ 0 & \text{si} & \mathbb{P}(Y = 1 | X = x) < \frac{1}{2} \end{array} \right. = f^*(x)$$

Demostración.

If $g(x) = \underset{y \in \{0,1\}}{\text{Argmax}} \; \mathbb{P}(Y = y | X = x)$ and let considerer $f : \mathcal{X} \to \{0, 1\}$ another classifier and $x \in \mathcal{X}$. We have that

$$\mathbb{P}(f(X) \neq Y | X = x) = 1 - \mathbb{P}(f(X) = Y | X = x).$$

As $\mathbb{P}(Y = g(X) | X = x) = \text{máx}\{\mathbb{P}(Y = 0 | X = x), \mathbb{P}(Y = 1 | X = x)\}$ we have :

$$\mathbb{P}(f(X) \neq Y | X = x) - \mathbb{P}(g(X) \neq Y | X = x) = \mathbb{P}(Y = g(X) | X = x) - \mathbb{P}(Y = f(X) | X = x) \geq 0$$

The classification problem

Suppose our problem is binary and we want to classify the observations into two categories: 0 and 1. In this case the Bayes classifier is the function f^* that minimizes the probability of being wrong:

$$f^* = \underset{f:\mathcal{X} \to \{0,1\}}{\text{Argmin}} \ \mathbb{P}(f(X) \neq Y)$$

We will prove that

$$g(x) = \underset{y \in \{0,1\}}{\text{Argmax}} \mathbb{P}(Y = y | X = x) = \left\{ \begin{array}{ll} 1 & \text{si} \quad \mathbb{P}(Y = 1 | X = x) \geq \frac{1}{2} \\ 0 & \text{si} \quad \mathbb{P}(Y = 1 | X = x) < \frac{1}{2} \end{array} \right. = f^*(x)$$

Demostración.

If $g(x) = \underset{y \in \{0,1\}}{\text{Argmax}} \ \mathbb{P}(Y = y | X = x)$ and let considerer $f : \mathcal{X} \to \{0, 1\}$ another classifier and $x \in \mathcal{X}$. We have that

$$\mathbb{P}(f(X) \neq Y | X = x) = 1 - \mathbb{P}(f(X) = Y | X = x).$$

As $\mathbb{P}(Y = g(X) | X = x) = \text{máx}\{\mathbb{P}(Y = 0 | X = x), \mathbb{P}(Y = 1 | X = x)\}$ we have :

$$\mathbb{P}(f(X) \neq Y | X = x) - \mathbb{P}(g(X) \neq Y | X = x) = \mathbb{P}(Y = g(X) | X = x) - \mathbb{P}(Y = f(X) | X = x) \geq 0$$

Then:

$$\forall x \in \mathcal{X} \quad \mathbb{P}(g(X) \neq Y | X = x) \leq \mathbb{P}(f(X) \neq Y | X = x)$$

The classification problem

Suppose our problem is binary and we want to classify the observations into two categories: 0 and 1. In this case the Bayes classifier is the function f^* that minimizes the probability of being wrong:

$$f^* = \operatorname*{Argmin}_{f:\mathcal{X} \to \{0,1\}} \mathbb{P}(f(X) \neq Y)$$

We will prove that

$$g(x) = \operatorname*{Argmax}_{y \in \{0,1\}} \mathbb{P}(Y = y | X = x) = \left\{ \begin{array}{lll} 1 & \text{si} & \mathbb{P}(Y = 1 | X = x) \geq \frac{1}{2} \\ 0 & \text{si} & \mathbb{P}(Y = 1 | X = x) < \frac{1}{2} \end{array} \right. = f^*(x)$$

Demostración.

If $g(x) = \operatorname*{Argmax}_{y \in \{0,1\}} \mathbb{P}(Y = y | X = x)$ and let considerer $f : \mathcal{X} \to \{0, 1\}$ another classifier and $x \in \mathcal{X}$. We have that

$$\mathbb{P}(f(X) \neq Y | X = x) = 1 - \mathbb{P}(f(X) = Y | X = x).$$

As $\mathbb{P}(Y = g(X) | X = x) = \text{máx}\{\mathbb{P}(Y = 0 | X = x), \mathbb{P}(Y = 1 | X = x)\}$ we have :

$$\mathbb{P}(f(X) \neq Y | X = x) - \mathbb{P}(g(X) \neq Y | X = x) = \mathbb{P}(Y = g(X) | X = x) - \mathbb{P}(Y = f(X) | X = x) \geq 0$$

Then:

$$\forall x \in \mathcal{X} \quad \mathbb{P}(g(X) \neq Y | X = x) \leq \mathbb{P}(f(X) \neq Y | X = x)$$

and therefore, if F is the distribution law of X, we get that:

$$\mathbb{P}(g(X) \neq Y) = \int \mathbb{P}(g(X) \neq Y | X = x) \, dF(x) \leq \int \mathbb{P}(f(X) \neq Y | X = x) \, dF(x) = \mathbb{P}(f(X) \neq Y) \quad \forall f$$

The classification problem

Suppose our problem is binary and we want to classify the observations into two categories: 0 and 1. In this case the Bayes classifier is the function f^* that minimizes the probability of being wrong:

$$f^* = \underset{f:\mathcal{X}\to\{0,1\}}{\text{Argmin}} \ \mathbb{P}(f(X) \neq Y)$$

We will prove that

$$g(x) = \underset{y\in\{0,1\}}{\text{Argmax}}\mathbb{P}(Y = y|X = x) = \left\{ \begin{array}{ll} 1 & \text{si} \quad \mathbb{P}(Y = 1|X = x) \geq \frac{1}{2} \\ 0 & \text{si} \quad \mathbb{P}(Y = 1|X = x) < \frac{1}{2} \end{array} \right. = f^*(x)$$

Demostración.

