Statistical methods in the big data era
Hotel prices visualization

2019

Eustat
EUSKAL ESTATISTIKA ERAKUNDEA
INSTITUTO VASCO DE ESTADÍSTICA
www.eustat.es

Organismo Autónomo del
EUSKO JAURLARITZA
GOBIERNO VASCO

Development:
EUSTAT
Euskal Estatistika Erakundea
Basque Statistics Institute

Publication:
EUSTAT
Euskal Estatistila Erakundea
Basque Statistics Institute
Donostia-San Sebastián 1,
01010 Vitoria-Gasteiz

Administration of the Basque Country

First Edition:
I/2019

Printing and Bounding:
Printing and Reprography Service. Basque Government.

ISBN: 978-84-7749-490-4

Legal Deposit: VI-155/19

Statistical methods IN THE ERA OF BIG DATA

Ander Juarez Mugarza

EUSKAL ESTATISTIKA ERAKUNDEA

BASQUE STATISTICS INSTITUTE

Donostia-San Sebastián, 1

01010 VITORIA-GASTEIZ

Tel.: 945 01 75 00

Fax: 945 01 75 01

E-mail: eustat@eustat.eus

www.eustat.eus

Contents

Preamble

This Technical Manual sets out the results of the work carried out on machine learning, thanks to the grant awarded in 2017 by the Basque Statistics Institute (Eustat) to provide training and conduct research into statistical and mathematical methodologies.

This paper seeks to achieve two main objectives. The first is to give an explanation of the main techniques currently used in the world of machine learning, and the second is to publish the results of the Big Data project carried out at the Basque Statistics Institute (Eustat) over the last two years. Accordingly, this publication will be divided into two main sections, each comprising a total of four chapters.

The first section will deal with the theoretical part, while providing various examples of the different methods. It will, as we mentioned, comprise four chapters. The first two look at the two main families of machine learning, namely supervised and unsupervised learning. The third chapter will cover neural networks, which have had a high impact in recent years. Lastly, in the fourth chapter we will discuss the test-validation-training divisions that are often carried out with the algorithms used.

As for the second section, we will take another look at the various steps followed throughout the working process, while exploring the theory behind them. The second section of the project analyses the variation in prices of hotel establishments in the Basque Country. To conduct this analysis, web platforms were used to obtain the prices of hotels in the Basque Country using a web scraping procedure developed by Eustat. These data were combined with those from the *Survey on Tourist Establishments*, thus achieving a more solid database.

On the basis of these data, an application was designed that intuitively shows the variation in prices of Basque Country hotel establishments over time, as well as the correlation between the hotels' prices and their surroundings. This application was presented at the BigSurv18 conference held in Barcelona in November 2018, organised by the European Survey Research Association (ESRA).

Furthermore, I would like to take this opportunity to thank everyone in the Eustat Methodology, Innovation and R&D Department for their support and the trust that they have placed in me over these last two years. My special thanks go to Anjeles Iztueta, Jorge Aramendi, Elena Goni, Inmaculada Gil and Marina Ayestarán. I would also like to thank all my colleagues in the Eustat family, including my workmate Asier Badiola, for creating the great atmosphere that has helped us in our work. Finally, I give my thanks to my family, Garazi and especially my grandad Alejandro Mugarza for their constant support.

Machine learning techniques

In the first section of this paper, we will revisit machine learning techniques. Although the term seems recent, it was originally coined in 1959. Researchers at the time thought that it would be interesting to develop algorithms capable of learning from data and making predictions, thereby taking the initial steps towards devising the techniques needed for machine learning today.

As we said, the machine learning process involves learning from data and using it to make future predictions. It can be divided into two major families: supervised and unsupervised learning. The goal of the first is to predict future data based on past data. The second, meanwhile, focuses on data-based learning, obtaining information that might sometimes be hidden or that cannot be seen simply by looking, and drawing connections between the elements being analysed. We will discuss these two families in the first two chapters.

The third will deal with neural networks that currently have a high impact. As with the term “machine learning”, to find the origin of neural networks we must step back in time to 1943, since although the methods seem very modern, the definition has been around for a number of years. Given that neural networks can be used for several tasks – both for supervised and unsupervised learning – it was decided to dedicate a specific chapter to them.

Finally, the supervised learning techniques (including the neural networks used for this type of problem) should also be tested, in that predictions will be made based on them. The last chapter of this section will therefore discuss the techniques for dividing the available data in order to develop algorithms and conduct different tests.

1. Supervised learning

This type of technique is applied when an output value or label for each of the elements is expected, i.e. when a series of output values is to be inferred, based on certain input data. The output data can be quantitative (regression) or qualitative (classification). The main goal will never be to learn about the data, but rather establish a rule that correctly interlinks the input with the output values. Thus, by obtaining new data at a future point, their corresponding output value can be inferred.

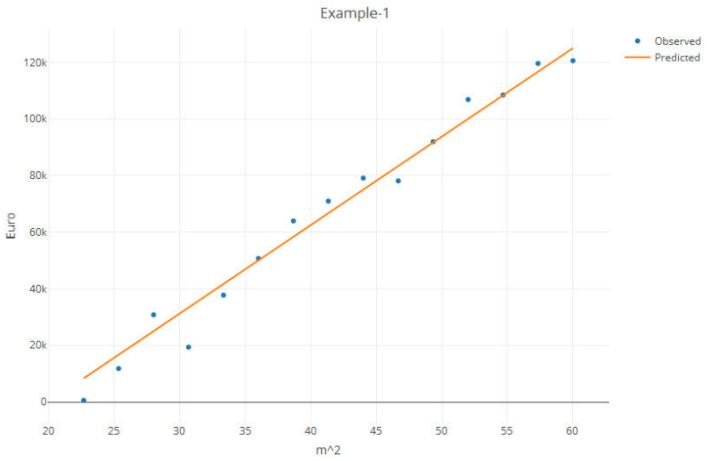
The following supervised learning methods will be analysed:

1. Linear regression
2. Logistic regression
3. KNN
4. Perceptron
5. SVM
6. Decision trees

1.1 Linear regression

Linear regression, when considering the quantitative label $Y_1, .., Y_n$ for each element $X_1, \dots, X_n$ from the database, attempts to predict the corresponding quantitative label Y', when obtaining a new element X'.

Let us suppose that we know the number of square metres and the prices of different homes, and that we made a graph to show the correlation between the two, thus obtaining the blue points in the graph on the right. As we can see, the points show the linear correlation between the number of square metres and the price. We would therefore be able to predict the price of a home based on the number of square metres it has. Linear regression attempts to fit each data element X_i with the label Y (orange line) through a linear function.



1.1.1 The algorithm

To predict the new Y' labels, as we can deduce from the name of the technique, the goal is to achieve the *linear* function $f(X, \theta)$, where $\theta = \{\theta_0, \dots, \theta_p\}$ represents the coefficient of the linear function, which is

$$Y \sim f(X, \theta) = \theta_0 + \sum_{i=1}^{p} \theta_i X + \epsilon$$

while ϵ represents the error that will always exist in practice.

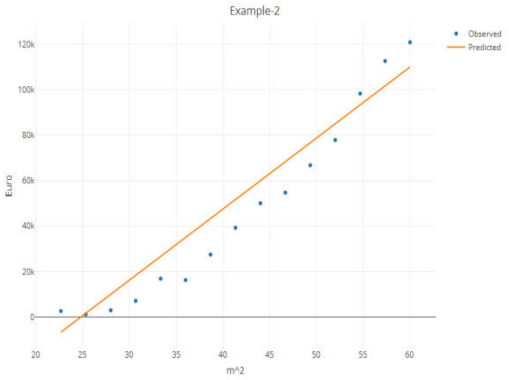
The optimal θ parameters of the linear regression are those which offer the lowest minimum error. When measuring the error, the most common practice is to measure the predicted value $f(X,\theta)$ and the difference between the label Y, for which the Euclidean technique will be used. The function for the value to be minimised is therefore:

$$Cost(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(f(X_i,\theta) - Y_i)^2 .$$

To minimise it, we can use the gradient descent algorithm. When it comes to applying the algorithm, the next step is repeated over and over again until convergence occurs, or until the pre-established maximum number of iterations has been exceeded:

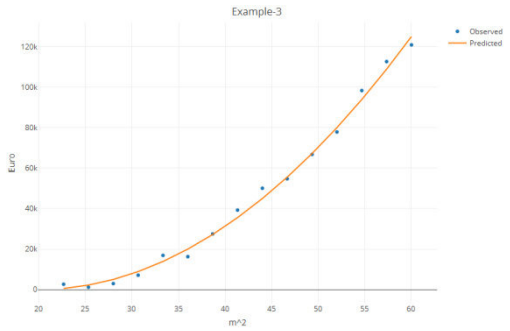
$\theta = \theta - \frac{\alpha}{2n}\sum_{i=1}^{n}\left(X_i^T\theta - Y_i\right)X_i$, $1 \le i \le n$ for each.

1.1.2 Variables in the non-linear relationship



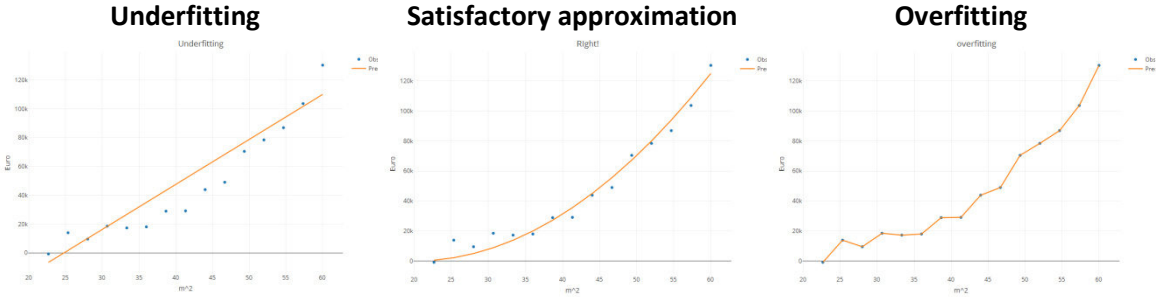
In some cases, it might be that the relationship between the variables is not linear, as we can see in the graph on the right. Linear regression can also be applied in these cases.

Let us suppose that our data have p different variables, in which case their powers can be used as new variables so that the linear regression algorithm better fits the data. Although with linear regression the θ_i coefficients have to be linear, the available data can also be fitted, if by doing so we improve the result. If, for example, we add the four powers from the previous case as variables, we obtain the result shown on the left. We can see a clear improvement. As well as applying different powers to the variables, several other functions can also be applied, such as the logarithmic, square root or exponential functions.



1.1.3 Regularisation

The model obtained by adding new variables can "learn" from our tests or input data, leading to greater error in future cases. This phenomenon, which should be avoided, is called overfitting. In the image below, we can see two extreme cases of both overfitting and underfitting. The latter occurs when our model does not adequately reflect the nature of the training points.



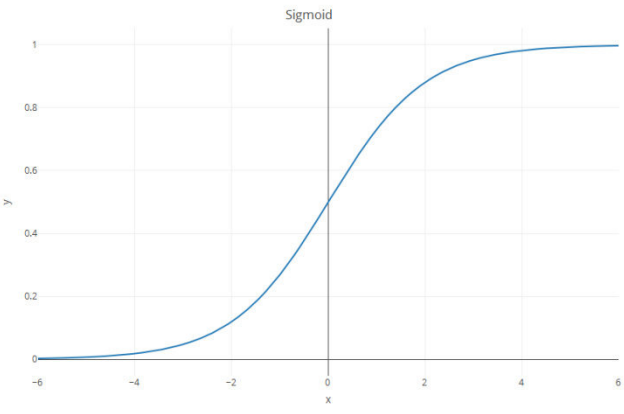
Variables can be eliminated to avoid overfitting. The constant λ, called the regularisation parameter, can also be added as a penalty for the parameters θ, with the result shown below.

$Cost(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(f(X_i,\theta) - Y_i)^2 + \lambda\sum_{j=1}^{p}\theta_j$ (Note that the penalty θ_0 does not exist)

This parameter increases the value of the value function as the θ_i parameter increases. Therefore, the higher its value, the closer the value of the θ_i ($i \neq 0$) elements will be to zero as the value function is minimised, and the regression will give a consistent answer, θ_0. Conversely, when the value of the parameter is reduced, the model obtained will look more like the non-regularised model. Lastly, where $\lambda = 0$, it will give the same result as the non-regularised model.

1.2 Logistic regression

It can be said that logistic regression is one of the most widely used classification techniques. Just as with all of these techniques, for each datum



$\{x_1, \dots, x_n\}$ of logistic regression we will have $\{y_1, \dots, y_n\}$ binary labels, giving two options such as yes/no, life/death, etc. The aim of the method is to predict the corresponding y' label for each of the new x' elements. In this case, the label that we want to estimate, or the positive label, shall be given the value 1, while the other will be 0.

Logistic regression, through the logistic function known as the sigmoid function, tries to find the correlation between variables of different elements and their labels. Having seen this correlation, the data with different labels are divided in a linear fashion. This method can be regarded as a special case of generalised linear regression.

1.2.1 The algorithm

Let us suppose that we have data in group X. The aim of this algorithm is to calculate the estimates for the parameter θ, and the size p, where the function $\sigma(x'^T\theta)$ represents the probability that the element x' has the label 1,

and where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid function.

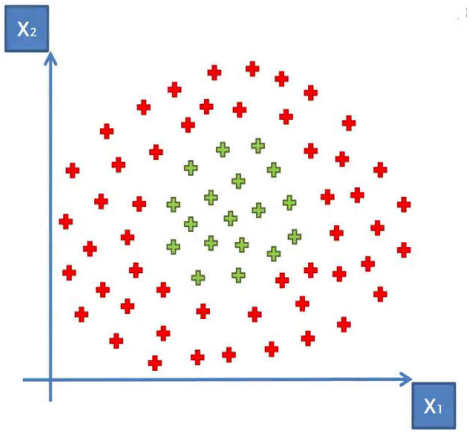
To obtain the optimal parameters for logistic regression, we must minimise the following cost function:

$$Cost(\theta) = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(\sigma(x_i^T\theta)) + (1-y_i)\log(1-\sigma(x_i^T\theta))) \ .$$

To do so, we can use the gradient descent, as well as a series of other techniques, such as the conjugate gradient, BFGS, L-BFGS, etc.

1.2.2 Non-linear limits

Given what we have seen up to this point, the data will generally not have these characteristics, even though we are able to disaggregate and predict different elements with linear limits. In our everyday work, we will more often find indivisible limits, as we can observe in the example on the right. In these cases, as we have seen with linear regression, it is worth assigning the powers of the variables analysed to the data, as new variables. Once this operation is complete, we can adjust our method to the data. The following video shows the progressive steps of the gradient descent algorithm, raising the data by the 5th degree power.



1.2.3 Regularisation

As with various supervised learning methods, the data used might be "studied" to construct the model and, when inputting data that differs from those entered previously, we might obtain an unsatisfactory result. To avoid this overfitting phenomenon, it is advisable to add a regularisation parameter to the model. Taking this into consideration, we will work with the following function:

$$Cost(\theta) = -\frac{1}{n}\sum_{i=1}^{n} y_i \log\left(\sigma\left(x_i^T\theta\right) + (1-y_i)\log\left(1-\sigma\left(x_i^T\theta\right)\right)\right) + \frac{\lambda}{2n}\sum_{j=1}^{p}\theta_j.$$

Despite the fact that this parameter may increase the error when it comes to designing the model, the results are improved in order to predict the labels of the new data sets. As we can see in the following video, although a limit that at first appeared very irregular has been attenuated, as the value of λ increases, the model tends to simplify, obtaining bad results. Great care should therefore be taken when selecting the value of the λ parameter.

1.2.4 Multiclass classification

While we have worked with data that have two different labels, we might want to work with elements that have three or more labels. In these cases, we are dealing with multiclass classification, for which One vs All would give the most intuitive solution. Let us suppose that we have a total of k different labels, i.e. $y \in \{e_1, \dots, e_k\}$. Binary logistic regression is applied in this case, adopting the label e_i as case 1, while the rest is taken as 0, thus creating k different models for each $i \in \{1, \dots, k\}$. Finally, the class attributed to the element will be the class with the maximum of the probabilities obtained in each model from the new data, i.e. e_j, where it would be $\mathrm{h}_{\theta_j} = \max_{i\in\{1,\dots,k\}}(h_{\theta_i}(x))$.

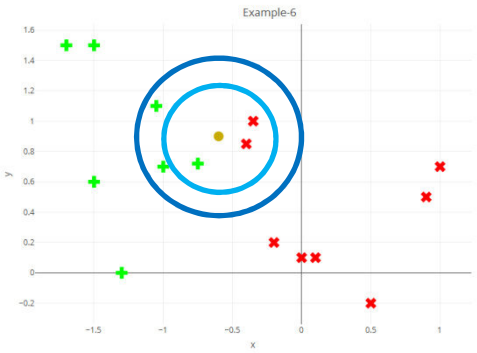
Aside from the One vs All method, there are other techniques including multimodal logistic regression and ordered logistic regression. The former deals with concepts similar to the One vs All technique, but instead of analysing all of the labels individually, it uses a label as an axis and analyses the rest with respect to it. The latter uses the ordered categories for the analysis.

1.3 K-nearest neighbours

One of the most intuitive of the classification techniques is K-nearest neighbours (KNN). Since this technique can be used for classification, our data will be n elements, each with their corresponding label.

The reasoning behind this is that similarly labelled elements tend to congregate in a similar manner. In this regard, when analysing a new element, the nearest points to it are taken into account. Once this step is complete, the most commonly occurring label will be the label of the new element.

As an example, let us suppose we dealing with the situation shown in the image on the right. We find elements labelled in two ways (green cross and red cross) and, when obtaining a new datum (yellow circle), we have to determine its label. Now, if we analyse its nearest $k = 3$ neighbours, we conclude that we must label the new element with a red cross. However, if we analyse the case of $k = 5$, we find a new green cross.



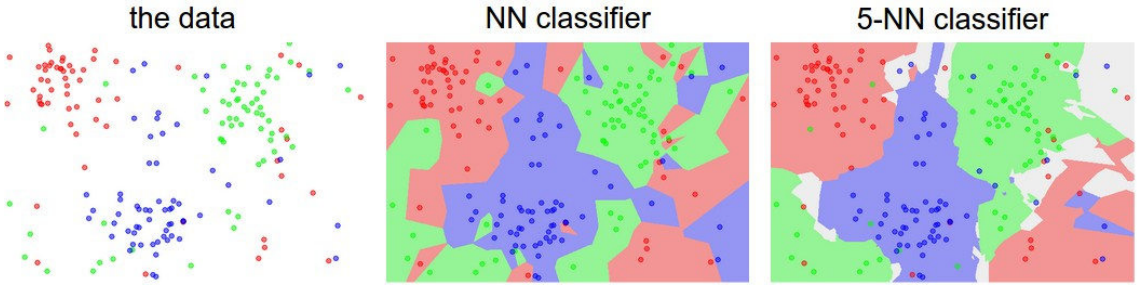
1.3.1 The algorithm

When analysing a new element with an unknown label, the distance between that element and the element with a known label is generally calculated through the Euclidean distance. To

do so, the elements analysed have to be quantitative variables. However, if we are dealing with qualitative variables of relevant data, another distance function can be defined. By way of an example, when analysing the signals, the correlation between the elements is considered, whereas to analyse strings of characters we count the elements to be added up, modified or eliminated to go from one word to another.

As we discussed distances, it would be most appropriate to normalise the variables so that they all have the same impact on the technique.