If $g(x) = \underset{y\in\{0,1\}}{\text{Argmax}} \ \mathbb{P}(Y = y|X = x)$ and let considerer $f : \mathcal{X} \to \{0, 1\}$ another classifier and $x \in \mathcal{X}$. We have that

$$\mathbb{P}(f(X) \neq Y|X = x) = 1 - \mathbb{P}(f(X) = Y|X = x).$$

As $\mathbb{P}(Y = g(X)|X = x) = \text{máx}\{\mathbb{P}(Y = 0|X = x), \mathbb{P}(Y = 1|X = x)\}$ we have :

$$\mathbb{P}(f(X) \neq Y|X = x) - \mathbb{P}(g(X) \neq Y|X = x) = \mathbb{P}(Y = g(X)|X = x) - \mathbb{P}(Y = f(X)|X = x) \geq 0$$

Then:

$$\forall x \in \mathcal{X} \quad \mathbb{P}(g(X) \neq Y|X = x) \leq \mathbb{P}(f(X) \neq Y|X = x)$$

and therefore, if F is the distribution law of X, we get that:

$$\mathbb{P}(g(X) \neq Y) = \int \mathbb{P}(g(X) \neq Y|X = x) \, dF(x) \leq \int \mathbb{P}(f(X) \neq Y|X = x) \, dF(x) = \mathbb{P}(f(X) \neq Y) \ \forall f \Rightarrow g = f^*$$

The classification problem

Another justification of Bayes Classifier; if $Y \in \{1, \ldots, K\}$ then:

$$R_L(f) = \underset{f \in \mathcal{C}}{\text{Argmin}} \; \mathbb{E}_{(X,Y)}\big(L(Y, f(X)) = \underset{f \in \mathcal{C}}{\text{Argmin}} \; \mathbb{E}_{\mathbf{X}}\left[\mathbb{E}_{\mathbf{Y}|\mathbf{X}=\mathbf{x}}(L(Y, f(X))|X = x)\right]$$

$$= \underset{f \in \mathcal{C}}{\text{Argmin}} \; \mathbb{E}_{\mathbf{X}}\left(\sum_{k'=1}^{K} L(k', f(x))\mathbb{P}(Y = k'|\mathbf{X} = \mathbf{x})\right)$$

We have to minimize pointwise this quantity:

$$f(x) = k^* = \underset{k \in \{1, \ldots, K\}}{\text{Argmin}} \sum_{k'=1}^{K} L(k', k)\mathbb{P}(Y = k'|\mathbf{X} = \mathbf{x})$$

Suppose we have two classes 1 and 2, then if $k = 1$:

$$L(1,1)\mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x}) + L(2,1)\mathbb{P}(Y = 2|\mathbf{X} = \mathbf{x}) = \mathbb{P}(Y = 2|\mathbf{X} = \mathbf{x}) = 1 - \mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x})$$

and if $k = 2$:

$$L(1,2)\mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x}) + L(2,2)\mathbb{P}(Y = 2|\mathbf{X} = \mathbf{x}) = \mathbb{P}(Y = 1|\mathbf{X} = \mathbf{x}) = 1 - \mathbb{P}(Y = 2|\mathbf{X} = \mathbf{x})$$

then

$$f(x) = \underset{k \in \{1,2\}}{\text{Argmin}} \left(1 - \mathbb{P}(Y = k|\mathbf{X} = \mathbf{x})\right) = \underset{k \in \{1,2\}}{\text{Argmax}} \mathbb{P}(Y = k|\mathbf{X} = \mathbf{x})$$

In the multiclass context, that is when there are more than two categories and the set of labels is $\{1, \ldots, K\}$, the Bayes' classifier is the one that assigns the label k to observation x that has the highest posterior probability, that is to say

$$f^*(x) = \underset{k \in \{1, \ldots, K\}}{\text{Argmax}} \ \mathbb{P}(Y = k | X = x)$$

In the multiclass context, that is when there are more than two categories and the set of labels is $\{1, \ldots, K\}$, the Bayes' classifier is the one that assigns the label k to observation x that has the highest posterior probability, that is to say

$$f^*(x) = \underset{k \in \{1, \ldots, K\}}{\text{Argmax}} \; \mathbb{P}(Y = k | X = x)$$

The Bayes classifier produces the smallest error that any classifier can make. The error if $X = x_0$ is

$$1 - \underset{k \in \{1, \ldots, K\}}{\max} \mathbb{P}(Y = k | X = x_0)$$

and therefore the error rate, covering all the possible values x_0 of X is

$$1 - \mathbb{E} \left(\underset{k \in \{1, \ldots, K\}}{\max} \mathbb{P}(Y = j | X) \right)$$

In the multiclass context, that is when there are more than two categories and the set of labels is $\{1, \ldots, K\}$, the Bayes' classifier is the one that assigns the label k to observation x that has the highest posterior probability, that is to say

$$f^*(x) = \operatorname*{Argmax}_{k \in \{1, \ldots, K\}} \mathbb{P}(Y = k | X = x)$$

The Bayes classifier produces the smallest error that any classifier can make. The error if $X = x_0$ is

$$1 - \max_{k \in \{1, \ldots, K\}} \mathbb{P}(Y = k | X = x_0)$$

and therefore the error rate, covering all the possible values x_0 of X is

$$1 - \mathbb{E}\left(\max_{k \in \{1, \ldots, K\}} \mathbb{P}(Y = j | X)\right)$$

This quantity is called *Bayes error rate*. However, in real problems, not knowing the distribution, it is impossible to calculate Bayes error rate.

Many algorithms try to estimate posterior probabilities and classify the observation in that class that maximize it (KNN, CART, Boosting, etc.)

The regression problem

In a regression problem we use as loss function $L(y, u) = (u - y)^2$ and look at a function $f : \mathbb{R}^d \to \mathbb{R}$ that minimizes risk $R_L(f) = \mathbb{E}_{(X,Y)}\left[(Y - f(X))^2\right]$ The function that minimizes $R_L(f)$ is the conditional expectation

$$f^*(x) = m(x) = \mathbb{E}(Y|X = x)$$

Instead of minimizing theoretical risk we minimize empirical risk $R_{L,n}(f) = \frac{1}{n} \sum\limits_{i=1}^{n} \left(y_i - f(x_i)\right)^2$ and then the solution is the function that minimizes the least squares method.

Demostración.

If $f : \mathbb{R}^d \to \mathbb{R}$ then: $\mathbb{E}\left[(f(X) - Y)^2\right] = \mathbb{E}\left[(f(X) - m(X) + m(X) - Y)^2\right] =$
$\mathbb{E}\left[(f(X) - m(X))^2\right] + \mathbb{E}\left[(m(X) - Y))^2\right] = \int \left(f(x) - m(x)\right)^2 dF(x) + \mathbb{E}\left[(m(X) - Y))^2\right]$ where F is the cumulative function of X and it is clear that expression above is minimum if $f = m$.
In this calculation we use that $\mathbb{E}_{(X,Y)}\left[(f(X) - m(X) + m(X) - Y))^2\right] =$
$\mathbb{E}_{(X,Y)}\left[(f(X) - m(X))^2\right] + \mathbb{E}_{(X,Y)}\left[(m(X) - Y)^2\right] + 2\mathbb{E}_{(X,Y)}((f(X) - m(X))(m(X) - Y))$ and the last term is null because

$$\mathbb{E}_{(X,Y)}((f(X) - m(X))(m(X) - Y)) = \mathbb{E}_X\left[\mathbb{E}_{Y|X}((f(X) - m(X))(m(X) - Y)|X)\right]$$

$$= \mathbb{E}_X\left[(f(X) - m(X))\mathbb{E}_{Y|X}(m(X) - Y|X)\right] = \mathbb{E}_X\left[(f(X) - m(X))(m(X) - \underbrace{\mathbb{E}_{Y|X}(Y|X)}_{m(X)})\right] = 0$$

We use as loss function $L(g(x)) = -\log(g(x))$. The associated risk is $R_L(g) = -\int \log(g(x))f(x)\,dx$ and the empirical risk is

$$R_{L,n}(g) = -\frac{1}{n}\sum_{i=1}^{n}\log\left(g(x_i)\right) = -\frac{1}{n}\log\left(\prod_{i=1}^{n}g(x_i)\right)$$
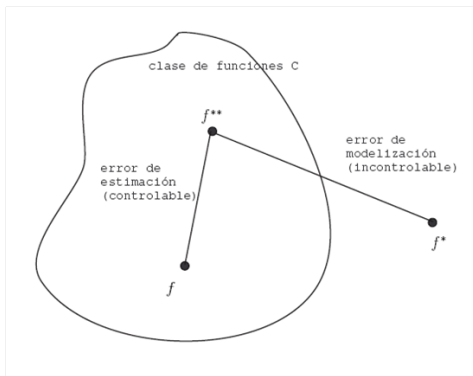
To find the function that minimize the empirical risk is equivalent to find the function that maximize the log-likelihood. Then It is straightforward to show that maximizing the log-likelihood is equivalent to minimize the Kullback-Leibler divergence

$$K(f,g) = \int \log\left(\frac{f(x)}{g(x)}\right)\,f(x)\,dx$$

For Jensen's inequality, it is easy to prove that $K(f,g) \geq 0$ but K does not satisfies symmetric condition and triangular inequality

Let summarize the different functions previously encountered:

- f is the theoretical predictor (we don't know it).
- f_C is the best among all possible predictors within a class of functions C (we don't know it).
- $\hat{f}_n$ is the predictor we use in practice, the function that minimizes empirical risk:

- Modelling error (associated with bias): $f - f_C$
  It depends on the choice of class C. Observe that if we consider as the family of all possible functions, we will have overfitting.
- Estimation error (associated with the variance): $\hat{f}_n - f_C$
  It is a statistical error, if the size of the sample is large, under certain hypotheses about the class C, it is true that $\hat{f}_n$ converge, when n tends to infinity to f_C. In fact it is a convergence of the risks (Vapnik's theorem)

Approach errors

- Modelling error (associated with bias): $f - f_C$
  It depends on the choice of class C. Observe that if we consider as the family of all possible functions, we will have overfitting.
- Estimation error (associated with the variance): $\hat{f}_n - f_C$
  It is a statistical error, if the size of the sample is large, under certain hypotheses about the class C, it is true that $\hat{f}_n$ converge, when n tends to infinity to f_C. In fact it is a convergence of the risks (Vapnik's theorem)

Theorem 1

The Fundamental Theorem of Learning (Vapnik, 1997) states that, under certain conditions on the class of functions C, $\hat{f}_n$ "converges" to f_C (risks through) . These conditions are related to the dimension of Vapnik-Chervonenkis (VC dimension) of the function class C. The VC dimension measures "how big" is an infinite class of functions, so if C is not too large, that is, the VC dimension is finite, is in the hypothesis of the Fundamental Theorem of Learning

- Modelling error (associated with bias): $f - f_C$
  It depends on the choice of class C. Observe that if we consider as the family of all possible functions, we will have overfitting.
- Estimation error (associated with the variance): $\hat{f}_n - f_C$
  It is a statistical error, if the size of the sample is large, under certain hypotheses about the class C, it is true that $\hat{f}_n$ converge, when n tends to infinity to f_C. In fact it is a convergence of the risks (Vapnik's theorem)

Theorem 1

The Fundamental Theorem of Learning (Vapnik, 1997) states that, under certain conditions on the class of functions C, $\hat{f}_n$ "converges" to f_C (risks through) . These conditions are related to the dimension of Vapnik-Chervonenkis (VC dimension) of the function class C. The VC dimension measures "how big" is an infinite class of functions, so if C is not too large, that is, the VC dimension is finite, is in the hypothesis of the Fundamental Theorem of Learning

Plan

1. General Framework and Introduction to Statistical Learning
   - Generalities
   - A little formality

2. **Modelling**
   - Choosing the more adequate way
   - Generalization Error
   - Bias-variance trade-off
   - Challenges to the Statisticians
   - Overfitting

3. Some Statistical Learning methods
   - Linear Model
   - Classification and Regression Trees
   - Support Vector Machines
   - *k*-Nearest Neighbor
   - Clustering

The goal is from a sample $\mathcal{L} = \{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots, (\mathbf{x_n}, y_n)\}$ estimate an unknown function f, finding an estimator $\widehat{f}$ such that

$$y \approx \widehat{f}(x)$$

for a new observation (x, y). As we say before, we suppose that observations of $\mathcal{L}$ are n independent realizations of a multivariate random variable (X, Y) of unknown distribution.

1. *Parametric methods*. The problem of estimating f is reduced to estimate some parameters, after assuming that f belongs to a certain family of functions.

   1) An assumption is made about the shape of the model, for example linear

      $$f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

      where we have to estimate $\beta_0, \beta_1, \ldots, \beta_p$.