The only decision that we have to make when approaching this method is the number of "neighbours" to analyse. When deciding on the value of this parameter, we should consider the following: the lower the parameter, the better the results we will obtain; the higher the parameter, the more attenuated the result. It is therefore advisable to divide the data into two groups (training and test data). In this case, we will compare the elements in the test group with those in the training group for the different values of k, and then we will see whether it matches the label obtained. Once this operation is complete, we will select the k value with the least error.



1.3.2 Examples

Although KNN is a simple and intuitive technique, it yields good results for many classification problems. For example, it is used to predict handwritten digits or to predict the landscape of images obtained via satellite.

1.3.3 Pros and cons

First, we will consider the positive aspects of this technique:

- very simple and effective,
- it does not require training and adding new examples is straightforward,
- it is very simple to interpret.

However, there are also negative aspects:

- it is computationally expensive since, as the elements and variables from the database increase, its speed will slow,
- close attention should be paid to the scale.

1.4 Perceptron

The Perceptron algorithm is one of the most straightforward of all the classification techniques. The data we will analyse here will be quantitative variables, apart from their labels, which will be qualitative.

To apply this algorithm, the groups of points with different labels have to be linearly divisible. To put it another way, if the elements have two possible labels (let us suppose that the labels are 1 and –1), there must be a straight line or a plane dividing the elements labelled 1 and –1.

1.4.1 The algorithm

This technique tries to find the parameter θ (dimension vector $p \times 1$), where for each x_i element whose label is analysed will be $y_i = sign(x_i\theta)$. The following value function should be minimised:

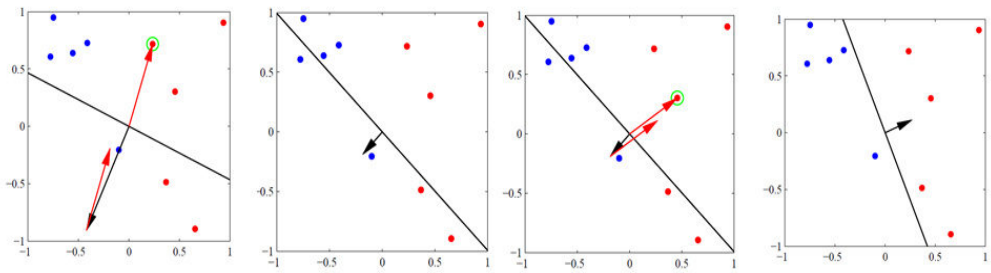
$$L = -\sum_{i \in M} y_i(\theta\, x_i),$$

where the group M is a group based on poorly classified indices, $y_i \neq sign(\theta\, x_i)$.

For this we use the stochastic gradient descent algorithm, in which all of the elements are analysed individually, rather than the elements having a combined impact on the negative direction of the gradient.

The algorithm follows these steps:

1. Start at $\theta^1 = 0$.
2. Find the incorrectly classified (x_i, y_i) elements (that satisfy $y_i \neq sign(\theta\, x_i)$.
3. If an element is found in the previous step, the θ parameter is updated as follows: $\theta^{t+1} = \theta^t + \rho y_i x_i$.
    a. Go back to step 2.
4. If no element is found in step 2, the algorithm terminates.



1.4.2 Considerations

- When the date is divisible, there are a lot of potential answers and an answer will be given depending on the initial values.
- Although a result may be achieved from a finite number of steps, the number can be high.

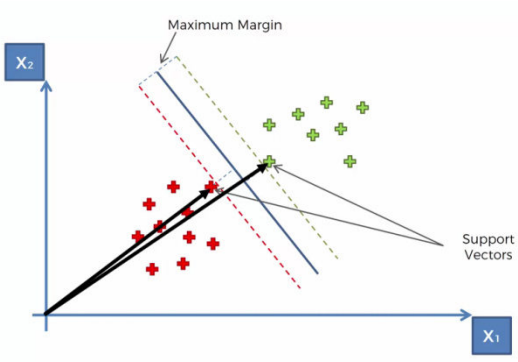
- If the date is not divisible, the method will never converge and the results will always follow a cyclical pattern.

1.5 Support-vector machine (SVM)

SVM is one of the more powerful classification algorithms and is quite common. It is used for classification, i.e. it seeks to learn the relationship between data and labels based on a set of labelled data. It can thus predict which group new unlabelled data belong to.

Let us suppose that we have a set of linearly divisible labelled data. This algorithm will attempt to divide the labelled groups differently (let us suppose that there are two possible labels) through a hyperplane, so that all of the data on one side of the hyperplane will have one label, and the rest of the data will have the other.

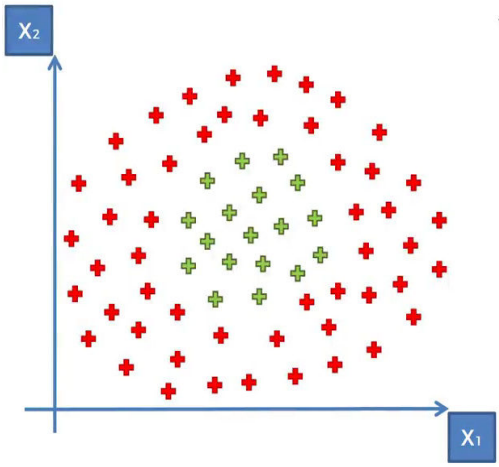
Other algorithms, such as Perceptron, are satisfied with finding the boundary that divides the labelled groups differently. The SVM algorithm, however, is known as a large margin classifier.



This classification method maximises the distance between groups and boundaries, as well as dividing the different groups.

When applying the SVM algorithm, we will have initial quantitative variables and, for each input datum, a qualitative variable labelling them.

1.5.1 Non-linear limits



The main problem with the SVM method is that the data have to be linearly divisible. Problems would therefore arise in most cases we find, as we can see in the image on the left.

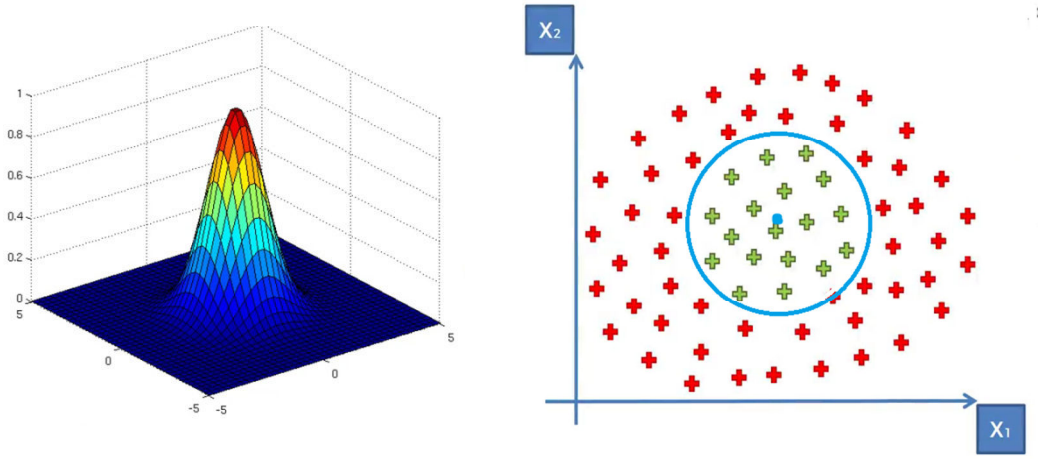
To overcome these issues, we can apply kernel functions to the data available. This being the case, using linearly indivisible points, we will obtain linearly divisible points, as we can see in this video.

The most widely used is the Gaussian kernel, also known as the RBF kernel:

$$K(\vec{x}, \vec{l^i}) = e^{-\left(\left\|\vec{x}+\vec{l^i}\right\|^2 / 2\sigma^2\right)}$$

1.5.1.1 The algorithm

To apply this kernel, the first thing we have to do is to select a centre $\vec{l^i}$, and a value for the variance, σ. Once this is complete, the function will show the distance between the points near to and far from the centre, and assign a value of zero to the distant points. Continuing with this example, and selecting the appropriate $\vec{l^i}$ and σ values, we obtain the following result:



Let us suppose that in the algorithm training group we have n $(\vec{x_1}, \dots, \vec{x_n})$ elements. One possibility for selecting the values for$\vec{l^i}$ would be to place the $\vec{l^i}$ point in the same place as each element, i.e. $\vec{l^1} = \vec{x_1}, \dots, \vec{l^n} = \vec{x_n}$. In this case, we would have n centres and thus n kernels.

In this case, we could define the following value function to train the algorithm:

$$C \sum_{i=1}^{n} [y^i cost_1(\theta^T f^{(i)}) + (1 - y^i) cost_0(\theta^T f^{(i)})] + \frac{1}{2} \sum_{j=1}^{n} {\theta_j}^2$$

where

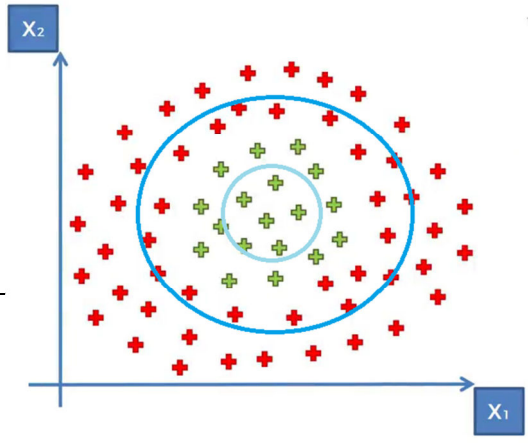
$$cost_1(z) = -\log\left(\frac{1}{1+e^{-z}}\right) \; and \; cost_0(z) = -\log\left(1 - \frac{1}{1+e^{-z}}\right)$$

and

are $f^{(i)} = \begin{pmatrix} {f_0}^{(i)} = 1 \\ {f_1}^{(i)} = K(\vec{x_i}, \vec{l^1}) \\ \vdots \\ \vdots \\ {f_n}^{(i)} = K(\vec{x_i}, \vec{l^n}) \end{pmatrix}$ and θ represents the vector formed of the parameters that we wish to learn.

1.5.1.2 C and values σ

When we apply the RBF kernel, we should take care when selecting the σ value. If we choose too low a value, our algorithm will be too strict and it will



exclude a series of positive data (the light blue circle). On the other hand, if we select too high a value, we will classify too many points as positive cases (dark blue circle).

The C value is a regulation parameter. If the value used is too high, the algorithm will learn the examples from the database rather than the relationships between them and thus fail with future examples. Furthermore, if the value used is too low, it might mean that the algorithm does not learn the predictions correctly when training, or in future cases.

1.5.1.3 Other types of kernel

While the RBF kernel is one of the most common, we can also use other types, such as:

1. The linear kernel/without kernel: if y=1 and it is only $\theta^T X \geq 0$.
2. Polynomial: $K(x, l) = (ax\text{^}T\ l + c)^{\ m}$ where a and c represent constants and m is an ordinary number.
3. Laplacian: $K(x, l) = \exp(-\alpha \|x - l\|)$ where $\alpha > 0$.

1.5.2 When to use SVM (LOGISTIC REGRESSION VS. SVM)

Let us suppose that the number of data obtained is n and that each datum has p quantitative variables, so that:

- If we have numerous variables, i.e. if p >>> n, it is advisable to use logistic regression or SVM without kernel.
- If p is relatively low (1-1000) and we have lots of data, i.e. if n falls within the 10-10,000 range (1:10), it would be appropriate to use SVM with a Gaussian kernel.
- If n is very high (>50,000) and p is relatively low (between 1 and 1,000), the performance of SVM would be very slow. Therefore, as well as adding or creating more variables, it is advisable to use logistic regression or SVM without kernel.

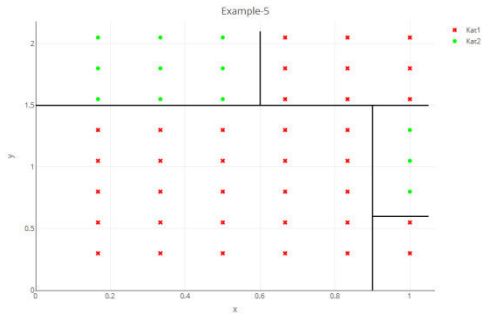
1.6 Decision trees

Decision tree-based methods divide the space of the data analysed and assign a specific label or value to each part. While these methods can be used for both regression and classification, we will limit ourselves to the latter here.

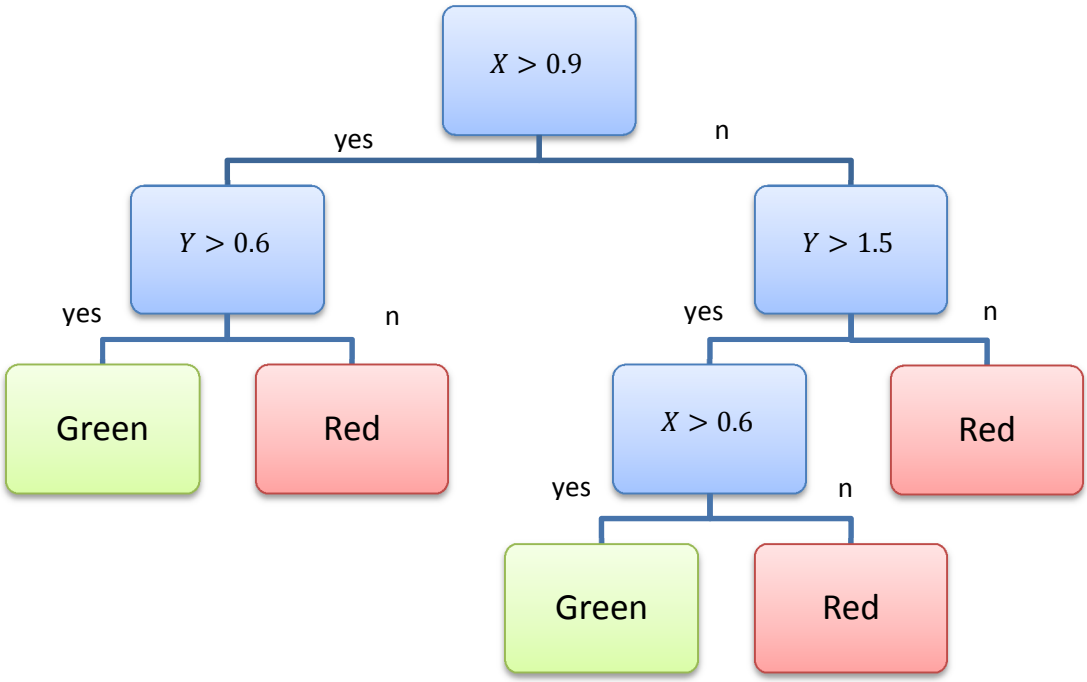
Different fields will be defined in a recursive and binary fashion to make the limits easier to interpret.

As an example, let us suppose that the elements of the database that we are analysing have two quantitative variables, X and Y, as well as a label comprising two values, Z,. Furthermore, let us suppose that the data are represented with respect to the variables X and Y, and each point receives a colour depending on their label, giving us the graph on the right.

As we can see in the image, the points with the green label are in isolated zones, whereas the elements with the red label occupy the rest of the space. The algorithm will divide the set of data into two, seeking to isolate a single type of label in each part with respect to the X or Y variable. Accordingly, the same process will be followed in the subsets obtained, until we have a single label in each zone or until the error obtained is lower than a predefined value. In this example, we obtain the result shown in the graph on the left. As we can see, the limits between the elements with the green and red labels are clearly differentiated.



When a new datum without a label is obtained, we only need to analyse the values for the X and Y variables to find out what zone it belongs to. With that in mind, we will be able to predict its label. The result obtained above can be summarised in a tree graph for handling in a more intuitive manner. Below we can see the tree for the example we are analysing.



1.6.1 The algorithm

Let us suppose that the data set that we are going to analyse has a total of n variables, i.e. $X_1, \dots, X_n$. In all its iterations, the algorithm should make two decisions: the j variable that will be used to divide data and the split point s. There is no fixed rule for this. The algorithm tests the various points of the different variables and selects the split point t_1 that minimises the error. Different values are used to calculate it. We will define three measurements below: *classification error, Gini index* and *entropy.* Let us suppose that, in the zone we are analysing, the p_i value represents the proportion of elements with the label number i, therefore:

1. **Classification error:** $1 - \max_{1 \le i \le k} p_i$.
2. **Gini index:** $1 - \sum_{i=1}^{k} p_i^2$.
3. **Entropy:** $-\sum_{i=1}^{k} p_i \ln(p_i)$.

The Gini index and entropy are most commonly used for this. As we can see, all of the values will be equal to 0 if, and only if, the zone we analyse corresponds to the zone with a single label type.

Once the split point has been found, the set of data are divided into two subsets and the same process is followed. However, we should pay close attention to the split point we make. Trees that are too large will learn the specific database analysed (overfitting), whereas trees that are too small will yield unsatisfactory results. The pruning technique is used to prevent this. First, the tree grows until it reaches the pre-established size. It is then made smaller until the optimal size is reached. To decide what subtree of the original tree we will keep, we will add a weight depending on the size of the tree to the tree's error, and the result with the smallest value will be selected.

1.6.2 Advantages and disadvantages

This method offers a number of advantages, namely:

- it is straightforward to interpret,
- it can handle both quantitative and qualitative variables,
- it does not require data processing,
- it yields good results with large-scale databases.

However, it also has disadvantages:

- it can be unstable, and a small change in the data may lead to major changes in the results obtained,
- the algorithm is not NP-complete, i.e. a solution to the problem can be obtained in polynomial time, however, it is not as easy to find the globally optimal result of the problem.

1.7 Ensemble methods

When applying the theory behind all of the methods, the results might not be as satisfactory as we might have hoped, even if we try different parameters. When this occurs, it is useful to have good models based on techniques that obtain results that do not prove as satisfactory.

This is the goal of the various methods that we will look at below. Before embarking on an analysis of the techniques, to improve the results by combining different models, the error must be lower than 50%. To put it another way, it must yield better results than the randomly selected method.

1.7.1 Bootstrap

When applying the same method to the same set of training data, we will of course achieve the same results. Therefore, if we want to develop and combine different models with a view to improving the results, we should use different groups of training data. The Bootstrap technique can be used for this.

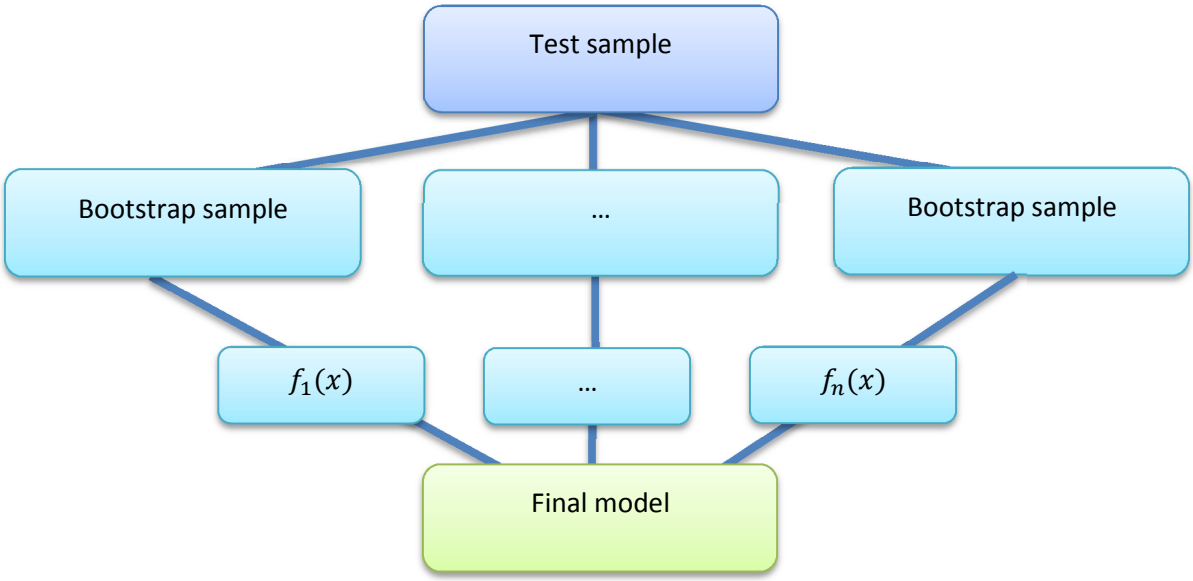
The technique takes $\{x_1, \dots, x_n\}$ input data and makes n random selections with repetition. In other words, if we want to train b sets, n elements from the initial set with $B_1, \dots B_b$ are randomly selected b times, creating different samples.