   1. After the model is selected, it is trained from $\mathcal{L}$. For example, in the case of the linear model,

      $$\widehat{\beta} = (X'X)^{-1} X' Y$$

      where

      $$X = \begin{pmatrix} 1 & x_{11} & \ldots & x_{1p} \\ 1 & x_{21} & \ldots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \ldots & x_{np} \end{pmatrix}_{n \times (p+1)} , \quad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n, \; \widehat{\beta} = \begin{pmatrix} \widehat{\beta}_0 \\ \widehat{\beta}_1 \\ \vdots \\ \widehat{\beta}_p \end{pmatrix} \in \mathbb{R}^{p+1}$$

2) *Non parametric methods*. No assumption is made about the nature of f. In general, it allows covering a greater spectrum of forms for f, making the model more plausible to the true f. However, in general, a large number of observations is needed to obtain a performant model.
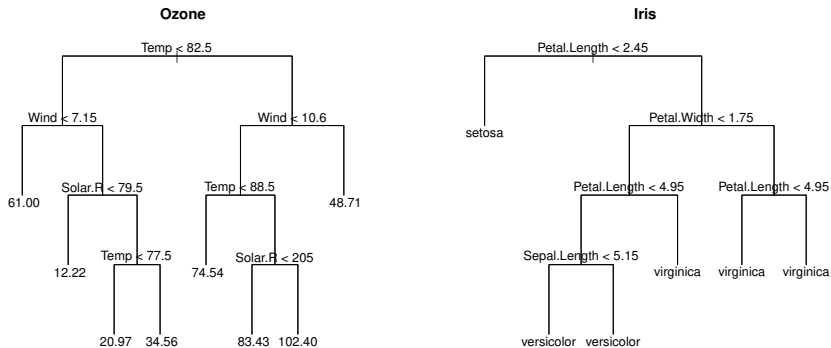
Figura: Classification and Regression Trees (Breiman, 1984)
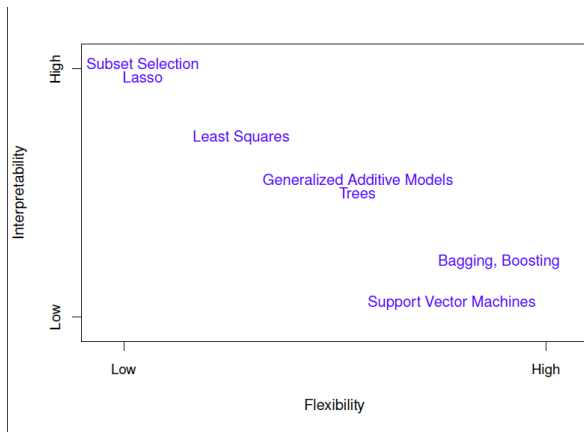
1. In regression quality of the fitting of a predictor can be evaluated by the *mean squared error MSE*:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \widehat{f}(x_i))^2$$

It will be small if the predictions are close to the true response values and large if for some observations the prediction and the label are very different.

1. In regression quality of the fitting of a predictor can be evaluated by the *mean squared error* MSE:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \widehat{f}(x_i) \right)^2$$

It will be small if the predictions are close to the true response values and large if for some observations the prediction and the label are very different.

However, evaluating the performance of the model on the data with which it has been trained, is not very interesting, or at least it is not as interesting as evaluating it on fresh data, which were not used for the estimation of $\widehat{f}$.

1. In regression quality of the fitting of a predictor can be evaluated by the *mean squared error* MSE:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \widehat{f}(x_i))^2$$

It will be small if the predictions are close to the true response values and large if for some observations the prediction and the label are very different.

However, evaluating the performance of the model on the data with which it has been trained, is not very interesting, or at least it is not as interesting as evaluating it on fresh data, which were not used for the estimation of $\widehat{f}$.

The performance of $\widehat{f}$ (construct over $\mathcal{L}$) is evaluated on a *testing set* $\mathcal{T} = \{(\mathbf{z_1}, u_1), (\mathbf{z_2}, u_2), \ldots, (\mathbf{z_s}, u_s)\}$ computing the *test*-MSE (generalization error):

$$\frac{1}{s} \sum_{i=1}^{s} (u_i - \widehat{f}(z_i))^2$$

1. In regression quality of the fitting of a predictor can be evaluated by the *mean squared error* *MSE*:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \widehat{f}(x_i))^2$$

It will be small if the predictions are close to the true response values and large if for some observations the prediction and the label are very different.

However, evaluating the performance of the model on the data with which it has been trained, is not very interesting, or at least it is not as interesting as evaluating it on fresh data, which were not used for the estimation of $\widehat{f}$.

The performance of $\widehat{f}$ (construct over $\mathcal{L}$) is evaluated on a *testing set* $\mathcal{T} = \{(\mathbf{z_1}, u_1), (\mathbf{z_2}, u_2), \dots, (\mathbf{z_s}, u_s)\}$ computing the *test*-MSE (generalization error):

$$\frac{1}{s} \sum_{i=1}^{s} (u_i - \widehat{f}(z_i))^2$$

In practice, original data set is divided in two parts: the first, $\mathcal{L}$, usually $2/3$, to train the model, and the remaining $1/3$, $\mathcal{T}$, to test it. Also in this way, the overfitting is avoided

1. In regression quality of the fitting of a predictor can be evaluated by the *mean squared error* MSE:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \widehat{f}(x_i))^2$$

It will be small if the predictions are close to the true response values and large if for some observations the prediction and the label are very different.

However, evaluating the performance of the model on the data with which it has been trained, is not very interesting, or at least it is not as interesting as evaluating it on fresh data, which were not used for the estimation of $\widehat{f}$.

The performance of $\widehat{f}$ (construct over $\mathcal{L}$) is evaluated on a *testing set* $\mathcal{T} = \{(\mathbf{z_1}, u_1), (\mathbf{z_2}, u_2), \ldots, (\mathbf{z_s}, u_s)\}$ computing the *test*-MSE (generalization error):

$$\frac{1}{s} \sum_{i=1}^{s} (u_i - \widehat{f}(z_i))^2$$

In practice, original data set is divided in two parts: the first, $\mathcal{L}$, usually $2/3$, to train the model, and the remaining $1/3$, $\mathcal{T}$, to test it. Also in this way, the overfitting is avoided

2. In classification the error is measured with the misclassified rate:

$$\frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{y_i \neq \widehat{y_i}\}}$$

where $\widehat{y_i}$ is the class prediction of f for observation i.