Once different samples have been obtained from the training data set, we will discuss different techniques for combining them.

1.7.2 Bagging (Bootstrap aggregation)

This method is possibly one of the most simple we will look at. In this case, a model, f_b, is obtained from each training group. Then, a new datum, x_0, is obtained whose label or value will be calculated:

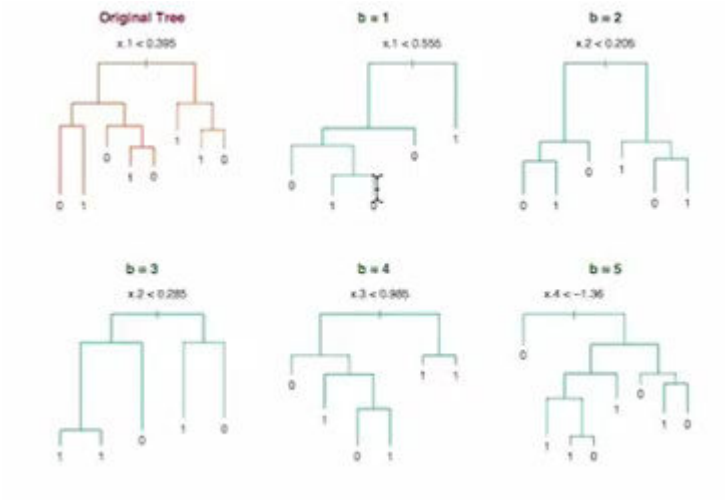
- If we are using a regression model, the result would be the average of the different models, i.e. we would calculate $f(x_0) = \frac{1}{b}\sum_{i=1}^{b} f_i(x_0)$.
- If, on the other hand, it is a classification model, it will be assigned the label obtained in most cases after applying the resulting b model.



1.7.3 Random forest

When creating decision trees, the Bagging technique yields poorer results than expected. The main reason for this is that there is a very high correlation between the different trees created with the Bagging technique. The results would therefore be similar, despite having been unified.

The random forest technique was developed to overcome these problems. Despite being a relatively simple technique that has similar characteristics to Bagging, we can see that it achieves more satisfactory results. The difference between the Bagging and random forest algorithms is that it selects random $m < p$ variables from the sample data obtained after the application of the Bootstrap technique, where selection of the m variables is different. $m \sim \sqrt{p}$ is generally used, but any value can be selected.



1.7.4 Boosting

Finally, the Boosting technique is substantially different from the others and combines several weaker models to achieve better results. In this case, the training sets should not be created using Bootstrap. Here, the new group is chosen for training based on the training data set and assigning a weight (α_i) to each element. After calculating all the models, the weights used to train them will indicate the impact that each model has. The final model will be calculated as follows:

$$f(x_0) = sign(\sum_{i=1}^{b} \alpha_i f_i(x_0))$$

1.7.4.1 AdaBoost algorithm

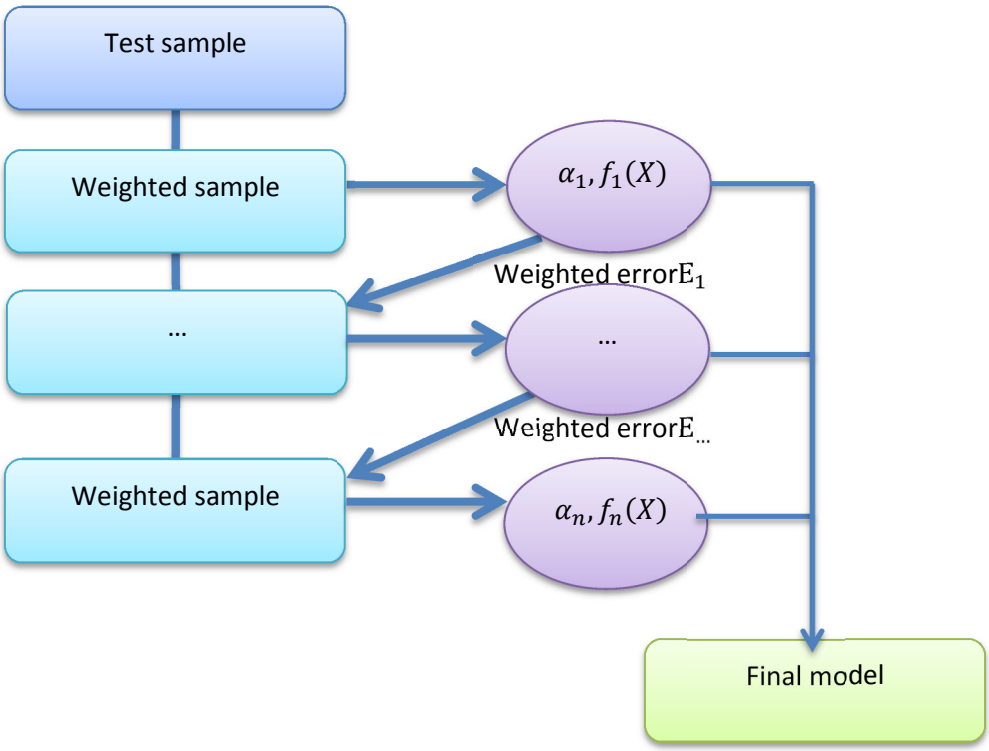
One of the best-known algorithms for Boosting is the AdaBoost algorithm. Let us suppose that we have n elements in our training group and that, for each element, we have a label (1 or -1). In other words, we have $(X_1, y_1), \dots, (X_n, y_n)$ pairs, with X_i elements, and their label y_i. At the outset, the probability of each of the elements being included in the first model will be $p_{0,i} = \frac{1}{n}$. The AdaBoost algorithm then follows these steps for each $t = 1, \dots, T$ iteration:

1. With the Bootstrap technique, each X_i element, with a $p_{t,i}$ probability of being chosen, a B_t group of size n is selected.
2. The f_t classification model is learned with the B_t group.
3. The errors and weights for the resulting model are calculated as follows:
    a. $e_t = \sum_{i=1}^{n} p_{t,i} 1\{y_i \neq f_t(X_i)\}$.
    b. $\alpha_t = \ln(\frac{1-e_t}{e_t})$.
4. The new probabilities are calculated in two ways:
    a. The impact of each element is calculated, attaching the highest importance to those classified erroneously:
        i. $\hat{p}_{t+1,i} = p_{t,i}\, e^{-\alpha_t\, y_i\, f_t(X_i)}$
    b. Considering the impact of all the elements, the probability of each of them being included is calculated in the following step:

i. $p_{t+1,i} = \frac{\hat{p}_{t+1,i}}{\sum_j \hat{p}_{t+1,j}}$

Lastly, the different models are combined to predict the label for each new element. When doing so, the weights calculated above will determine the impact of each model.

$f_{Boost}(x_{new}) = sign\ (\sum \alpha_t\ \ f_t(x_{new}))$.



We can see the variation of the algorithm over 300 iterations here.

2. Unsupervised learning

With unsupervised learning techniques the data will have no assigned label or value. Here, the main goal will be to obtain "hidden" information. Using these techniques we can combine elements with similar characteristics, such as making recommendations based on two

individuals' tastes (as Netflix and Amazon do). Also, the dimension of the data can be reduced with minimal loss of information or their interdependence analysed.

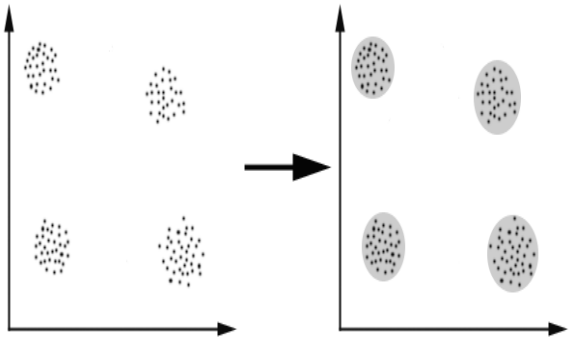
Among the various unsupervised learning methods, the following will be analysed:

1. Clustering
2. PCA
3. Association rule learning
4. Content-based filtering and collaborative filtering
5. Markov models

2.1 Clustering

This is a technique used as part of the unsupervised methods, which seeks to find structures that appear in the data set and attempts to combine them into different groups.

As we have stated, the purpose of this method is to group the available data (incorporating them into different clusters). For this, two main ideas will be considered. First, elements in the same group have to be very similar. Second, the elements in different groups must be as different from each other as possible.



2.1.1 K-means algorithm

One of the most well-known techniques for defining clusters is the k-means algorithm. To apply it, we need to look at how many k groups the data have to be distributed across. Once this operation is complete, we will select k from among the elements in the data set $(\mu_1, \dots, \mu_k)$, which will form the cluster centroids. The algorithm will then repeat the following steps:

1. Each element x_i will be assigned the nearest c_i cluster by calculating $\arg\min_k \|\mu_k - x_i\|_2$.
2. If the value n_k represents the number of elements appearing in the k cluster, the centroids are renewed as follows: $\mu_k = \frac{1}{n_k}\sum_{i=1}^{n_k} x_i$ where the elements x_i appear in the k cluster.

The process is repeated over and over, until they converge or until the pre-established number of iterations has been exceeded. We can see an intuitive example of the algorithm in this video.

2.1.2 Cluster selection

One of the characteristics of the K-means technique is that we must decide at the outset how many clusters the data will be divided into. However, it is advisable to apply the algorithm for different values of k and compare the results. A cost function should be defined for this. As an example, if our database X has n elements, a cost function would be:

$$f(X, \mu_1, \dots, \mu_k) = \frac{1}{n}\sum_{j=1}^{k}\sum_{i=1}^{n_j}\left\|x_{ji} - \mu_j\right\|_2$$

where the element x_{ji} is the element i from the cluster j.

It should be noted that the more clusters are defined, the lower their cost. Zero cost is achieved when defining n clusters for n elements. The value of this variable is selected depending on the objective. For example, if the database analysed contains the clients of a company and if the clients have to be divided between k employees while maintaining the balance between the number of clients assigned to each employee, it would be best to create k clusters.

2.1.3 Common problem

The K-means algorithm can be applied to an optimal local result depending on the random selection of centroids at the start. In such cases, while the result has a low cost, there is another result with an even lower cost. To avoid this, the algorithm should be applied with different initialisations and the result with the lowest cost should be retained. We can see the results produced by the algorithm with different initialisations in this video.

2.1.4 K-medoids

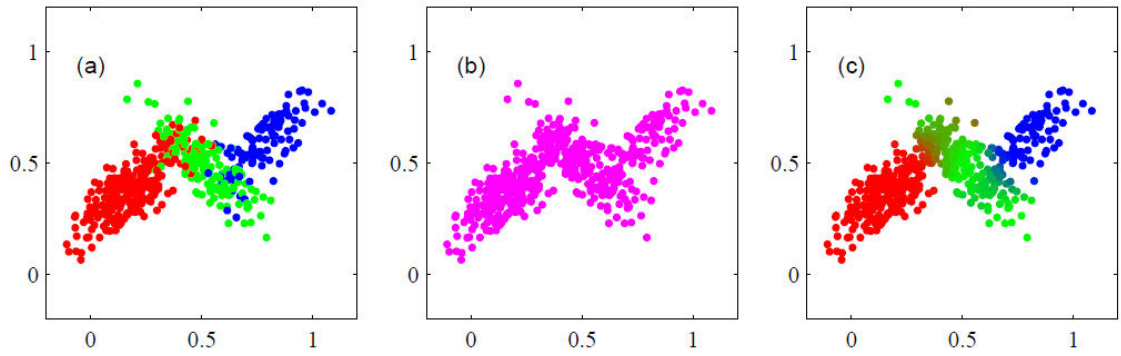
In the case of the k-means function, the distance or similarity between the two elements is measured using the Euclidean norm. To apply this algorithm, the data must be quantitative. Furthermore, since the Euclidean distance is used, elements at a high distance will have a significant impact on our method. To address this, the k-medoids method is used. The sole difference between this algorithm and k-means lies in the first part, where another D differentiation function is used when assigning the elements to different clusters and comparing them with each other. The first step of the algorithm is thus:

1. Each element x_i will be assigned the nearest c_i cluster by calculating $\arg\min_k D(\mu_k, x_i)$.

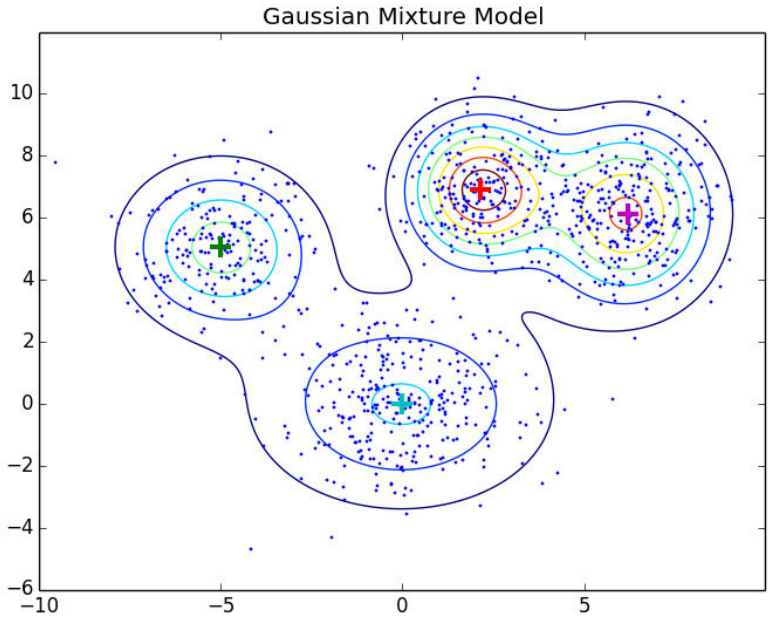
In the case of the k-medoids method, elements are selected from the database as a cluster centre, unlike with the k-means method.

2.1.5 Mixture model

When it comes to clustering, there are two types of techniques: hard clustering and soft clustering. The first involves assigning a cluster to each element and the second involves attaching to each element the probability of belonging to each cluster. The k-means algorithm explained above belongs to the first group of techniques. The image below shows the clusters achieved by the k-means algorithm (3 clusters) for a database (a), the database analysed (b) and the results produced by a soft clustering technique (c).



In these probabilistic models, techniques known as mixture models are used. Here, we will look at Gaussian mixture models (GMM). This technique, like the k-means algorithm, seeks to divide the data analysed into *k* clusters. To do this, it is supposed that there are *k* normal distributions of different parameters among the input data. The model will assign each element to a certain distribution which will yield the same results as the k-means algorithm as the distribution variance approaches zero.



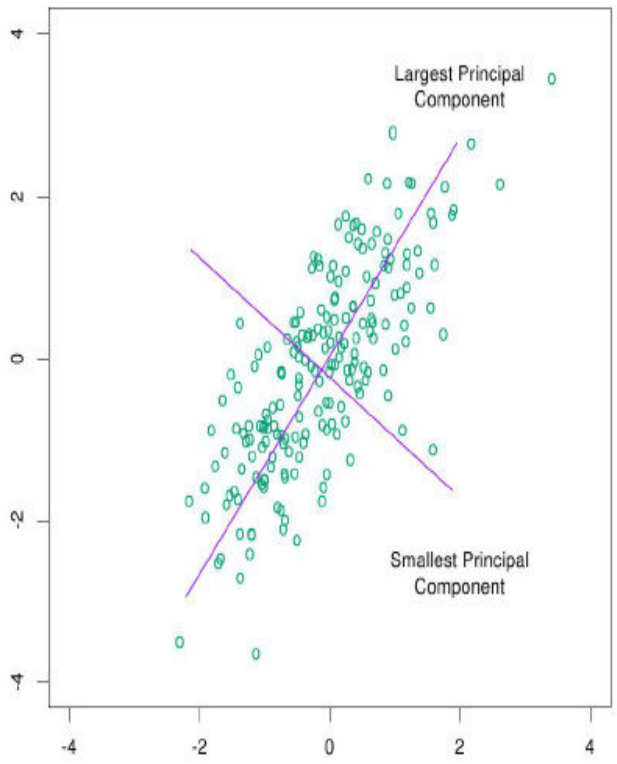
The main advantage to these models is that they provide more specific results than hard clustering techniques. However, as the data are divided beforehand, the different clusters will have a certain fixed appearance – ellipsoid, in the case of Gaussian mixture models. The k-means algorithm, on the other hand, is very fast and can work with a large number of data.

2.2 PCA

When we talk about Big Data, we are referring to a vast amount of data. The techniques we use to conduct different analyses should therefore be able to cope with such quantities. Considering the vast number of variables that we will work with, it is useful to substitute all or almost all of them with a smaller number of variables. This is the goal of the PCA technique;

namely, to reduce the dimension of the information variables, depending on the variables in the database.

Let us suppose that our database contains a total of n elements, each with d variables. We can say that our data lie in a dimension space d. PCA's task is to observe in which directions of this dimension data variance is the highest; in other words, which directions show the most significant changes. Knowing these directions, the PCA technique will provide the best approximation for our data in a lower-dimensional space.



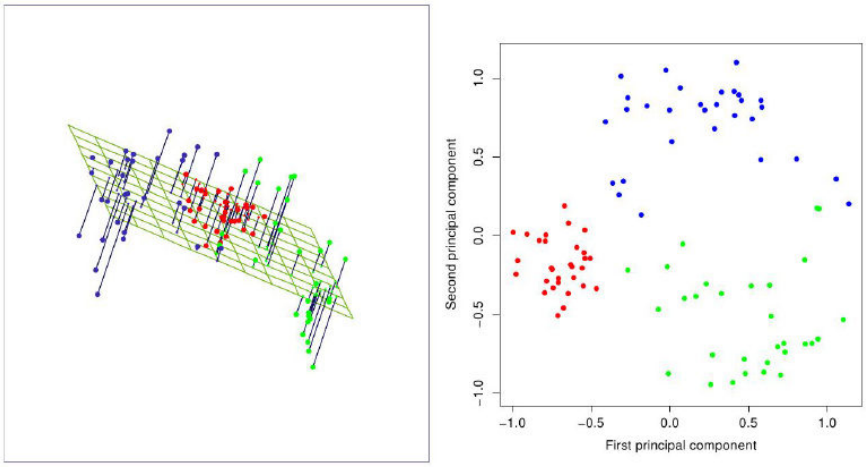
2.2.1. Example d=2

To better understand the concept, the image shows an example representing $d = 2$. In addition to showing our points on the graph, we can also see the main components (principal components) that we will defined later on, explaining which directions produce the most significant information change.

In this specific case, our data are in a two-dimensional space. If we wish to analyse them in a smaller space, the only option would therefore be to transfer the data to a one-dimensional space. To do this, we will project all data to the main component and leave all of them on the one-dimensional line.

2.2.2. Example d=3

Another example we can see represents $d = 3$, where the data from a three-dimensional space are projected in a two-dimensional space.



2.2.3 Theoretical aspect

When we project the data available in a lower dimension, we know that information will be lost. It is logical, therefore, that the directions with the least information loss should be selected when carrying out the projection. As we commented at the start, the variance between different variables will be analysed so as to observe their relationship. Taking all this into consideration, it can be proven that these directions are obtained from the single value decomposition (SVD) of the covariance matrix. Therefore, if $Z = \frac{1}{p}XX^T$ represents the covariance matrix of the data, $Z = US^2U^T$ would be its SVD, where the orthonormal matrix U would be the dimension $p \times p$, and S^2 would be the diagonal matrix, comprising the eigenvalues $\lambda_1 > \lambda_2 > \cdots > \lambda_d > 0$ of Z and constituting $d \leq m$.

The vectors necessary for the transformation will therefore be the first $q \leq p$ columns of the U matrix. These vectors will create the projection matrix U_q. Note that when $q = p$ the dimension will not be reduced since only the coordinates change.

Finally, if our data are $\{x_1, \dots, x_n\}$, after application of the PCA the data will be $\{\tilde{x}_i\}_{i\in\{1,\dots,n\}}$, where $\tilde{x}_i = U_q x_i$.

2.2.4 Dimension selection

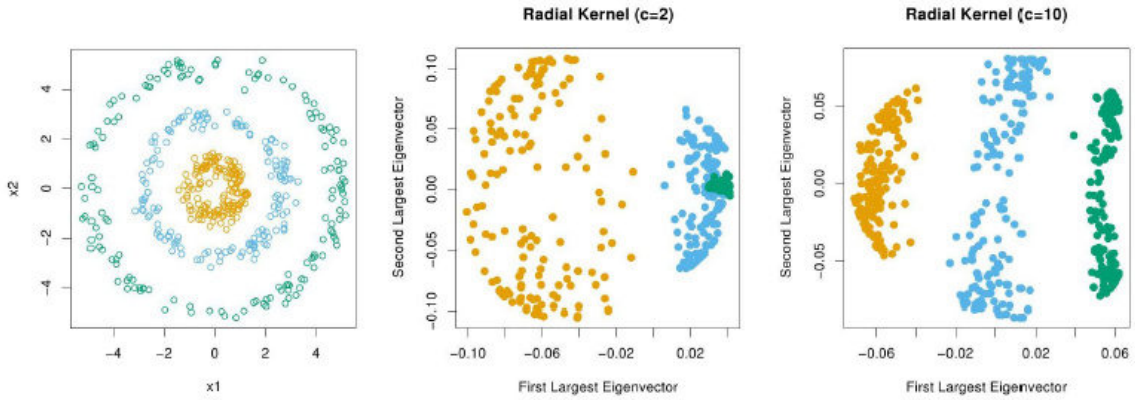
Once the project matrix for the data set has been calculated, we must select the dimensions that we will use. To do this, we analyse the percentage variance of the input data that each of the dimensions represent and select the appropriate number. There is no fixed rule for this operation, so the choice we make will depend on our needs. Let us suppose that the eigenvalues obtained are $\lambda_1, \dots, \lambda_d$, in which case the variance represented by the first main $k < d$ variable will be:

$$\frac{\sum_{i=1}^{i=k} \lambda_i}{\sum_{j=1}^{j=d} \lambda_j} * 100.$$

2.2.5 Kernel PCA

As with other techniques, the data are presumed to be linear, but there is no reason why this should always so. In such cases, before starting to apply the technique, the data will need to be transferred to a higher dimension to ensure that they are linear. Different kernels, such as the Gaussian kernel, can be used for this.

The results can be observed by applying Gaussian kernel PCA to a set of data.



As we can see, after application of the Gaussian kernel with value $c = 10$, the result is a much better reflection of the true nature of the data.

2.2.6 Considerations when applying

- Great care should be taken with missing values. The goal of this method is to summarise information using the correlation or covariance matrix. This is why it is essential for the matrix to be properly defined.
- Considering that the variance between the variables is measured, they need to be at the same scale. Before starting the method, it is therefore a good idea to standardise the variables.
- The more eigenvalues are selected, the greater the percentage general variance. However, there is sometimes little difference between selecting n and $n + 1$ eigenvalues. The number of eigenvalues should therefore be correctly selected, maintaining the balance between the variance and the number of new variables.
- It should be borne in mind that the U_q matrix is calculated based on the training group. This matrix will be used both the validation of the training and the test.

2.3 Association rules

The association analysis seeks to find events appearing together within a set of data. This technique analyses, in particular, commercial databases with binary variables and is known as market basket analysis. In this context, each variable is assigned to a different product. In other words, if we have $\{X_1, \dots, X_d\}$ elements, for every element i the variable X_j would adopt two distinct values. The value $x_{ij} = 1$ would indicate that in the observation i the product j appears, whereas the value $x_{ij} = 0$ would indicate a lack of product j. Meanwhile, the

association rule assesses the impact that the presence of a product or element (or a group of products or elements) has on the presence of another distinct element.

As we said, the technique is generally used in a commercial context. It is particularly used for arranging shop shelves to promote products, designing catalogues and segmenting customers based on their buying patterns.

By way of example, let us suppose that the owner of a food shop has the shopping lists of the last five customers who entered his shop.

Buyer	Product
1	{bread, milk}
2	{bread, nappies, beer, eggs}
3	{milk, nappies, cola, beer}
4	{bread, milk, nappies, beer}
5	{bread, milk, nappies, cola}

The association rules would allow him to identify the buying patterns. For example, there is a high probability that customers who buy nappies would also buy beer. The association analysis would suggest that bread and milk are generally bought at the same time.

2.3.1 Association analysis

Before explaining the algorithm, we will look at a few concepts that we will use. Let us indicate the presence of the elements $X_1 = 1, \dots, X_d = 1$, where $A, B \subset \mathcal{K}$, representing $A \cap B = \emptyset$ and $A \cup B = \mathcal{K}$, with the subset $\mathcal{K} \subset \{1, \dots, d\}$. The following concepts should be borne in mind:

- $\mathcal{P}(\mathcal{K}) = \mathcal{P}(A, B)$; $\mathcal{K}$ the frequency of the elements in the set. It would be useful to observe which combination of elements recurs,
- $\mathcal{P}(A|B) = \frac{\mathcal{P}(\mathcal{K})}{\mathcal{P}(B)}$; knowing that element B exists to determine the *confidence* that element A would appear. Generally, we use $A \Longrightarrow B$ to indicate that there is a *rule*,
- $\mathcal{L}(A, B) = \frac{\mathcal{P}(A,B)}{\mathcal{P}(A)\mathcal{P}(B)} = \frac{\mathcal{P}(B|A)}{\mathcal{P}(B)}$; called the lift of the rule $A \Longrightarrow B$. Knowing that we have the element A, it measures the confidence that element B will appear.

2.3.2 Apriori algorithm

The aim of this algorithm is to find the $\mathcal{K} \subset \{1, \dots, d\}$ subsets with a higher frequency than a predefined threshold t. To help us achieve this aim, the following two properties are used.

- If the frequency of the subset $\mathcal{K}$ is insufficient, the frequency of any set existing within $\mathcal{K}$ will also be insufficient. The mathematically equivalent result, if $P(\mathcal{K}) < t$ and $\mathcal{K}' = \mathcal{K} \cup A$, where $A \subset \{1, \dots, d\}$, will be $P(\mathcal{K}') < t$.
- If the frequency of the subset $\mathcal{K}$ is high enough and if $A \subset \mathcal{K}$, then the frequency of the set A will be relatively high. The equivalent, if $P(\mathcal{K}) > t$ and $A \subset \mathcal{K}$, will thus be $P(A) > t$.

With the help of these properties, a simple version of the apriori algorithm would be:

1. set the threshold $0 < t < 1$.
2. For each element, we will keep those $|\mathcal{K}| = 1$ subsets that have a frequency higher than t.
3. Of the $|\mathcal{K}| = 2$ subsets for two elements obtained in combination with the elements kept from the previous step, we will retain those with a frequency higher than t.
4. This step is repeated, increasing the elements in the $\mathcal{K}$ set, $|\mathcal{K}| = k$, until there are no $k \leq p$ elements with a frequency higher than t in the sets.

From these properties, it is clear that as k increases, the number of subsets eliminated also increases. As a result, there is no need to analyse all of the combinations. Furthermore, it is highly likely that there will be a $k < p$, where all the subsets that exceed it will be rejected.

2.3.3 Improving the algorithm

The algorithm given is no more than a basic implementation of the apriori algorithm and, as is to be expected, it can be improved in a number of different ways.

- Knowing beforehand that the frequency of a set of $|\mathcal{K}| = k - 1$ elements is higher than t, we can say that the frequency of the elements in this set or its combinations will be higher than t. There is, therefore, no need to analyse these subsets.
- Since $\{a, b\} \cup \{c\} = \{c\} \cup \{a, b\}$, it would be a good idea to use indices to avoid repetitions.

2.3.4 Association rules

Once we have found the elements that tend to appear together, we examine whether there is a causal link between them. For this, a second threshold t_2 is defined and the subsets $\mathcal{K}$ obtained in the previous algorithm are analysed. If we suppose that $A \cup B = \mathcal{K}$ is satisfied, then $A \Longrightarrow B$ will also be satisfied if $\mathcal{P}(A|B) > t_2$. Let us recall that in the apriori algorithm $P(A)$ and $P(B)$ are calculated and that it represents $\mathcal{P}(A|B) = \frac{\mathcal{P}(\mathcal{K})}{\mathcal{P}(B)}$. There is therefore no need to calculate the probabilities again. Example

2.3.4 Lift

Of the concepts defined in section 2.3.1, we need to look at what the lift element is for. As we said, lift measures the confidence that element B exists, knowing that we have element A. When analysing the value of this element, a distinction should be made between three cases: lift < 1, lift = 1 and lift > 1. Where the lift has a value of less than 1, it means that the appearance of one of the elements that we have analysed negatively influences the appearance of the second. Where the lift value is 1, it means that there is no connection between the elements appearing. Lastly, if the lift value is higher than 1, it indicates that the elements tend to appear together.

2.4 Content-based filtering and collaborative filtering

These techniques seek to predict the users' preference or rating of an item or product. Nowadays, this type of technique is used by popular web platforms such as Amazon, Netflix and YouTube to recommend new series or videos, once the users' tastes are known.

These systems can be classified into two main branches: content-based filtering and collaborative filtering

2.4.1 Content-based filtering

This type of method is based on product descriptions and users' tastes. Keywords are used to describe different products, which is how users' tastes are determined. If the user gives a positive rating (and if he/she purchases the product), the algorithm recommends other products with similar characteristics.

There are different ways of developing content-based filtering algorithms. However, we will explain an algorithm to resolve the problem in a much more analytical manner below. In any case, this type of problem can be resolved using Bayesian classifiers, cluster analysis, decision trees or even neural networks to obtain simpler results.

2.4.1.1 The algorithm

Let us suppose that we have the ratings of a series of films given by different users in the table below:

Name/film	The Lord of the Rings	Harry Potter	Star Wars	American Pie	Deadpool
Ane	0	?	3	?	4
Leire	2	0	?	5	?
Aitor	?	5	3	2	5
Jon	5	4	4	?	4

Note that the table has missing elements, since not everyone has seen all of the films. Let us also suppose that in the second table we have a series of parameters that have been used to summarise the characteristics of the films:

Film/characteristic	Action	Fantasy	Comedy
The Lord of the Rings	0.8	0.9	0.2
Harry Potter	0.6	0.9	0.3
Star Wars	0.6	0.8	0.1
American Pie	0.01	0	0.9
Deadpool	0.8	0.	0.8

Let us suppose that we have to analyse a database created by N_1 users and N_2 different films. We would have two matrices. On the one hand, we have a matrix of $N_1 \times N_2$ dimensions, with the rating y_{ij} given by each user i to each of the films j. On the other, we have the matrix $N_2 \times d$ with the characteristics x_j of each film, where $d > 0$ is an arbitrary value.

The objective of this algorithm is to obtain a vector that describes the tastes θ_i of each user i. Once these vectors have been determined, the rating of the films given by the users can be obtained as $(\theta_i)^T x_j$.

To find the vector θ_i that serves as a model for each person, the function value used to calculate the error must be minimised, as we have done with other methods:

$Cost(\theta) = \frac{1}{2}\sum_{i=1}^{N_1}\sum_{j:r(i,j)=1}^{N_2}\left((\theta_i)^T x_j - y_{ij}\right)^2$ whereby if the user i gives the film a rating j, it will be $r(i,j) = 1$. We will also add a regularisation parameter in this case so that it does not learn the data used to analyse the model. We thus obtain the following function:

$Cost(\theta) = \frac{1}{2}\sum_{i=1}^{N_1}\sum_{j:r(i,j)=1}^{d}\left((\theta_i)^T x_j - y_{ij}\right)^2 + \frac{\lambda}{2}\sum_{1}^{N_1}\sum_{k=1}^{d}\theta_{ik}^2$.

To achieve the minimum function value, the gradient descent technique can be used as explained above.

2.4.2 Collaborative filtering

Unlike content-based filtering methods, these techniques consider users' activity, preferences and context and attempt to find similarities between them. Once this operation is complete, it is supposed that similar users tend to like similar products. Unlike the previous methods, this one does not analyse the characteristics of the products, meaning that it can recommend complex items such as films without knowing their characteristics.

2.4.2.1 The algorithm

The first table from the previous section is sufficient for this case:

Name/film	The Lord of the Rings	Harry Potter	Star Wars	American Pie	Deadpool

Ane	0	?	3	?	4
Leire	2	0	?	5	?
Aitor	?	5	3	2	5
Jon	5	4	4	?	4

Techniques of this type will be used to mould the characteristics of the different films x_j as well as the tastes θ_i of each individual. The following value function should be minimised:

$Cost(\theta, x) = \frac{1}{2}\sum_{i=1}^{N_1}\sum_{j:r(i,j)=1}^{N_2}\left((\theta_i)^T x_j - y_{ij}\right)^2$ whereby if the user i gives the film a rating j, it will be $r(i,j) = 1$. As we have seen previously, a regularisation parameter will be added so that the model does not learn the data used to analyse it. We thus obtain the following function:

$Cost(\theta, x) = \frac{1}{2}\sum_{i=1}^{N_1}\sum_{j:r(i,j)=1}^{d}\left((\theta_i)^T x_j - y_{ij}\right)^2 + \frac{\lambda}{2}\sum_{1}^{N_1}\sum_{k=1}^{d}\theta_{ik}^2 + \frac{\lambda}{2}\sum_{j=1}^{N_2}\sum_{k=1}^{d}x_{jk}.$

As in the previous case, the minimum can also be determined using the gradient descent technique.

Continuing with the film example, we will distinguish between two different cases when it comes to making recommendations. On the one hand, we can find users with similar tastes by comparing different θ_i elements. Let us suppose that the users i_1 and i_2 have similar tastes ($\left\|\theta_{i_1} - \theta_{i_2}\right\|_2 \sim 0$). Logically, therefore, the films that i_1 like, i_2 will also like, and vice versa. On the other hand, knowing the characteristics of the films and that user i_1 likes film j_1 leads us to suppose that he might also like similar films. He can therefore by recommended any film j if it is similar to j_1, i.e. any film that satisfies $\left\|x_j - x_{j_1}\right\|_2 \sim 0$ can be recommended to user i_1.

When we introduce new users to the platform using this system, we do not initially know their tastes and so we cannot compare them with the other users. For example, continuing with the above examples, let us suppose that we have the following state when incorporating a new user:

Name/film	The Lord of the Rings	Harry Potter	Star Wars	American Pie	Deadpool
Ane	0	?	3	?	4
Leire	2	0	?	5	?
Aitor	?	5	3	2	5
Jon	5	4	4	?	4
Jone	?	?	?	?	?

In this case, it is impossible to infer the tastes of the new user as we have no information. We will mention two of the possible options for overcoming such problems. The first and simplest is to regard the average rating of the films as the new user's rating. This means that films most users like will be recommended at first. As the new customer rates new products, the recommendations will become more personalised. The second is, when adding new users, to ask them to rate a set of products. This will give us the input data necessary to model their tastes.

2.4.3 Improving the techniques: hybrid methods

Although both techniques offer satisfactory results, we have seen how combining them improves them both.

2.4.4 Considerations

- The majority of the cases analysed will have missing data; it is highly unusual for all users to have rated all the products.
- When making recommendations, there needs to be a correlation among the users.
- As well as user ratings, we can also consider other data such as age, nationality etc.

2.5. Markov models

The variables analysed to obtain the models we have seen up to this point are recognised as independent and identically distributed (IID). However, in real life there are cases where it is impossible to satisfy the IID hypothesis, such as the amount of precipitation per hour, the sound characteristics of an audio recording or the daily currency exchange rate. In such cases, we can clearly see that the next value will be dependent on previous values.

We cannot use the techniques we have seen thus far. To work with this type of variable, we use Markov models. We will now turn our attention to Markov chains and hidden Markov models.

2.5. Markov chain

In the theory of probability, the Markov process is one that satisfies the Markov property. This property is also known as memorylessness, meaning that future events in an event chain are independent from past events. Therefore, in order to model a future event, we will only analyse the present state.

When referring to a more mathematical model, if we have a series of different states $(x_1, \dots, x_n)$, the state of element s_t will depend solely on the state of element s_{t-1}, i.e.:

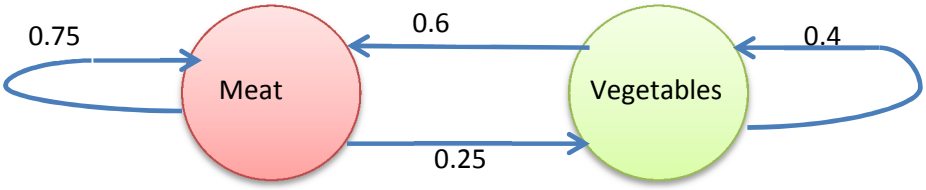
$$P(x_t|x_1, x_2, \dots, x_{t-1}) = P(x_t|x_{t-1})\ \forall i \in \{1, \dots, n\}$$

A well-known example of a Markov chain is the drunkard's walk. Let us suppose that he walks along the line of common numbers (0,1,2,…,n) and the probability of going from the present number to the next is 0.5. Thus, the probability of going from number 5 to 4 would be 0.5, the same for going from 5 to 6. Note that these probabilities are independent with respect to the previous state, i.e. it does not matter whether we arrived at 5 from 6 or from 4.

To use another very simple example, let us suppose that are analysing the diet of an omnivorous animal. We would like to know whether the animal will eat meat or plants the following day. Let us suppose that we have the table of probabilities below:

Today/tomorrow	**Meat**	**Vegetables**
Meat	0.75	0.25
Vegetables	0.6	0.4

We would have the following Markov chain:



The probabilities of going from one state to another are used to create the transition matrix. In this case, we would have the following matrix:

$$M = \begin{pmatrix} 0.75 & 0.25 \\ 0.6 & 0.4 \end{pmatrix}.$$

It indicates the probability that each element M_{ij} in the matrix would go from state i to state j. In a more general case, let us suppose that we have a total of six different states, giving us this matrix:

$$M = \begin{pmatrix} p_{11} & p_{12} & p_{13} & P_{14} & p_{15} & p_{16} \\ p_{21} & p_{22} & p_{23} & p_{24} & p_{25} & p_{26} \\ p_{31} & p_{32} & p_{33} & p_{34} & p_{35} & p_{36} \\ p_{41} & p_{42} & p_{43} & p_{44} & p_{45} & p_{46} \\ p_{51} & p_{52} & p_{53} & p_{54} & p_{55} & p_{56} \\ p_{61} & p_{62} & p_{63} & p_{64} & p_{65} & p_{66} \end{pmatrix}.$$

Note that the sum of the elements on the various matrix lines will always be 1.