If we assume that $y = f(x) + \epsilon$, it is possible to prove that the expected value of the MSE for a fixed test value x_0, can be decomposed as:

$$\mathbb{E}\big(y_0 - \widehat{f}(x_0)\big)^2 = \mathsf{Var}\big(\widehat{f}(x_0)\big) + \big[\mathsf{Sesgo}\big(\widehat{f}(x_0)\big)\big]^2 + \mathsf{Var}(\epsilon)$$

If we assume that $y = f(x) + \epsilon$, it is possible to prove that the expected value of the MSE for a fixed test value x_0, can be decomposed as:

$$\mathbb{E}\big(y_0 - \widehat{f}(x_0)\big)^2 = \mathsf{Var}\big(\widehat{f}(x_0)\big) + \big[\mathsf{Sesgo}\big(\widehat{f}(x_0)\big)\big]^2 + \mathsf{Var}(\epsilon)$$

- As $\mathsf{Var}\big(\widehat{f}(x_0)\big)$ and $\big[\mathsf{Sesgo}\big(\widehat{f}(x_0)\big)\big]^2$ are non negatives, it follows that $\mathbb{E}\big(y_0 - \widehat{f}(x_0)\big)^2$ has as lower bound $\mathsf{Var}(\epsilon)$.
- We call *variance* to the amount that varies $\widehat{f}$ if we change the training set (different set of workouts produce different $\widehat{f}$). Under ideal conditions, the estimate of f does not change much if we change the training sets. In general, very flexible statistical models (with many parameters) have high variance. For example in the case of simple linear regression, when we change an element of the data set, the estimator does not vary so much. On the other hand if the model is very adjusted, changing a point produces a significant change in the estimation.
- *Bias* refers to the modelling error: explaining a real and complicated problem by a simpler mathematical model. For example, linear models assume that there is a linear relationship between Y and explanatory variables $X_1, \ldots, X_p$ which clearly has little chance of happening, so the bias will be important. In general, flexible statistical methods have a little bias.

Lineal (g1)

Polinomio g15

Baja Varianza
Gran sesgo
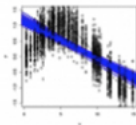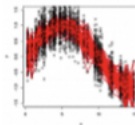
Alta Varianza
Bajo sesgo

$$\text{Var}[X] = \text{E}[X^2] - \text{E}[X]^2$$
$$y = f + \epsilon$$

$$\text{Bias}\big[\hat{f}(x)\big] = \text{E}\big[\hat{f}(x) - f(x)\big]$$
$$\text{Var}\big[\hat{f}(x)\big] = \text{E}[\hat{f}(x)^2] - \text{E}[\hat{f}(x)]^2$$

$$
\begin{aligned}
\text{E}\big[(y - \hat{f})^2\big] &= \text{E}[y^2 + \hat{f}^2 - 2y\hat{f}] \\
&= \text{E}[y^2] + \text{E}[\hat{f}^2] - \text{E}[2y\hat{f}] \\
&= \text{Var}[y] + \text{E}[y]^2 + \text{Var}[\hat{f}] + \text{E}[\hat{f}]^2 - 2f\text{E}[\hat{f}] \\
&= \text{Var}[y] + \text{Var}[\hat{f}] + (f^2 - 2f\text{E}[\hat{f}] + \text{E}[\hat{f}]^2) \\
&= \text{Var}[y] + \text{Var}[\hat{f}] + (f - \text{E}[\hat{f}])^2 \\
&= \text{Var}[y] + \text{Var}[\hat{f}] + \text{E}[f - \hat{f}]^2 \\
&= \sigma^2 + \text{Var}[\hat{f}] + \text{Bias}[\hat{f}]^2 \\
&= error\ irreducible + varianza(\hat{f}) + Sesgo^2 \hat{f}
\end{aligned}
$$