It is possible for the matrix to always be defined at source, as in the example above, or each one might have to be defined individually. The maximum likelihood method can be used for this. Let us suppose that we have a series of n elements, and that we want to know the probability of going from state i to state j (which will be M_{ij}). For the element M_{ij}, we can calculate from the elements n available to us how many times they go from state i to j, divided by the number of elements in state i. That is:

$$M_{ij} = \frac{\sum_{k=0}^{k=n-1} \mathbb{1}(x_k = i, x_{k+1} = j)}{\sum_{k=0}^{k=n-1} \mathbb{1}(x_k = i)}.$$

This is a very important matrix since it defines the model set. If the element analysed is in state w_t and we wish to know what state it will be in next, w_{t+1}, we just need to work out $w_{t+1} = w_t M$. Generalising this concept based on an initial state w_0, if we wish to know the state of the element after t moments, we just need to work out $w_t = w_0 M^t$.

Before examining a few applied examples, we will look at one last concept which is key for understanding them, known as stationary distribution. This is the state arrived at after moving an infinite number of times from the initial state, i.e. $w_\infty = \lim_{t\to\infty} w_t$. There will be a stationary distribution only if the following two conditions are satisfied:

1. Any state can be arrived at from any other state.
2. The series of states have no loops of any kind.

Note that, up to this point, we have covered one of the simplest Markov chains, namely the first-order Markov chain. Here, to infer new states, only the previous state has to be considered. However, there are more complex models that consider additional states, such as the m-order Markov chain.

2.5.1.1 Example applications: rankings

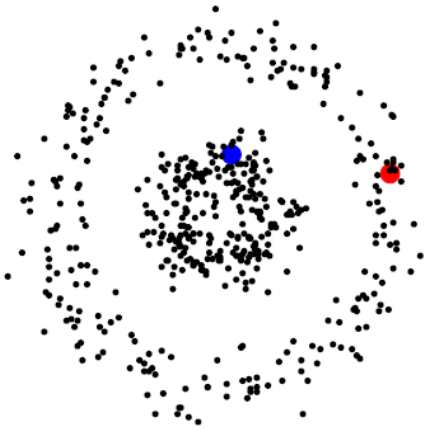
Markov chains can be used for rankings or lists. This time, the data we will analyse will be comparisons between different items. We can take rankings of sports groups or sportspeople as an example. Here, the objective would be to rank the elements from best to worst.

To obtain the list, we will create a transition matrix, where each state will be a group or a sportsperson. The stationary distribution will then give us the result we are looking for.

When creating the matrix, we will ensure that the losing groups go to those that are winning. In other words, if group A beats group B, the probability of B->A will be higher than the probability of A->B. This link may be stronger depending on the baseline marker. Lastly, the matrix lines are normalised so they add up to 1. Once this is done, the stationary distribution of the Markov chain is found, and the vector value will give us the list of the different groups. The most likely states will be the best groups, whereas those with the lowest probability will be at the bottom of the list.

2.5.1.2 Example applications: classification

Let us suppose that we have lots of data, only a few of which are labelled, and that, based on their structure, we want to know the labels of the unlabelled elements. If the data are structured, we could infer the label of the elements through the Markov chains. For example, the image on the right shows us a large number of data (black points), with two labelled points (blue and red elements). In this case, it appears that the elements have a fixed structure (blue points in the centre and red in the ring), meaning that we can use Markov chains.



We need to make a transition matrix here too, in order to increase the probability of moving between the nearest points. On arriving at a labelled element, it will remain there, i.e. if x_i is a labelled element, it will be $M_{ii} = 1$. These points are called absorbing states.

Finally, to assign a label to unlabelled elements, the first absorbing state will be analysed, which is obtained from these points by applying the transition matrix. The label of the point in this state will be the label of the initial point.

2.5.2 Hidden Markov models

The observations in the examples that we have seen up to this point have been measurable. However, this is not necessarily always the case. States may sometimes be unmeasurable, or we might not have the chance to measure them. Hidden Markov models come in useful in such cases. Based on a measurable observation sequence, $\{x_1, \dots, x_n\}$ states are obtained and will be analysed.

As a theoretical example, let us suppose that Ane and Jon are two friends who live far away from each other. Let us also suppose that they talk every day by telephone. Jon, depending on the time, does three main activities in the afternoons: sport, shopping and working on his computer. Let us suppose that Jon tells Ane what he is doing every day. With these data, Ane wants to know what the weather was like on each of those days because she has noticed a distinct correlation between both events. This is an ideal situation in which to apply hidden Markov models. Although Ane is aware of her friend's activities, she has no information on what the weather was like every day. Let us suppose that Ane knows that the weather in Jon's city follows this probability distribution:

Today/tomorrow	**Sunny**	**Raining**
Sunny	0.6	0.4
Raining	0.3	0.7

Sunny	**0.4**
Raining	0.6

Ane also knows that Jon has the following habits:

Today/tomorrow	**Sport**	**Purchases**	**Computer**
Sunny	0.6	0.3	0.1
Raining	0.1	0.2	0.7

With all of these data, Ane will have the information she needs to obtain the probability of the weather on a certain day, knowing Jon's activity.

In the following two articles, we can see some practical examples. In the first, the models display results from their application in the field of medicine. In the second, attempts are made to infer the meaning of sign language gestures.

With this method, we can see three main components:

1. the transition matrix M_t, which contains the probabilities of changing from one state to another;
2. the emission matrix M_e, which indicates the probability that each of the elements has been created for each state;
3. π the distribution of probabilities for the initial states.

We have the following matrices for the examples above:

$$M_t = \begin{pmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{pmatrix}; M_e = \begin{pmatrix} 0.6 & 0.3 & 0.1 \\ 0.1 & 0.2 & 0.7 \end{pmatrix}; \pi = (0.4 \quad 0.6).$$

It is possible that all of these data might have been established from before, as we have seen in the example. However, there will be cases in which they have to be inferred. In the case at hand in particular, it would be a discrete hidden Markov model because the states are discrete. This technique is mostly used in three situations:

1. When based on the $\{\pi, M_t, M_e\}$ elements and the $\{x_1, \dots, x_n\}$ observation sequence, we want to know the probability of the $\{s_1, \dots, s_n\}$ state. We use the forward-backward algorithm for this.
2. When we want to obtain the most likely sequence of $\{s_1, \dots, s_n\}$ states based on a $\{x_1, \dots, x_n\}$ observation sequence and $\{\pi, M_t, M_e\}$ elements. We use the Viterbi algorithm for this.
3. When we want to obtain the $\{\pi, M_t, M_e\}$ in the model based on a $\{x_1, \dots, x_n\}$ observation sequence. For this we use the maximum likelihood method.

Lastly, although this model is discrete, it can be applied to a state in a continous space where the state is weather, for example.

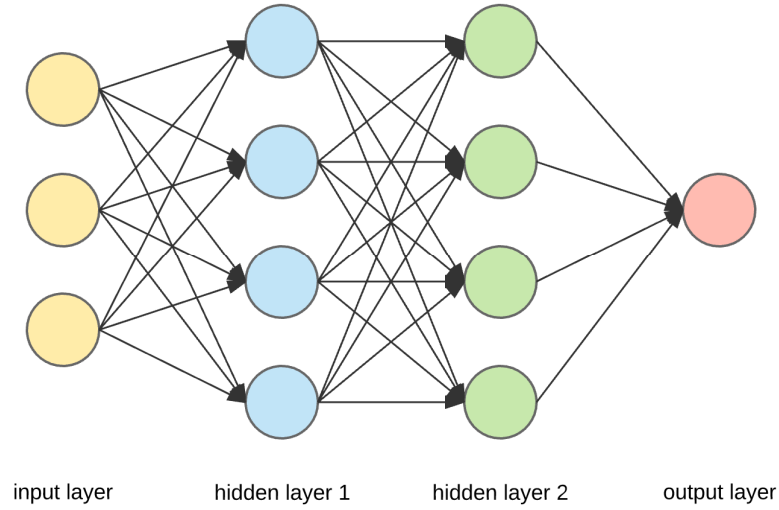
3. Neural networks

Neural networks are techniques that seek to imitate how neural networks function in animals. As in nature, the networks are composed of different interconnected nodes called neurons. While the techniques are mainly used for supervised learning, there are networks capable of handling unsupervised learning tasks. It is also used in robotics and in designing self-driving cars.

The structure of neural networks is generally distributed in three ways:

1. Input layer: input data are introduced into the neurons (categorical and sequential values, photos, text, etc.)
2. Hidden layer: where precise combinations are made and the different functions are applied to the input data.
3. Output layer: this is made up of neurons that contain output data (categories, successive values, photos, etc.) Depending on the objective, we will have a linear combination of hidden layers or a function applied to that combination.

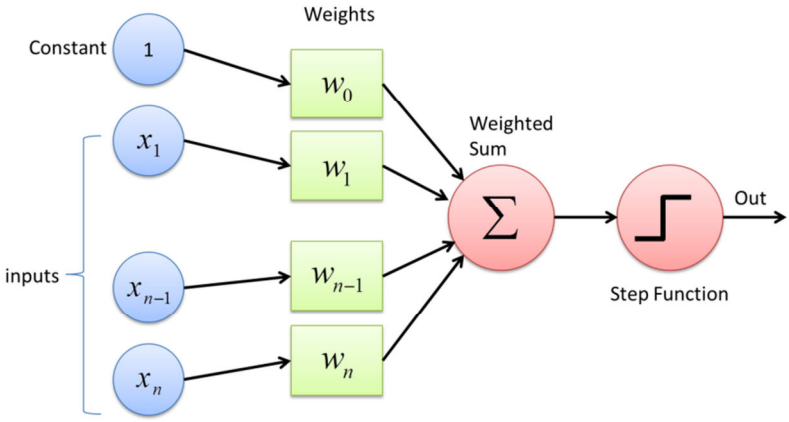
During the transition from one layer to the next, different linear combinations of neurons in the layer below are made, using scale values known as weights. A step function or activation function is then applied to the value obtained in the combination at each node, to determine whether or not the neuron activates and to find the impact on each neuron. In the image below, we can see the structure of a basic neural network.



The neural network in the image above will undergo the following process:

1. Variables from our first element $X_1 = (x_{11}, x_{12}, x_{13})$ will be included in the input layer.
2. For each node $j \in \{1, \dots, 4\}$ in the first hidden layer, the weights will be w_{j1}, w_{j2} and w_{j3}.
3. At each node $y_{1j} = f_j(w_{j1}x_{11} + w_{j2}x_{12} + w_{j3}x_{13})$, we will apply the activation function f_j to a linear combination of values from the previous laye.
4. In the second hidden layer, the same process from the previous layer will be repeated, obtaining the values y_{2k}, with the new activation functions f_k with w_{k1}, w_{k2} and weight w_{k3} for each k neuron.
5. Finally, as in the previous cases, we will assign to the initial layer the value obtained by applying the most recent activation function to a linear combination of values in the last hidden layer, using the weight w_1, w_2, w_3 and the activation functions f and obtaining the output value y.

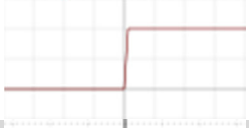
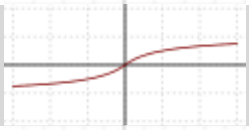
The Perceptron algorithm, simple linear regression or simple logistic regression, which we analysed in section 1, are among the most straightforward neural networks. They would all have the structure shown in the image below, applying a different application function.



In the case of Perceptron, we use the sign function. For simple linear regression, the identity function or ReLU function would suffice, as will be discussed later on (when our labelled data do not have negative values). Lastly, the sigmoid function is sufficient for simple logistic regression.

3.1 Activation functions

As we have seen, selecting the activation function has a major impact on the final model. It allows us to go from a model that is used for classification to another that is suitable for regression. Below are a series of principal functions:

Function name:	*Function*	Graph
Identity	$f(x) = x$	
Binary step	$f(x) = \begin{cases} 0 \; non \; x < 0 \\ 1 \; non \; x \geq 0 \end{cases}$	
Softsign	$f(x) = \frac{1}{1 + \lvert x \rvert}$	
Sigmoid (Logistic)	$f(x) = \frac{1}{1 + e^{-x}}$	
ReLU (Rectified linear unit)	$f(x) = \begin{cases} 0 \; non \; x < 0 \\ x \; non \; x \geq 0 \end{cases}$	

All these examples affect a single node, but there are functions that take into account the information inserted into all nodes in the same layer, namely:

Function name:	*Function*
Softmax	$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{k=1}^{n} e^{x_k}}$

This function is used, inter alia, to classify other multi-classes.

To finish with activation functions, it should be noted that those explained here are no more than a small percentage of all the existing functions. We could also use our own functions that have been invented or that are variations on those that we have already seen.

3.2 Input and output data

As well as handling numbers, neural networks are capable of handling categorical variables with photos or text. However, considering that numbers are essential when making calculations, all variables of this type have to be transformed.

If the variables are categorical, we can assign the values 1 and 0 when there are two categories. Furthermore, if there are n categories, we only need to create n new variables with values 1 or 0. Every variable defines a category.

When the variables are photographs, we can distinguish two cases. On the one hand, when colour is not needed, we can change it to greyscale. Thus, the photo would be a $n * m$ matrix made up of pixels. In this case, the matrix will be converted into a vector of nm elements where each element is a variable. On the other hand, when colour is important, the same operation can be used but we would instead obtain a matrix of dimension $n * m * 3$ and we would distribute the three vectors one after the other.

When we are handling text, we have several options. On the one hand, we can create a variable for each word that appears. Here, we would observe how many times each of the words appear in the sentences we analyse. On the other hand, we can calculate the roots of the words that appear (*dimensional* -> *dimension*, for example). When carrying out this operation, we just need to obtain one variable for each root and assign a value of 1 or 0.

3.5 Testing the networks

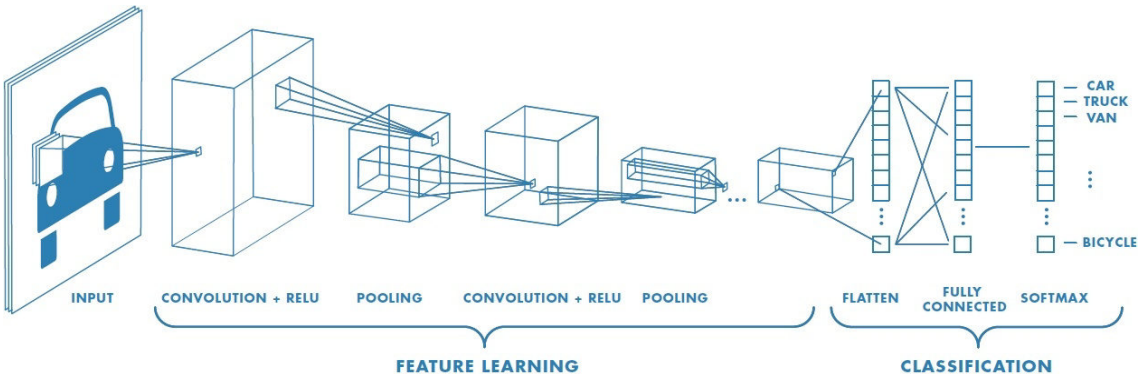
When forming a neural network, we should firstly decide on its structure. In other words, we must decide how many layers and how many nodes we will use. There is no fixed answer for this. Depending on the purpose, there might be certain pre-established settings. Most of the time, however, tests will have to be conducted to achieve an appropriate structure. After selection, we have to define the weights that we will use. The most common method for this is very similar to the gradient descent technique used in several methods in the first section and is known as backpropagation. To start with, the weights are randomly selected and the network is applied to the data. The result (which will probably be unsatisfactory at first) is then compared with the real result and the weights are adjusted depending on the errors. To visualise the method in greater detail, follow this link.

3.6 Other types of neural networks

We have, up to this point, analysed the most basic networks, which are known as multi-layer feed-forward fully connected neural networks. There are, however, many more network types. We will see several examples of another four network families below.

3.6.1 Convolutional neural network: CNN

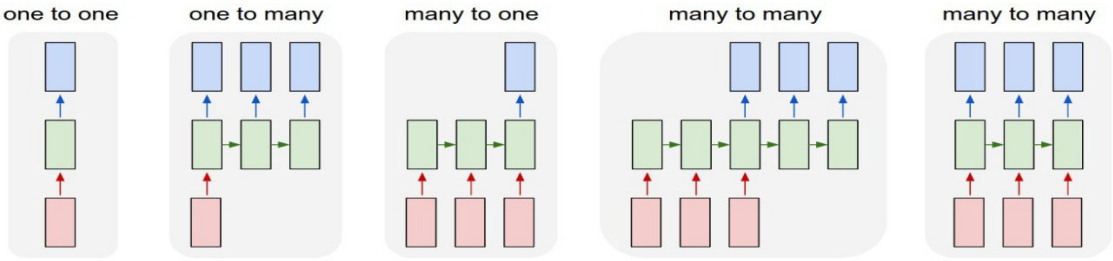
When handling photographs – generally for classification – it is useful to have a model that is capable of recognising their characteristics. To put it another way, if we want a model that has facial recognition, it would be helpful if our model could detect ears, eyes, lips, etc. This is precisely the function of networks of this type.



As well as having layers as mentioned in the previous section, a further two types are used: convolutional layers and pooling layers. The first passes the photos through different filters in order to isolate their characteristics, whereas the second reduces the dimension of the characteristics analysed for greater precision. These two layers are generally used in these kinds of networks to determine the impact of the characters. The networks explained at the start of this paper are then used for classification.

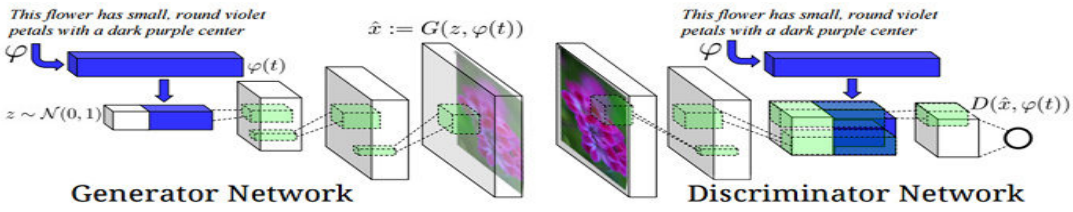
3.6.2 Recurrent neural networks:

Sometimes, it is useful for the model to have "memory", i.e. to take into account the results previously obtained for future calculations. We use this type of network in these cases. They are generally used in networks for creating or analysing text or sound, since in this case the context is highly important.



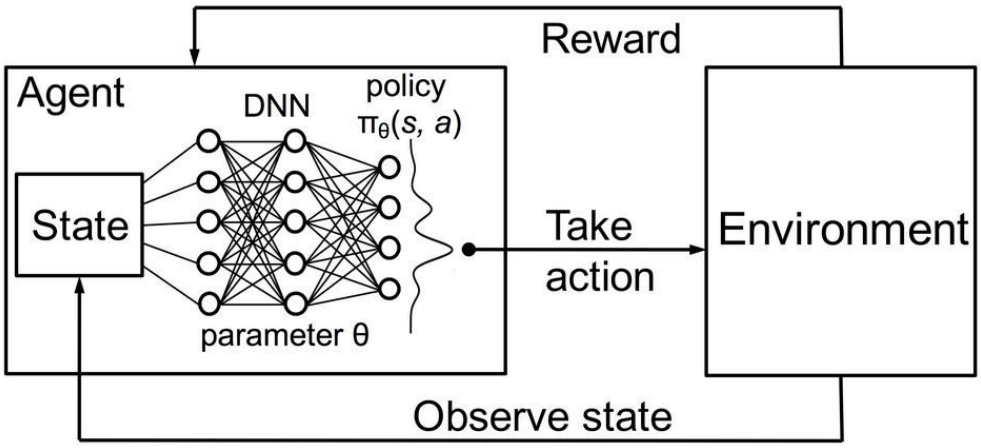
3.6.3 Generative adversarial networks

As its name suggests, this type of network is generated in an adversarial manner. They generally comprise two different networks, namely generative and discriminative. In many cases, the aim is to create data. The first type of network learns to create data, whereas the second type seeks to determine whether the data are real or fictitious. As the data are created in an ever more reliable manner, the discriminative network will yield ever poorer results, meaning that the weights will make more significant corrections. Otherwise, the opposite will happen and the generative network will obtain larger corrections.



3.6.4 Deep reinforcement learning

Networks of this type learn from their environment. They are generally used in robotics and the models improve with experience. They start off with a random behaviour and, as the number of tests increase, the better the results become.



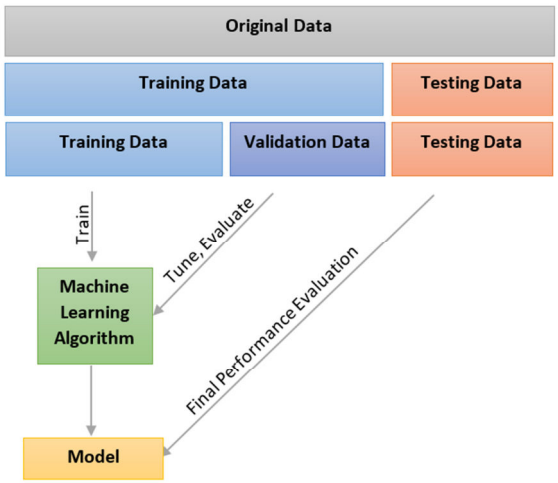
In the following videos, we can see examples of convolutional neural network, recurrent neural networks, generative adversarial networks and deep reinforcement networks.

3.7 Advantages and disadvantages

The main advantage is that the networks are capable of doing practically anything. They can also be reused. Let us suppose that we have a network that detects cats, but we want to go on to detect dogs. We will not need to create a new network from scratch. Based on the network we used to detect cats, we can create a model capable of detecting dogs. On the other hand, they are difficult to interpret, as hundreds if not thousands of neurons are used in each layer, and it is difficult to know what each of them does.

4. Training, validation and test sets

In the case of supervised learning and neural networks, as well as developing models, it is also necessary to measure their quality. If in order to evaluate the models we use the same data as we use to develop them, we might obtain better results than in a real case (with unknown data for the model). To avoid this, the data are distributed across three available groups: the training group, the validation group and the test group.



First, the group of training elements will contain data for the model to learn. These data will be used to learn the parameters explained in the previous pages – generally weights. Together with the training group, we will also select the validation group. The purpose of this second group will be select the models' hyperparameters and control the overfitting of the training. This set is used, for example, to select the number of layers in neural networks and the activation functions. Different types of neural networks will be defined using the training group and calculating the weights. Once this is operation is complete, the models will be applied to the validation group of data and the results will be analysed, selecting the network with the best hyperparameters. Lastly, the test group is used to measure the quality of the final model. The model is applied to these data, comparing the predefined results.

As an example of how these three groups are used, we outline the process of a neural network below:

1. Starting with data from the training group, we create different networks that will serve as candidates for the final model.
2. The elements in the validation set are applied to the models.
3. The results are analysed and the network that present the best hyperparameters are chosen. This model is likely to be suitable for the data in the validation set or group.
4. The results given by the model from the validation process is analysed in the test group,
   4.1. using parameters such as accuracy, sensitivity, specificity, F-measure, etc.
5. If the results are not satisfactory,
   5.1. the model will not be suitable.
   5.2. New models must be developed.
6. If the results are satisfactory,
   6.1. the final model has been created and it will be ready for testing with real cases.

4.1 Defining the sets

As we often do in machine learning, we will select the sets based on the problem we are analysing and the data we have. If the model that we create has few hyperparameters, a small validation set would be sufficient. On the other hand, if we have a large number of hyperparameters, it would be a good idea to expand the validation set. Furthermore, if we have a small quantity of data, it is likely that we will need all of them to develop the model, despite the fact that it is not the optimal case.

With this in mind, we will divide the initial set of ideal cases into two: the set for developing the model and the test set. It is advisable for the first set to be higher than the test set. Once the initial division is complete, we obtain the states that we will use for training and validation from the model development set. The cross-validation technique is generally used for this second step. There are different types of cross-validation, which we can categorise into two main groups: exhaustive cross-validation techniques and non-exhaustive cross-validation techniques. We will mention and explain some of them below:

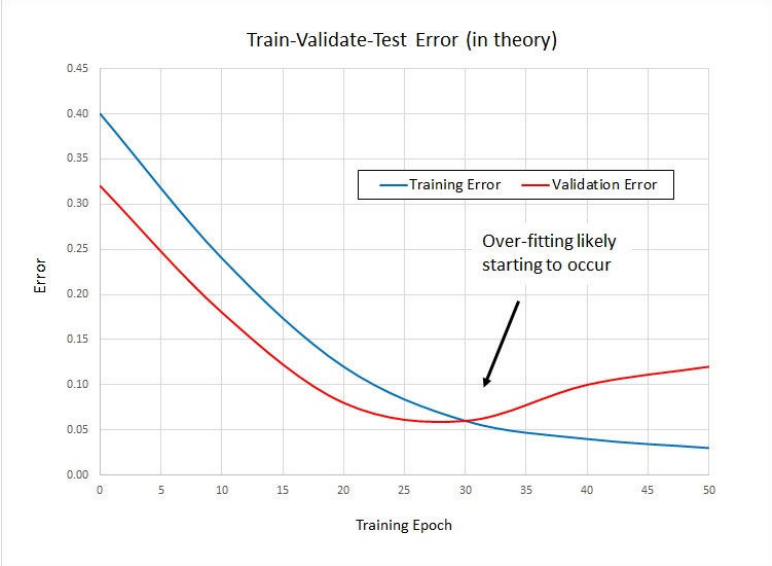
1. Exhaustive cross-validation techniques:
   1.1. Leave-one-out cross-validation
   1.2. Leave-p-out cross-validation
2. Non-exhaustive cross-validation techniques:
   2.1. K-fold cross-validation
   2.2. Holdout method
   2.3. Monte Carlo cross-validation

The exhaustive cross-validation techniques consider all existing combinations in the set. In the case of leave-one-out, for example, if the set used for the model has n training elements, it will take $n-1$, and leave the missing element for the validation process. This will be done n times, using a different validation element each time. The leave-p-out method, however, is a more general version of the case above. Here, $n-p$ elements will be used in each iteration for training, leaving p cases for the validation process. It should be noted that in the first case, while for element n there are only n possible combinations, in the case of the leave-p-out method the number of combinations (C_p^n combinations for element n) increases substantially. Let us suppose, for example, that we have $n = 50$ and $p = 2$. In the case of leave-one-out 50 combinations would be tried, whereas in the case of leave-p-out there would be $C_2^{50} = 1225$.

In the case of non-exhaustive cross-validation, on the other hand, not all of the possible options need to be analysed. With k-fold cross-validation, the set selected for the model is randomly divided into k subsets. A subset will then be kept for validation and the remaining $k-1$ subsets will be used for training. To finish, as with leave-one-out, the process will be repeated k times, using a different validation subset for each iteration. As for the holdout method, the set used for the model is randomly divided into two subsets, following a predetermined proportion (bear in mind that it is generally a good idea for the training subset to be higher than the validation subset). In both these cases, the elements that we find in both subsets must be divided in a similar fashion, i.e. all of the subsets have to have examples of all types of elements being analysed. Lastly, we will explain Monte Carlo cross-validation. In this case, we set a percentage in advance, for example 80%. Thus, 80% of the elements will be randomly chosen for training, whereas 20% will be used for validation. The process will then be repeated several times.

4.2 Detecting overfit

As mentioned previously, the validation set is used to detect overfit. As the algorithms are applied, the errors obtained in each iteration are analysed in both the training and validation sets. We will then observe the development of the errors on a graph:



As we can see from the graph, the training error will go down or remain constant. This will always happen; as the same elements are used to train the algorithm, we will obtain better and better results as there are more iterations. However, from the 25th iteration, the validation data error stops decreasing and starts to increase. At this point, the algorithm will stop learning new information and will start to learn the training results. We will therefore obtain the best model in this case with that number of iterations.

Spatial autocorrelation and heat maps

The data used for the analysis of hotels and guesthouses in the Basque Country come from two main sources: web platforms and the tourism directories of the Basque Statistics Institute, Eustat.

Data have been used from Eustat's surveys on tourist establishments in particular. They hold additional information on hotels and guesthouses: category, occupancy, number of rooms, stratum, geographical coordinates, etc.

As for the websites, the prices of hotels and guesthouses in the Basque Country have been obtained from a customised web scraping technique developed by Eustat. Furthermore, as the prices may be subject to change over time, the prices over the previous 120 days have been taken. This means we could have 120 different prices, by day and by hotel. Once we have approximately 120 pieces of data, by day and by establishment, the data are summarised so that there is a single price in each case. With this objective, we have calculated the median for all of the elements because this statistic cancels out the impact of the opposite values and it can therefore be used for preprocessing.

As we might suppose, not all the hotels in the Eustat database appear on web platforms. However, coverage of approximately 80% of the hotels and guesthouses in the Basque Country has been achieved.

Once the information from both sources has been merged, we follow these steps:

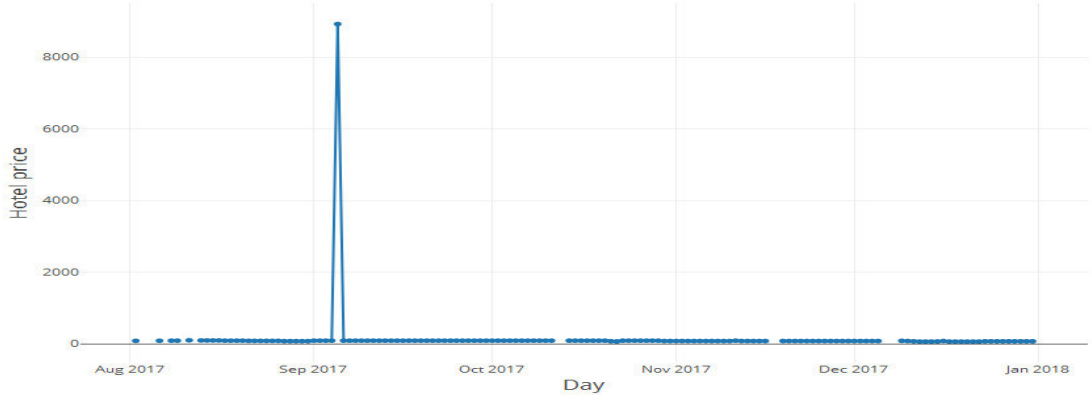
1. outlier analysis,
2. imputation of missing values,
3. spatial analysis of the hotels,
4. heat map.

All of the analyses have been performed using the programming language R, which, as well as being used for statistics, has gained considerable force in the world of machine learning in recent years. For R programming, the RStudio integrated development environment (IDE) was used for its code editor, debugging tools and visualisation.

5. Outlier analysis

The data used for the analysis are obtained automatically, meaning that it is possible for them to be incorrect because of a problem with the scraping process or the website itself. Although poorly obtained values might be incorrect, they will not have a major impact on future analyses as long as they are similar to those obtained at other nearby times. On the other hand, if the errors provide extreme or extremely anomalous data, they may have an adverse impact in the future.

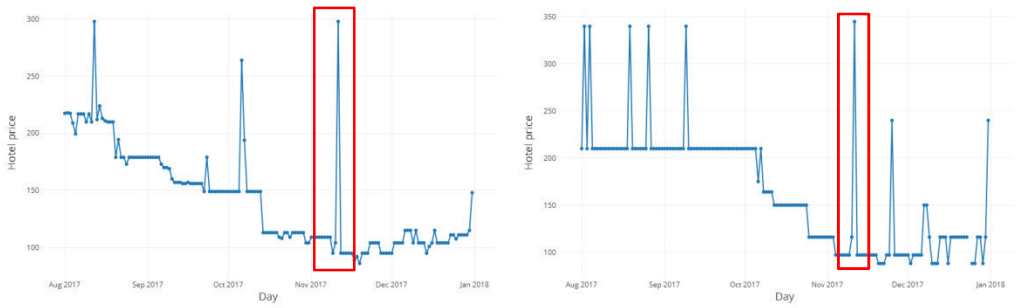
Once the scraping has been completed, the median of all the data for each day is calculated, at which point many of the anomalous values disappear. This does not, however, eliminate all of the extreme values and a more exhaustive analysis needs to be carried out.



There are three main types of anomalous value.

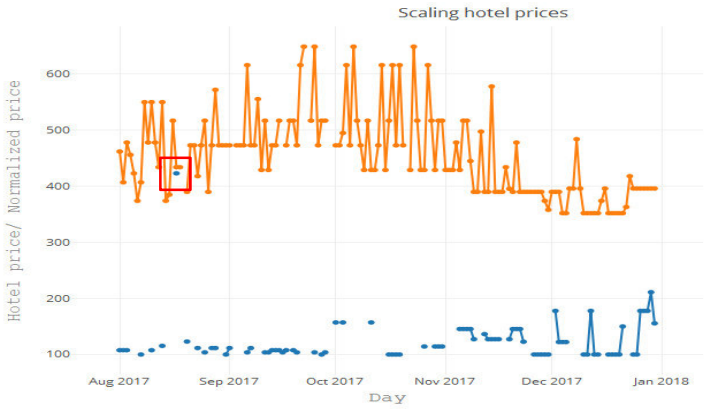
1. Outlier: these are a small number of data items that are very distant from each other and from the majority as a whole.
2. Anomaly: these are a small number of data that are close to others in their respective category but far from the majority as a whole.
3. Innovative value: these are points that make up a new, unknown category.

In this case, we might have the first two types of value. On the one hand, outliers are data equal to the points in the image above. Here we are dealing with an “impossible” value. On the other hand, the anomalies are special days such as public holidays, weekends, etc. As we can see in the image, on 11 November several hotels have anomalous values. However, this phenomenon occurs in various cases, and it they do not just affect one hotel.

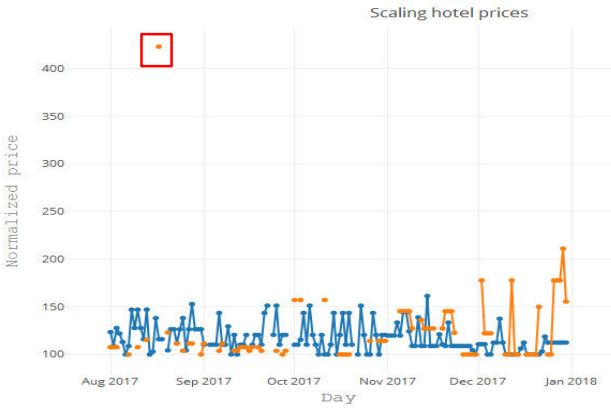


5.1 Comparable hotels

To apply the same criteria to different hotels, they need to be placed on the same scale. In other words, the price of a room at a five-star hotel will never be comparable to the price of a one-star guesthouse. Furthermore, guesthouses in San Sebastián generally do not have the same price ranges as hotels in Vitoria. We can see the prices for a five-star hotel room and a guesthouse room in the graph below.



Although the value highlighted by the red square might seem to be an outlier, by analysing the hotels simultaneously we might not detect this point from among the five-star hotel values. To avoid this problem, the hotels have been transferred to the same scale. To do this, instead of analysing the price of the hotels, we have analysed the price variation by month. Considering that the minimum values for each month were very stable, we assigned the minimum value of 100 to each month. Once this was done, we assigned the rest of the values a value proportional to the increase with respect to the minimum value. To put it another way, if the monthly minimum is €200 and if we have a value of €600 during the same month, we will attribute the value of 100 to the minimum point, while we will attribute the value 300 to the €600 price. Once this is done, we will obtain the following graph by analysing the same hotels as before:



As we can see, the point that we at first thought was an outlier clearly stands out.

5.2 Grubbs method

The Grubbs test finds a single outlier. It analyses the maximum and minimum values observed and tells us that one of them is an outlier. For this, it targets the element furthest away from the elements analysed and their average, considering the standard deviation:

$$G = max_{i=1,..,N} \frac{|Y_i - \bar{Y}|}{s}.$$

The test is available in the Outlier package of the R software, through the function *grubbs.test()*. This function tells us whether the maximum and minimum elements analysed are extreme values. It gives us a statistical $0 \leq p - value \leq 1$. We will use this value, together with a limit defined by us, to determine whether or not a certain element is an outlier.

5.3 Analysis of hotels

In the outlier analysis that we have performed, the hotels were taken separately at first. It should be noted that we will obtain more points than the outliers, since we will also have anomalies. This process has been completed by month. To obtain the data for a new month, the data from a previous month will not be required.

5.4 Analysis of days

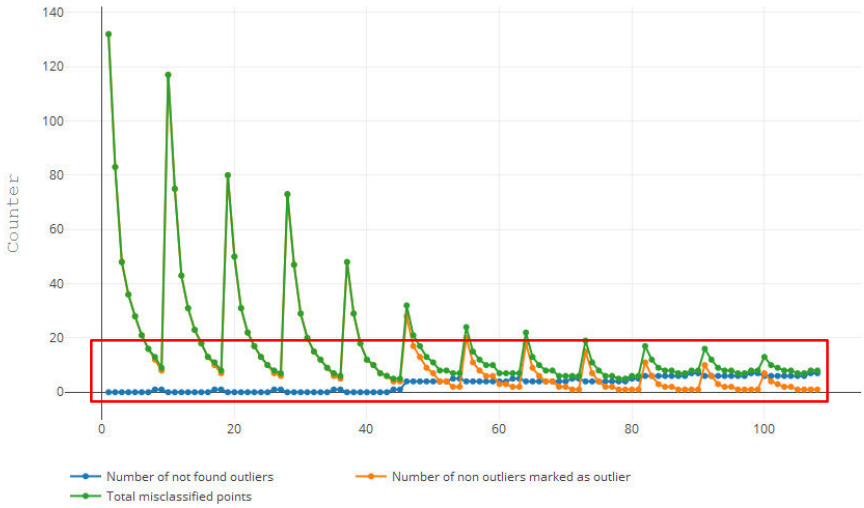
Days were analysed as well as hotels. Once this was done, the points will not be regarded as outliers if the hotel prices generally increased on a specific day because of a special event (public holidays or events, etc.) This analysis was done by province, since each one has its own public holidays.

5.5 Unification of the results

Finally, in the two cases of outliers analysed among our data, only the data that appear as such is considered. In other words, a value will be an outlier only if it is an outlier among the values for the hotels and also for the corresponding day.

5.6. P-value

We said that the Grubbs test returned a $p - value$ and that a limit should be set where the point analysed, $p < limit$, constitutes an outlier. We have manually indicated the outliers and compared the results obtained from the combination of different $p - value$ (one for the hotel analysis and another for the days) with the results indicated. In the following graph, we compare the values obtained, where the different points $p - value$ indicate a different pair. The blue lines represent the missed outliers, while the orange lines are normal points shown as outliers. The green lines are the sum of both types of error:



Once this is done, the following criteria were used to define the limit of $p - value$:

1. Analysis of $p - value$ that minimise the total number of errors (green line).
2. Selection of the $p - value$ that find the largest number of outliers (minimising the blue line).