Several estimators (smoothing splines) are considered for different data sets (example extracted of James, Witten, Hastie and Tibshirani book).
Example 1. On the left hand three estimators with different flexibility adjusting the same data points and on the right hand the MSE curve of the flexibility on the training set (grey) and on a generalization set (red).
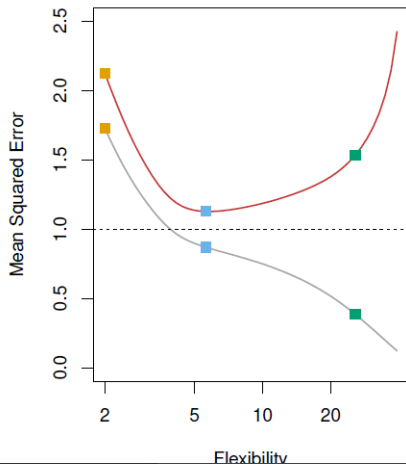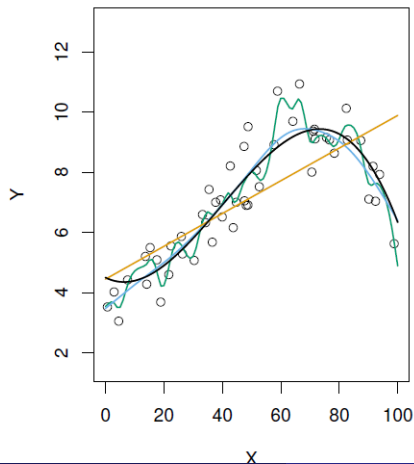
Example 2. On the left hand three estimators with different flexibility adjusting the same data points and on the right hand the MSE curve of the flexibility on the training set (grey) and on a generalization set (red).
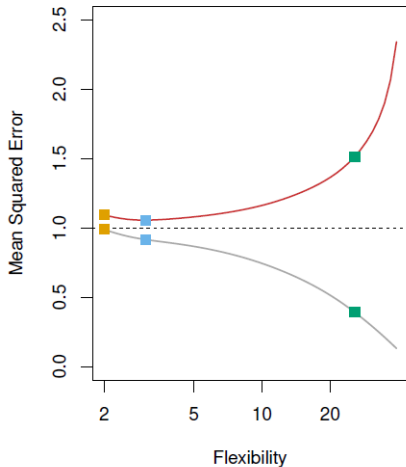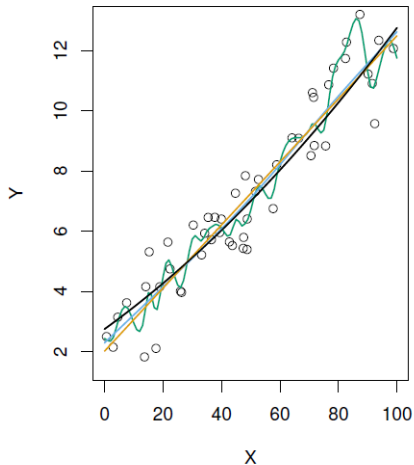
Example 3. On the left hand three estimators with different flexibility adjusting the same data points and on the right hand the MSE curve of the flexibility on the training set (grey) and on a generalization set (red).
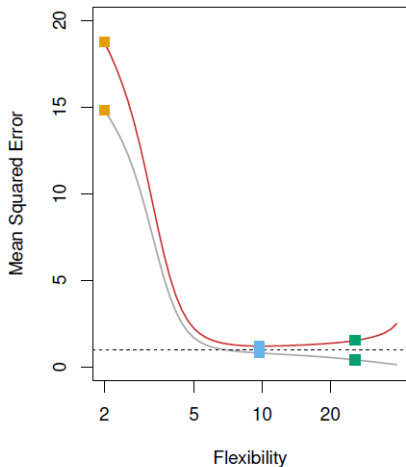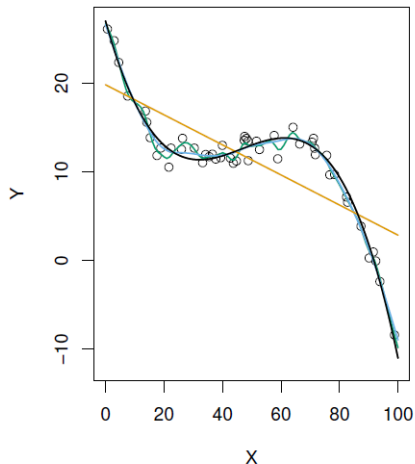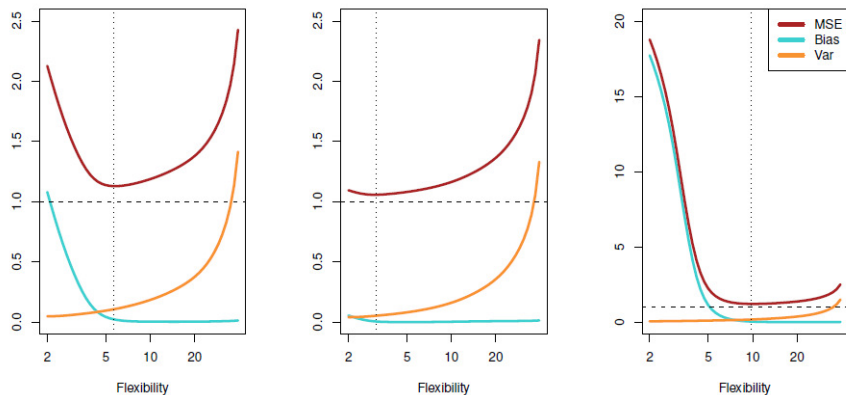
Figura: The three graphs refer to the MSE, bias and variance curves of three previous examples

The choice of the model will also be important to consider it a classification problem:
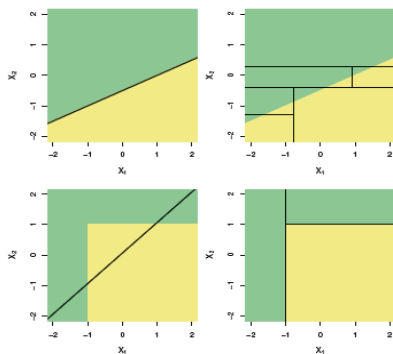
FIGURE 8.7. Top Row: *A two-dimensional classification example in which the true decision boundary is linear, and is indicated by the shaded regions. A classical approach that assumes a linear boundary (left) will outperform a decision tree that performs splits parallel to the axes (right). Bottom Row: Here the true decision boundary is non-linear. Here a linear model is unable to capture the true decision boundary (left), whereas a decision tree is successful (right).*

1. Data Complexity: involves many variables which are often related in complex (nonlinear) ways.
2. Big Data (datasets with large number of observations, large number of variables, large number of observations and variables).
3. Feature Selection: many features are available but some are redundant, leading to the feature selection or dimension reduction problem.
4. Optimization: many methods involve finding the "best" parameters values by solving complex and large (containing many parameters) optimization problems. Therefore, efficient optimization techniques are required.
5. Visualization: much harder in a high dimensional space.
6. Curse of dimensionality.
   In high dimension, the points are very far one of the other. Suppose we send out a hypercubical neighborhood about a target point to capture a fraction r of the observations. Since this corresponds to a fraction r of the unit volume, the expected edge length will be $e_p(r) = r^{1/p}$. In ten dimensions $e_{10}(0{,}01) = 0{,}63$ and $e_{10}(0{,}1) = 0{,}80$, while the entire range for each input is only 1.0. So to capture 1 % or 10 % of the data to form a local average, we must cover 63 % or 80 % of the range of each input variable. Such neighborhoods are no longer "local". Reducing r dramatically does not help much either, since the fewer observations we average, the higher is the variance of our fit.

Overfitting

Notice that a very simple model will probably have a high modelling error and we will not learn too much from the data (underfitting) whereas a model with many parameters will have a high statistical error (overfitting).
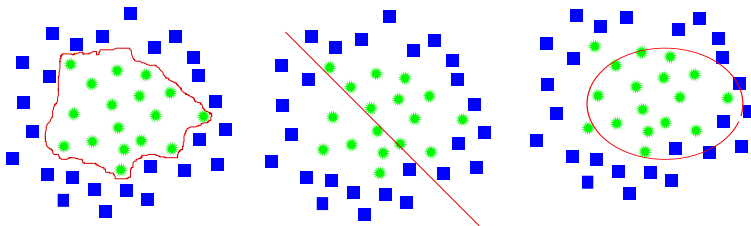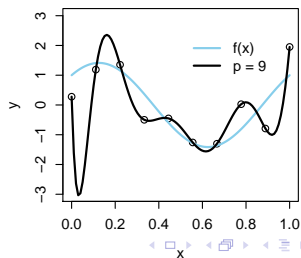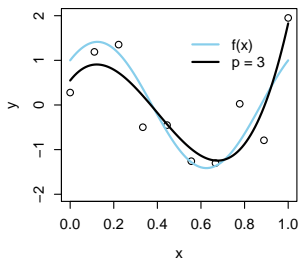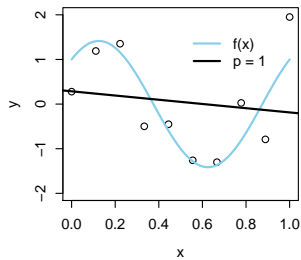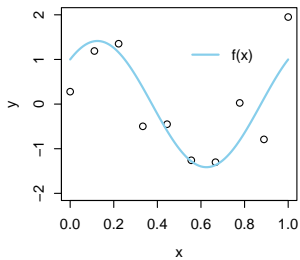We must achieve a compromise between both errors, in such a way that the "generalization error" is the least as possible.