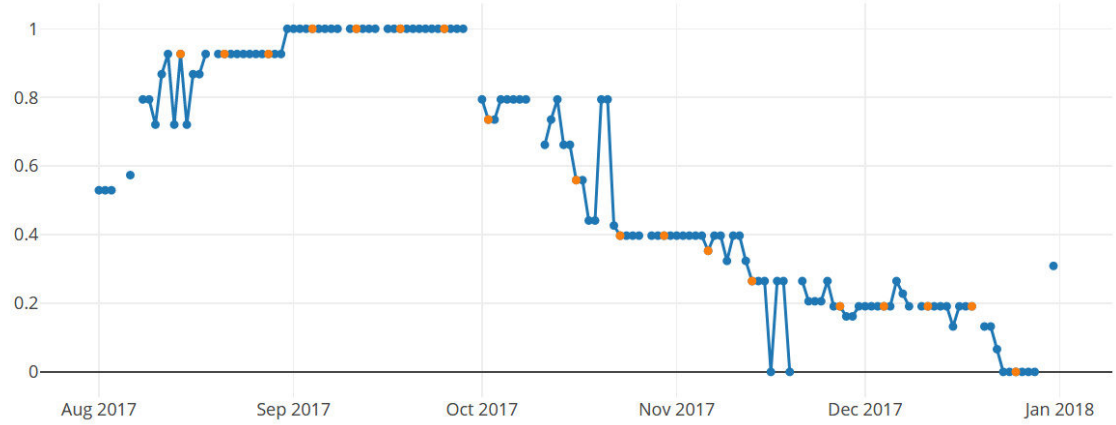
6. Imputation of missing values

Once we have detected the outliers, we must decide what to do with them. In our case, we decided to eliminate these values and impute the data from the same day, together with the missing values. We used the *imputeTS* package in R software for this. The package has several methods for imputing time series, namely *na.seasplit, na.seadec, na.interpolation, na.kalman, etc.* In our particular case, we can say that the series has periodicity, i.e. that we generally observe seven-day patterns, with price increases on Fridays and Saturdays.

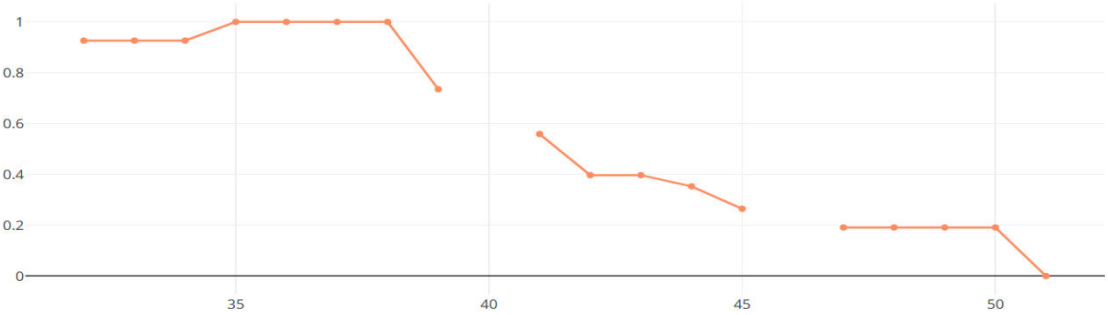
Before starting with the selection criteria and the results of the imputation method, we will analyse the main functions recommended for working with series with periodicity, such as *na.seasplit, na.seadec, na.kalma.*

6.1 na.seasplit

The first step for this function is to indicate the periodicity of the series. We will attribute a series format to the series of prices for one of our hotels x in the R application `x<-ts(x, frequency=7)`. With this function, we can see a periodicity of 7 days. Once this is done, the method will obtain a total of 7 series other than the original series (one for each day of the week). Let us suppose, for example, that we have a series of normalised prices for a hotel, where we have highlighted Mondays in orange:



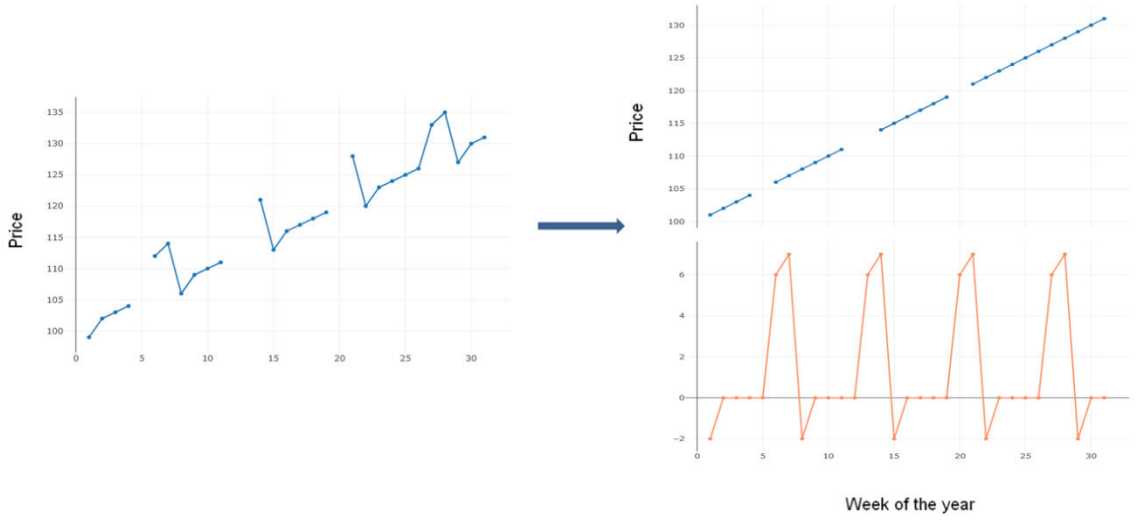
In this case, the method would obtain another 7 series similar to the one we can see in the image below:



Once this operation is complete, we will apply the desired imputation method from among the options available in the *imputeTS* package to each series: *ARIMA, interpolation, basic structural models, etc.*

6.2 na.seadec

As in the previous case, we must indicate that our series has a periodicity of 7 days. We will then eliminate the periodicity of the series for imputation of the remaining series. We will then add the periodicity of the simple imputed series as shown below:



In the graph on the left, we can see the price variation at one hotel, with several missing values. As we can see in the graph on the right, the periodicity of the series (orange series) will be eliminated. We will then apply the desired imputation method (*ARIMA, interpolation, basic structural models, etc.*) to the remaining series (blue) and add back the periodicity of the imputed series.

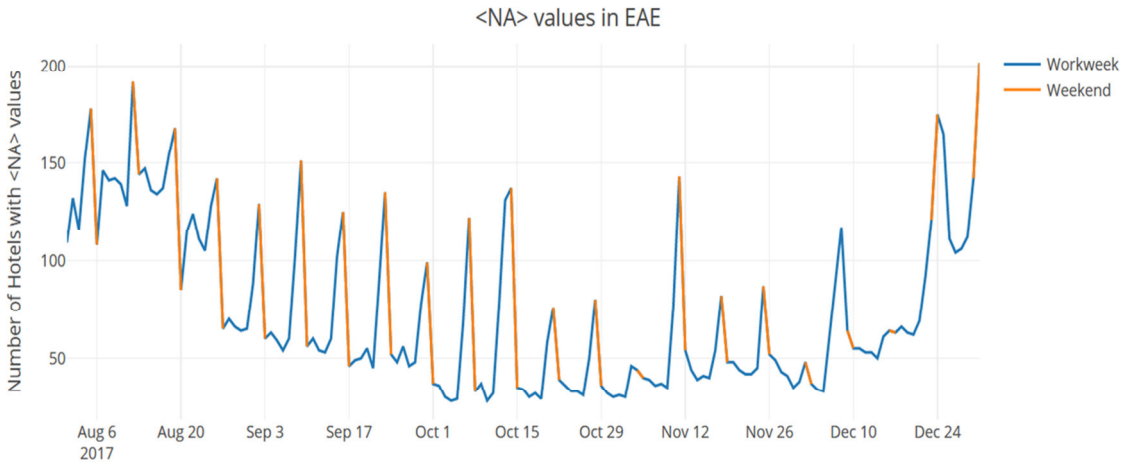
6.3 na.kalman

This function is based on the modelling of the series being analysed. Any series model can be applied to the function as required, though the function has two integrated modelling methods: *auto.arima* and *StructTS*. The function is based on Kalman filters.

Kalman filtering is also known as linear quadratic estimation (LQE) and uses values obtained over time to produce estimates of unknown values. It estimates a distribution of probability for each time window. This model is similar to hidden Markov models, with continuous space and Gaussian distribution of the variables.

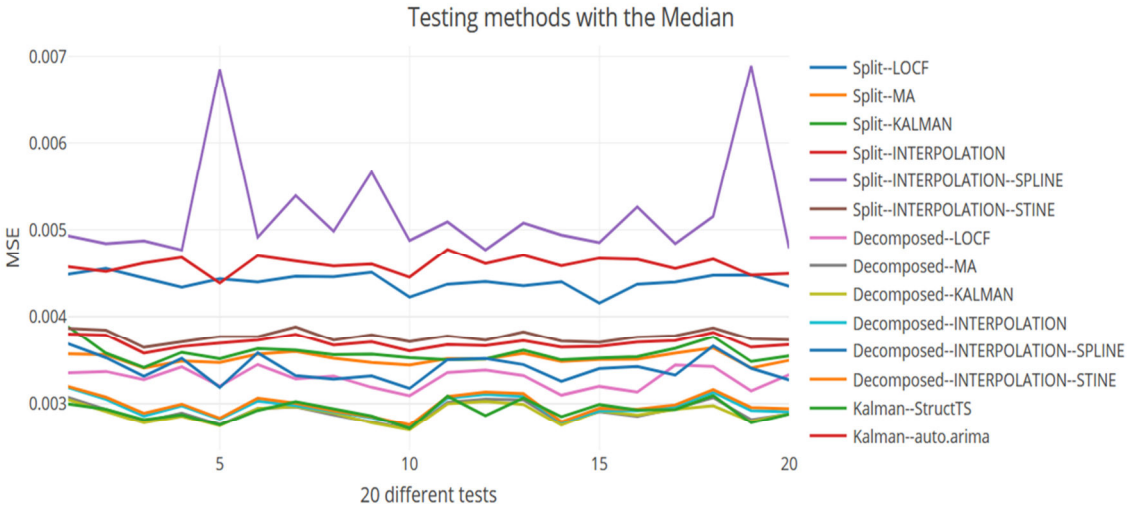
6.4 Selecting the optimal imputation

When selecting the final imputation method, we followed a process similar to that used for outliers. In this case, we have selected the complete series without outliers (approximately 150) and around 15-20 points have been randomly eliminated. Different methods were then applied to the series and the results have been compared to the complete series. Since we have noticed that the missing values form the series accumulate around weekends in particular (see below), the probability of the weekend value being eliminated from the complete series has increased.

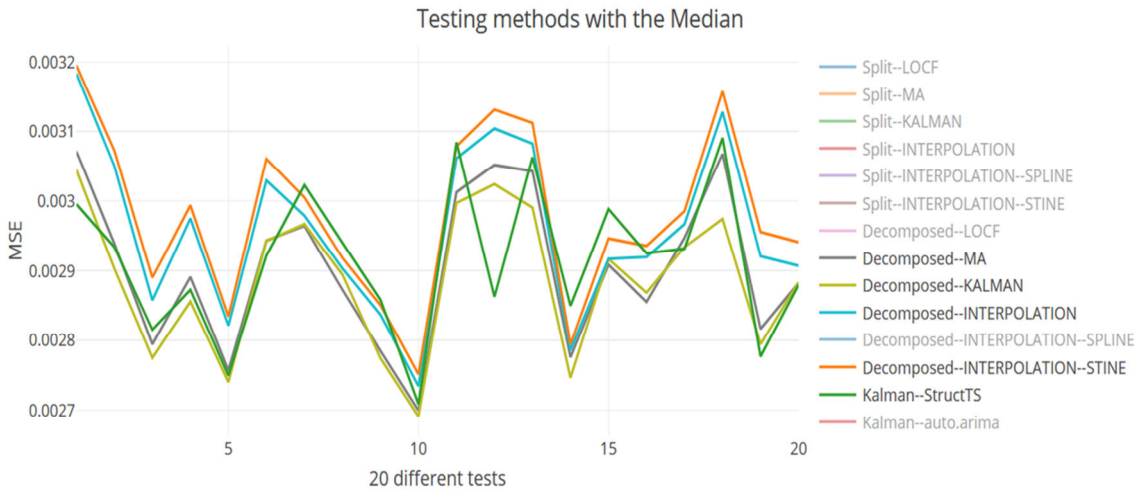


The mean squared error was used to measure the difference between the imputed series and the original.

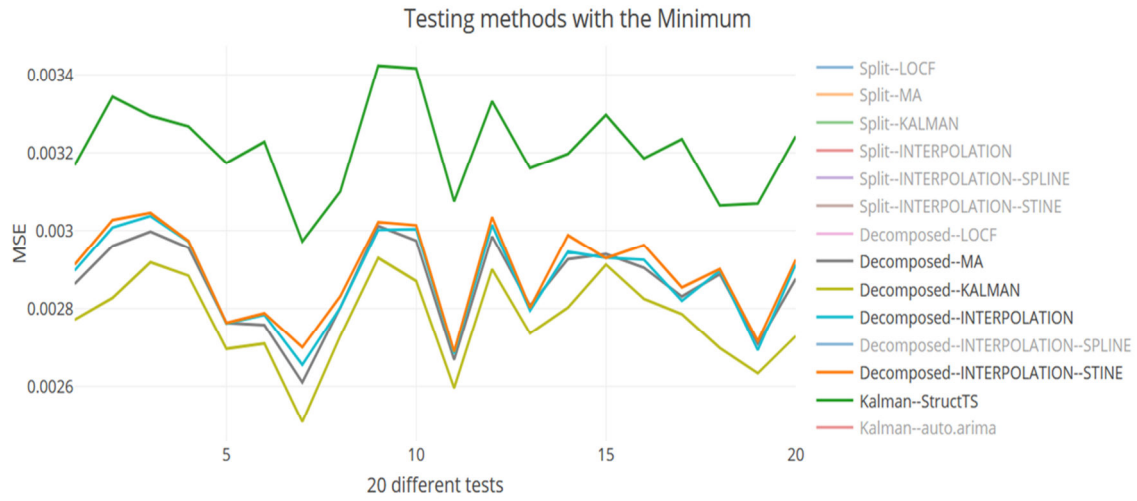
This process was carried out 20 times for each function in the *ImputTS* package and the same was done with the minimum daily hotel prices (these prices are more stable and have a more pronounced periodicity). The image below shows the error obtained in each period and with each method:



As we observe, the minimum errors were obtained with the *na.seadec* function of the *MA, kalman, interpolation* and *stin-interpolatio* model, and with the *na.kalman* function in StructTS:



The same goes for the minimum price analysis:



Considering all of these data, we have decided to use the *na.seadec* function with the Kalman filter.

7. Spatial autocorrelation

Let us consider $\Omega = \{w_i : i = 1, \dots, n\}$ as a spatially structured set (as in the example of the hotels we are analysing), where the spatial autocorrelation involves analysing the organisation model of the map Ω. If the magnitude value for the structure analysed at a point is relatively high/low, which in our case would be hotel price, and if the values of the adjacent magnitudes are also high/low, we will have positive (high) autocorrelation. However, if we have a relatively high/low magnitude value and it is surrounded by low/high values for that magnitude, there is a is negative (low) autocorrelation. We call the phenomenon whereby the variables adopt similar or different values compared to nearby variables spatial dependence. To measure this similarity or difference, we use spatial autocorrelation indices.

7.1 Basic definitions

When calculating correlation indices, there are two essential elements: the weight matrix and the structural matrix. The weight matrix, $P = [p_{ij}]$, depends on the values of the elements. It is a square matrix and, if we take its absolute values, it will be symmetrical. On the other hand, the structural matrix, $A = [a_{ij}]$, indicates the structure of the elements in space and is symmetrical.

There are numerous ways of defining the structural matrix. For example, we can consider the distance between the elements d_{ij} and define it as $a_{ij} = \frac{1}{d_{ij}}$. In this case, we have to take great care when $d_{ij} = 0$. We can also assign a value of 1 to the correlative elements in the structural matrix and a value of 0 to the other elements. In the case of hotels, the concept of correlation may indicate that establishments belong to the same stratum, municipality or province. We generally assign a 0 value to the component a_{ii}.

Once these matrices have been defined, the autocorrelation index will look like this:

$$\Gamma = \lambda \sum_i \sum_j a_{ij} p_{ij}.$$

The various spatial statistics used will be defined before the correlation indices. First, each element w_i will have a spatial weight of A_i, which is defined as follows:

$$A_i = \sum_j a_{ij}.$$

Knowing this, we can also define the total weight of the set.

$$A_{Tot} = \sum_i A_i = \sum_i \sum_j a_{ij}.$$

It should be noted that, when it is $a_{ii} = 0$, A_{Tot} will be even, given that $a_{ij} = a_{ji}$. Once the weight for each of the elements and the total weight have been defined, the adjusted average

and variance will be specified. In our case, we will give the price of each hotel w_i as x_i. First, the adjusted average is defined as follows:

$$\overline{x_A} = \frac{1}{A_{Tot}} \sum_i A_i x_i.$$

The main characteristic of the average is that the elements that have the most/least elements around them will have the most/least influence on the final value. In the case of the hotels, establishments in a city like San Sebastián will have a greater weight than the hotels in places such as Galdakao. It should be noted that the average also satisfies the properties of the standard average.

$$\frac{1}{A_{Tot}} \sum_i A_i (x_i - \overline{x_A}) = \frac{1}{A_{Tot}} \sum_i A_i x_i - \frac{1}{A_{Tot}} A_{Tot} \overline{x_A} = 0.$$

The adjusted variance is defined in an equivalent manner:

$$s_A^2 = \frac{1}{A_{Tot}} \sum_i A_i (x_i - \overline{x_A})^2.$$

As is the case with the average, the elements with a greater number of neighbouring elements will have a bigger impact than more isolated elements. Once this pair of statistics is defined, the next logical step would be to define the standardised variables. The adjusted average of these new variables would be 0, whereas the adjusted variance would be 1:

$$z_i = \frac{x_i - \overline{x_A}}{s_A^2}.$$

We can also characterise a hotel depending on its environment. The environment for the element w_i can be summarised as follows:

$$\mathrm{y_i} = \frac{1}{\mathrm{A_i}} \sum_{\mathrm{j}} \mathrm{a_{ij} x_j}.$$

Elements obtained in this way maintain the average of the x_i, and the set they comprise would be like a smoothed version of the initial set.

7.2 Global indices

Once the basic concepts have been defined, it is time to consider correlation indices. As we said before, the indices generally look like this:

$$\Gamma = \lambda \sum_i \sum_j a_{ij} p_{ij}.$$

First, we will discuss Moran's index, which takes multiplications of standardised elements as weights, $p_{ij} = \frac{(x_i - \bar{x})}{s}\frac{(x_j - \bar{x})}{s}$. As we are analysing the spatial aspect, we will use the adjusted average and variance $p_{ij} = z_i z_j$. The adjusted Moran's index will therefore be:

$$I^* = \frac{1}{A_{Tot}} \sum_i \sum_j a_{ij} z_i z_j = \frac{1}{A_{Tot}} \sum_i \sum_j a_{ij} \left(\frac{x_i - \overline{x_A}}{s_A^2}\right)\left(\frac{x_j - \overline{x_A}}{s_A^2}\right).$$

The higher the value of this index, the greater the correlation, and the more negative the value, the less correlation.