Overfitting

Notice that a very simple model will probably have a high modelling error and we will not learn too much from the data (underfitting) whereas a model with many parameters will have a high statistical error (overfitting).
We must achieve a compromise between both errors, in such a way that the "generalization error" is the least as possible.
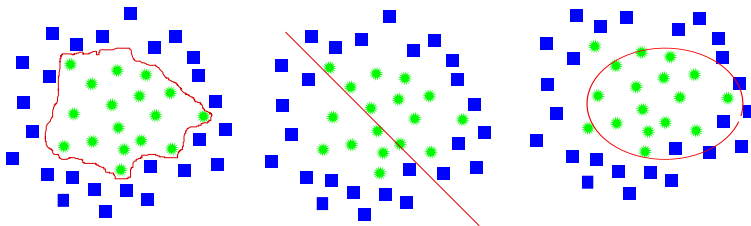
Notice that a very simple model will probably have a high modelling error and we will not learn too much from the data (underfitting) whereas a model with many parameters will have a high statistical error (overfitting).

We must achieve a compromise between both errors, in such a way that the "generalization error" is the least as possible.

To avoid overfitting, the predictor performance (classification error, mean quadratic error) is evaluated with a new sample called the evaluation sample, independent of the training sample.

Other ways to evaluate the predictor: cross validation, bootstrap.

Plan

Data: $\mathcal{L} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.

Data: $\mathcal{L} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.
We look for the line $y = ax + b$ that passes as close as possible to the data.

We find a and b that minimize the sum of squared errors

$$\sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} \big(y_i - (ax_i + b)\big)^2$$

The simple linear regression model is

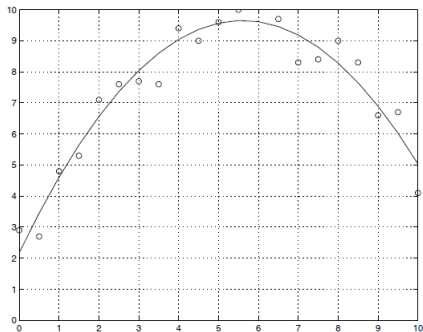$$y_i = \underbrace{ax_i + b}_{y_{est}} + \varepsilon_i, \ \forall \, i = 1, \ldots, n$$

Linear Model: method of least squares

The above method can be easily extended.

Linear Model: method of least squares

The above method can be easily extended.
For example the parabola that adjusts a set of points:

Linear Model: method of least squares

The above method can be easily extended.
For example the parabola that adjusts a set of points:

$$y = a + bx + cx^2$$

(linear model on the coefficients!)

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \end{pmatrix}$$

Multiple Linear Regression

Now we want to predicto a real random variable $Y \in \mathbb{R}$ from d real variables $X_1, \ldots, X_d$. We consider model:

$$f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_d X_d$$

As in simple linear regression, if $\mathcal{L} = \{(\mathbf{x_1}, y_1,), \ldots, (\mathbf{x_n}, y_n)\}$ is the data set, we look at a vector

$\beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{pmatrix} \in \mathbb{R}^{d+1}$ that minimizes

$$\sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{i1} + \cdots + \beta_d x_{id}) \right)^2$$

Observe that $\sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{i1} + \cdots + \beta_d x_{id}) \right)^2 = ||Y - X\beta||^2$ so we have a linear algebra problem:

$$X = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1d} \\ 1 & x_{21} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} \end{pmatrix}_{n \times (d+1)} \quad , \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n, \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{pmatrix} \in \mathbb{R}^{d+1}$$

whose solution is given by $(X^t X)\beta = X^t \mathbf{y}$.

Classification and Regression Trees (CART)

Classification **A**nd **R**egression **T**rees (Breiman 1984).

Two types of trees: regression trees to predict continuous variables and classification trees to predict categorical variables.

The tree is constructed from binary partitions with respect to the coordinates of the data. For example if the variables are $X_1, \ldots, X_d$, the cut condition for the data will be of type $X_2 < c$ or $X_2 \geq c$ if X_2 is continuous or $X_2 \in \mathcal{A}$ or $X_2 \notin \mathcal{A}$ if X_2 is categorical.

Three steps:

1. Binary separation of the data of each node in two subnodes according to some criterion;

Classification and Regression Trees (CART)

Classification **A**nd **R**egression **T**rees (Breiman 1984).

Two types of trees: regression trees to predict continuous variables and classification trees to predict categorical variables.

The tree is constructed from binary partitions with respect to the coordinates of the data. For example if the variables are $X_1, \ldots, X_d$, the cut condition for the data will be of type $X_2 < c$ or $X_2 \geq c$ if X_2 is continuous or $X_2 \in \mathcal{A}$ or $X_2 \notin \mathcal{A}$ if X_2 is categorical.

Three steps:

1. Binary separation of the data of each node in two subnodes according to some criterion;
2. Decision of the size of the tree: stop and prune criteria

Classification and Regression Trees (CART)

Classification **A**nd **R**egression **T**rees (Breiman 1984).

Two types of trees: regression trees to predict continuous variables and classification trees to predict categorical variables.

The tree is constructed from binary partitions with respect to the coordinates of the data. For example if the variables are $X_1, \ldots, X_d$, the cut condition for the data will be of type $X_2 < c$ or $X_2 \geq c$ if X_2 is continuous or $X_2 \in \mathcal{A}$ or $X_2 \notin \mathcal{A}$ if X_2 is categorical.

Three steps:

1. Binary separation of the data of each node in two subnodes according to some criterion;
2. Decision of the size of the tree: stop and prune criteria
3. Assigning a class or value to terminal nodes.

Example: Iris

Goal: Predict the species of the iris flower.

Data: 150 flowers

Dependent variable: Species (setosa, virginica, versicolor)

Independents variables: Sepal Length, Petal Length, Sepal Width, Petal Width
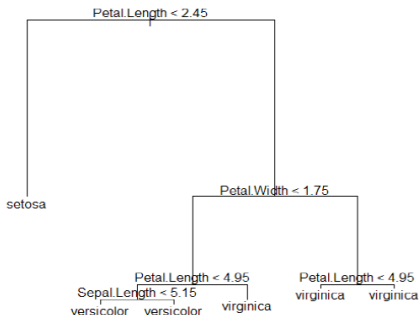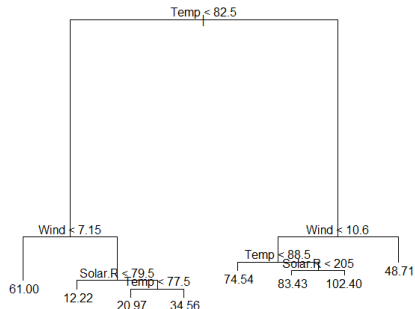
Example: airquality

Goal: Predict the ozone level in New York.

Data: 153 days

Dependent Variable: ozone level

Independents variables Date, Solar Radiation, Wind and Temperature

Easy to interpret, but ... very unstable: a small change in the sample leads to completely different results.

Easy to interpret, but ... very unstable: a small change in the sample leads to completely different results.

Aggregation Methods:

1. **Bagging** (Breiman, 1996): average of several trees based on data re-samples.
2. **Random Forests** (Breiman, 2001): combines the Bagging and CART algorithms.
3. **Boosting** (Freund and Shapire, 1997): weighted average of trees. The weighting takes into account the performance of each tree in each stage of the algorithm.

Support Vector Machines (SVM)

In the classification context, SVM (Vapnik, 1995) is a method that consists of finding a curve that separates the data as best as possible.

In the classification context, SVM (Vapnik, 1995) is a method that consists of finding a curve that separates the data as best as possible.
If the data are linearly separable:
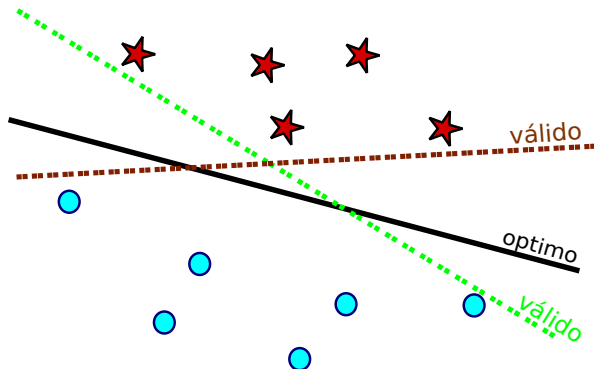
If the data are not linearly separable, we transform them to a space where they are:
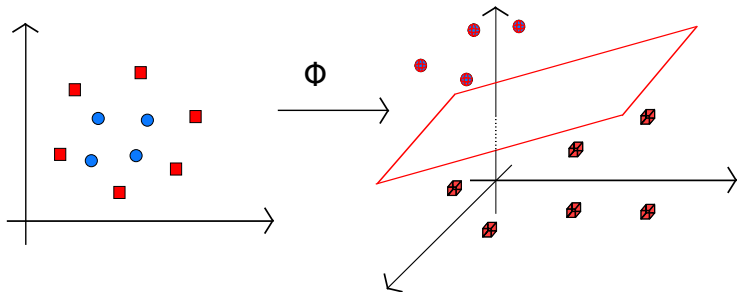
In *k*-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.

In *k*-NN regression, the output is the property value for the object. This value is the average of the values of its k nearest neighbors.
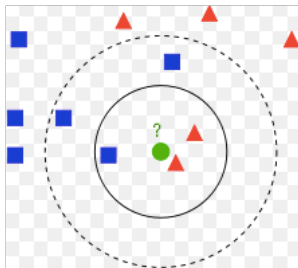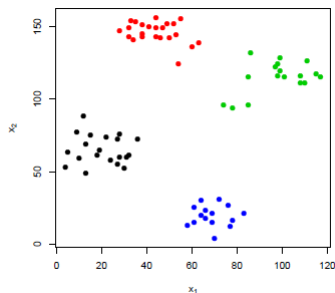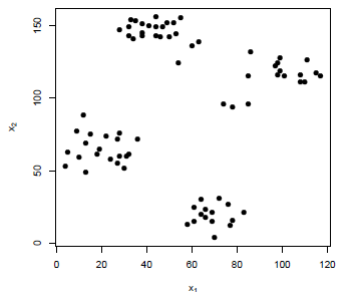
Here we have a data set but without output, that is, $\mathcal{L} = \{x_1, \ldots, x_n\}$ where $x_i \in \mathbb{R}^d$ and we want to create K different homogeneous groups.

Here we have a data set but without output, that is, $\mathcal{L} = \{x_1, \ldots, x_n\}$ where $x_i \in \mathbb{R}^d$ and we want to create K different homogeneous groups.
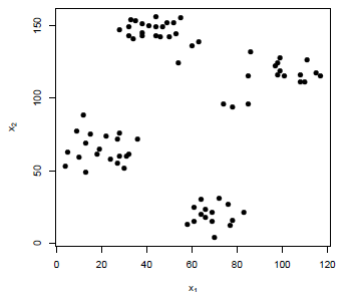
Here we have a data set but without output, that is, $\mathcal{L} = \{x_1, \ldots, x_n\}$ where $x_i \in \mathbb{R}^d$ and we want to create K different homogeneous groups.

- D. Peña, *Análisis de Datos Multivariantes*, Mac Graw Hill, 2002.
- A. I. Izenman, *Modern Multivariate Statistical Techniques*, Springer, 2008.
- G. James, D. Witten, T. Hastie, R.Tibshirani, An Introduction to Statistical Learning with Applications in R. Springer Texts in Statistics, 2013
- Devroye,L., Györfi,L. and Lugosi,G. A Probability Theory of Pattern Recognition. Springer, 1996
- Hastie, Tibshirani, Friedman, The Elements of Statistical Learning, Springer, 2001.