Another one is Geary's index, which uses subtractions from standardised elements such as elements from the weight matrix. In the case of the adjusted Geary's index, we would therefore have $p_{ij} = z_i - z_j$:

$$C^* = \frac{1}{2}\frac{1}{A_{Tot}} \sum_i \sum_j a_{ij} (z_i - z_j) = \frac{1}{2}\frac{1}{A_{Tot}} \sum_i \sum_j a_{ij} \frac{\left(x_i - x_j\right)^2}{s_A^2}.$$

The Geary's index would therefore take positive values. The closer we are to 0, the greater the correlation and the higher the value, the greater the negative correlation.

Thirdly, we have Lebart's index. In this case, the adjusted Lebart's index, the square of the difference between the elements and their environment, would take $p_{ij} = \frac{(x_i - y_i)^2}{s_A^2}$, as weight:

$$L^* = \frac{1}{A_{Tot}} \sum_i \sum_j a_{ij} \frac{(x_i - y_i)^2}{s_A^2} = \frac{1}{A_{Tot}} \sum_i A_i \frac{(x_i - y_i)^2}{s_A^2}.$$

As with Geary's index, the closer the index is to 0, the more positive the correlation. However, the higher the value of this index, the greater the negative correlation.

The fourth index is defined in paper [11], η_1. As an element in the weight matrix, it uses the square of the difference between the elements' environment and the adjusted average, $p_{ij} = (y_i - \overline{x_A})^2$:

$$\eta_1^* = \frac{1}{A_{Tot}} \sum_i \sum_j \frac{a_{ij}(y_i - \overline{x_A})^2}{s_A^2} = \frac{1}{A_{Tot}} \sum_i A_i \frac{(y_i - \overline{x_A})^2}{s_A^2}.$$

Finally, we will define another index that appears in the same paper indicated, the index η_2. In this case, the environment of the element $p_{ij} = \left(y_i - x_j\right)^2$ is compared with other elements through subtraction, and the square is calculated:

$$\eta_2^* = \frac{1}{A_{Tot}} \sum_i \sum_j a_{ij} \frac{(y_i - x_j)^2}{s_A^2} .$$

When these two indices are closed to 0, they indicate a positive autocorrelation, and the higher the value, the more negative the correlation.

7.3 Local indices

While global indices measure the spatial correlation of the entire set, it would be interesting to observe the impact of each element on the overall spatial correlation. To do so, local correlation indices are defined based on the criteria established by Anselin[2]:

1. Local indices show the impact that elements have on the global index.
2. The sum of the local indices for the set of elements is proportional to the global index.

Based on these criteria, the local parts of the indices we have seen in the previous section can be defined.

1. **Moran:** $I_i^* = \frac{1}{A_i} \sum_{j=0}^{n} a_{ij} (\frac{x_i - \overline{x_A}}{s_A})(\frac{x_j - \overline{x_A}}{s_A})$.
2. **Geary:** $C_i^* = \frac{1}{2} \frac{1}{A_i} \sum_{j=0}^{n} a_{ij} \left(\frac{x_i - x_j}{s_A}\right)^2$.
3. **Lebart:** $L_i^* = \left(\frac{x_i - y_i}{s_A}\right)^2$.
4. $\boldsymbol{\eta_1}$: $\eta_{2i}^* = \left(\frac{y_i - \overline{x_A}}{s_A}\right)^2$.
5. $\boldsymbol{\eta_2}$: $\eta_{2i}^* = \frac{1}{A_i} \sum_{j=0}^{n} a_{ij} \left(\frac{y_i - x_j}{s_A}\right)^2$.

Here, each local index indicates the impact of the hotel w_i on the global index. Furthermore, global indices can be obtained from the linear combination of local indices. Let us suppose that Γ is a global index and that Γ_i represents its local index. Thus:

$$\Gamma = \frac{1}{A_{Tot}} \sum_i A_i \Gamma_i .$$

To put it another way, all of the general indices are proportional to the sum of the local indices for all of the elements. In particular, all of the global indices are the average of the local indices for the set of elements. We can therefore state that all of the criteria established at the outset have been satisfied and that the local indices are thus well defined.

7.4 Hotel indices

After defining the indices and before starting to apply them, two final aspects must be resolved. On the one hand, we have to define the structural matrix. On the other, not only the hotels' location but also their category influence their prices. The impact of the category must therefore be neutralised. Failure to consider this issue might mean the analysis gives erroneous results. For example, upmarket hotels surrounded by hotels with a lower number of

stars stand out because of the impact of the higher prices, which is attributable to category rather than location.

In order to negate the impact of category, three different analyses have been conducted. For a better understanding, let us suppose that we have the following data:

Hotel	Category	Date	Cost
1	H3	19/08/2039	75
1	H3	15/11/2039	56
2	H5	19/08/2039	500
2	H5	15/11/2039	300
3	P1	19/08/2039	30
3	P1	15/11/2039	15
4	H3	19/08/2039	90
4	H3	15/11/2039	60
4	H3	15/07/2039	75

Firstly, in the initial comparison, the hotel prices were assigned values between 0 and 100, depending on the prices in their categories. In other words, the maximum price in a category has been given the value 100, whereas the others have been given values of 0–100 proportionally.

Hotel	Category	Date	Value_1
1	H3	19/08/2039	55
1	H3	15/11/2039	0
2	H5	19/08/2039	100
2	H5	15/11/2039	0
3	P1	19/08/2039	100
3	P1	15/11/2039	0
4	H3	19/08/2039	100
4	H3	15/11/2039	11
4	H3	15/07/2039	55

Secondly, the hotels were analysed by category, i.e. the hotels were compared only with hotels in the same category.

Hotel	Category	Date	Value_2
1	H3	19/08/2039	75
1	H3	15/11/2039	56
2	H5	19/08/2039	500
2	H5	15/11/2039	300
3	P1	19/08/2039	30
3	P1	15/11/2039	15
4	H3	19/08/2039	90
4	H3	15/11/2039	60
4	H3	15/07/2039	75

Finally, instead of looking at the price of the hotels, we analysed the price trend. Each hotel was analysed independently and the prices were converted to a number ranging from 0 to 100. In this case, we will attribute a value of 100 to the day on which the hotel reaches its historic maximum, a 0 to the day on which it reaches the minimum, while the remaining days will have a proportional value. Furthermore, we will assign a value of 50 to the hotels whose prices remain constant.

Hotel	Category	Date	Value_3
1	H3	19/08/2039	100
1	H3	15/11/2039	0
2	H5	19/08/2039	100
2	H5	15/11/2039	0
3	P1	19/08/2039	100
3	P1	15/11/2039	0
4	H3	19/08/2039	100
4	H3	15/11/2039	0
4	H3	15/07/2039	50

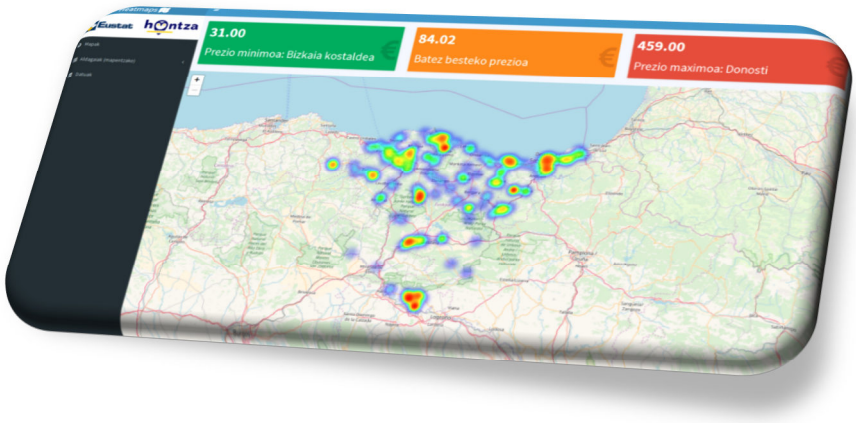
Once the impact of category has been minimised, it is time to select the structural function. Here we will also analyse three different cases, in all of which $a_{ii} = 0$. Firstly, when analysing the hotel w_i, $a_{ij} = 1$ was considered only when hotel w_j was in the same stratum. In the second and third cases, the distances between the hotels was calculated. The function *distHaversine()* from R software's *geosphere* package was used. The function measures the distance between the points, taking into account the curvature of the Earth. The second case $a_{ij} = 1$ only if the hotel w_j is at a maximum predefined distance *r* from hotel w_i. The distances $r = 3km$ and $r = 5km$ were used for the analysis. Finally, the impact of all of the hotels was analysed in the third case. For this the definition $a_{ij} = \frac{1}{d_{ij}}$ was used, where the value of d_{ij} represents the distance between hotels w_i and w_j. All of the other establishments will thus be considered in the analysis of each of the hotels w_i, giving greater weight to the nearest ones.

It should be noted that we will obtain different results depending on the indices used. Geary's index, for example, compares the price of the hotel analysed with the other prices, whereas the η_1, on the other hand, will not take the analysed hotel into account because it compares the location of the selected hotel and the average across the hotels as a whole. For this reason, of all of the indices we have considered, we will highlight the ones most relevant to our particular case below.

Firstly, given that we can show that Moran's and Geary's indices are equivalent[11], we have decided to eliminate Moran's index and work only with positive values. Furthermore, because the purpose of the project is to analyse hotel prices, η_1 and η_2 are not part of the analysis, seeing that they consider the price of the surrounding hotels rather than the price of the hotel analysed.

Taking these points into account, the results were examined using Geary's and Lebart's indices, with the different types of matrix as explained above.

8. Heat map



8.1 Software used

We used R software to develop a heat map. A graphical application was developed using the *shiny* package which included, among other things, an interactive map with the help of the *leaflet* and *leaflet.extras* packages.

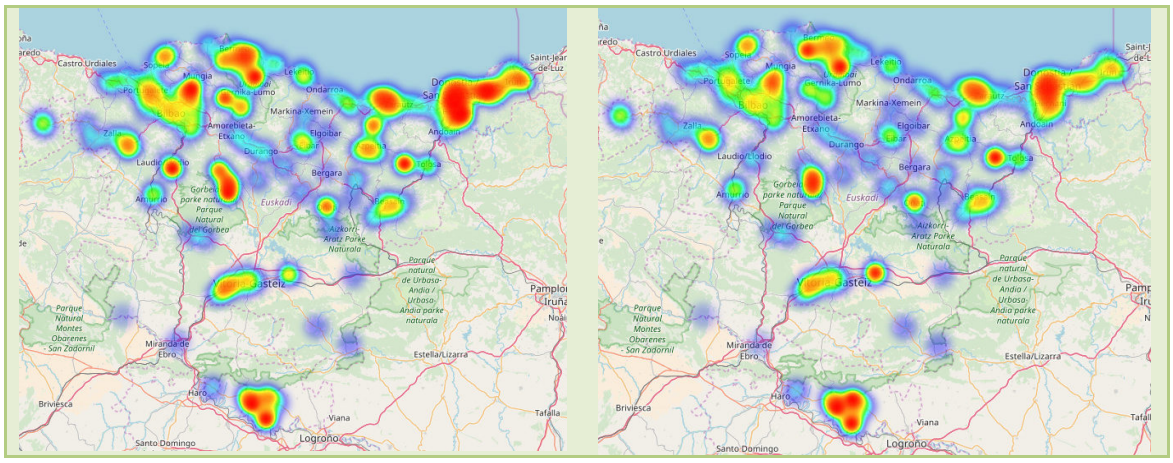
8.2 Visualised data

The heat map shows the hotel-related information in a different way. Firstly, there is a heat map for the prices themselves; a second one shows the price trend; and the third map displays the spatial autocorrelation. The level of zoom will indicate the value of the hotels or the area under analysis.

8.2.1 Heat map of prices

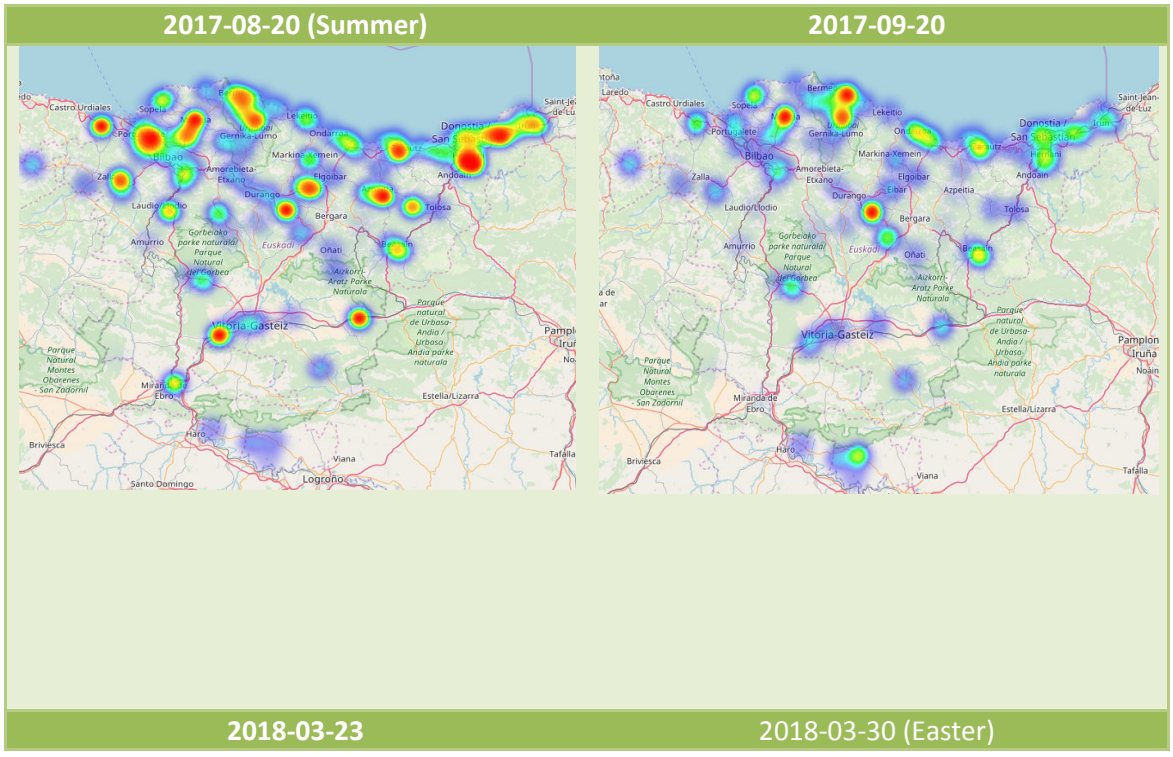
This is the most simple map of all and, as might be expected, it shows the variation in prices of hotel establishments over time. The higher the hotel price, the redder the map, while the lower the values, the more blue the colour will be. The value of the hotel or area analysed will produce the most intense red colour when it is equal to or higher than the user-defined value.

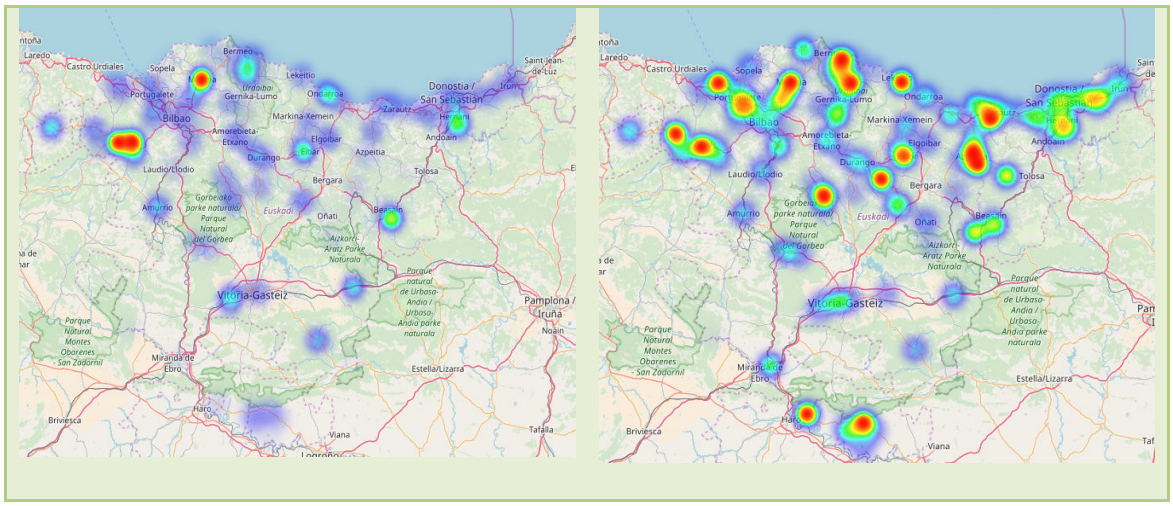
2019-08-20	2018-09-20



8.2.2 Price trend

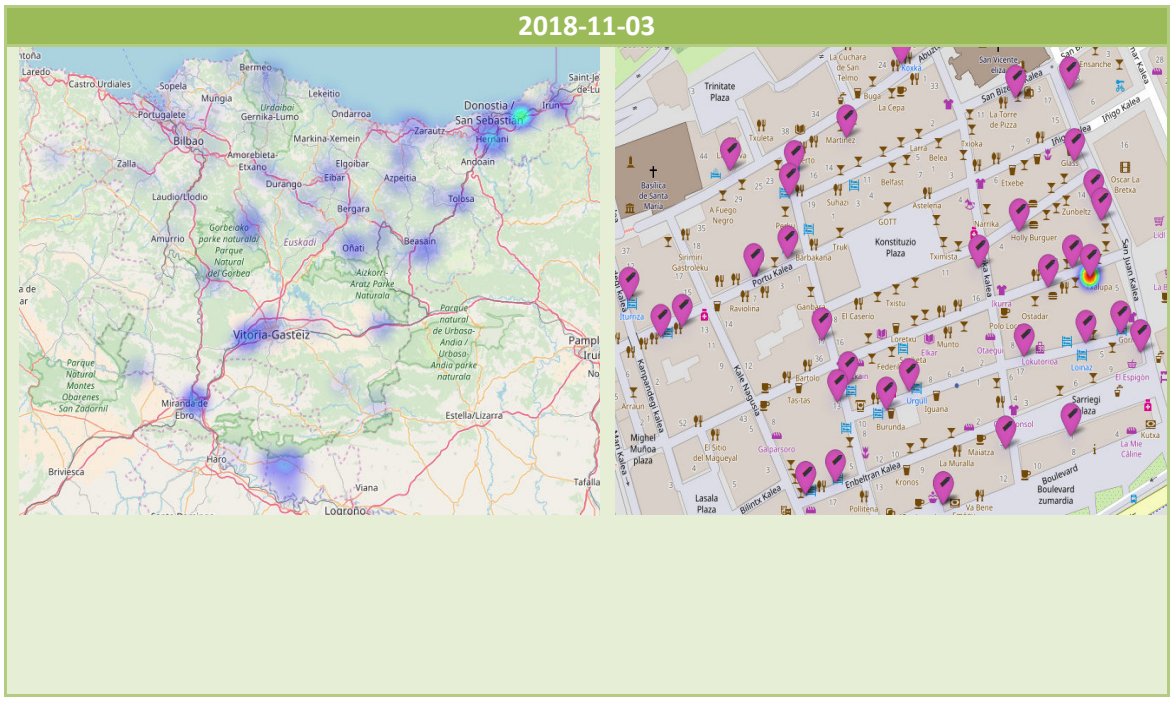
The main problem with the first method is that it is sometimes difficult to visualise price variation. For example, relating to the Basque Country, prices in Donostialdea will almost always be higher than those in Vitoria-Gasteiz and will therefore have more intense shades of red. To avoid this, a second visualisation method has been developed which displays price trends instead of hotel prices. After analysing each hotel individually, their price was translated into a range of 0–100. Once this is done, we can see the general price variation. For example, we can see that hotels increase their prices during the summer and at Easter, and that prices go down from September.





8.2.3 Spatial correlation

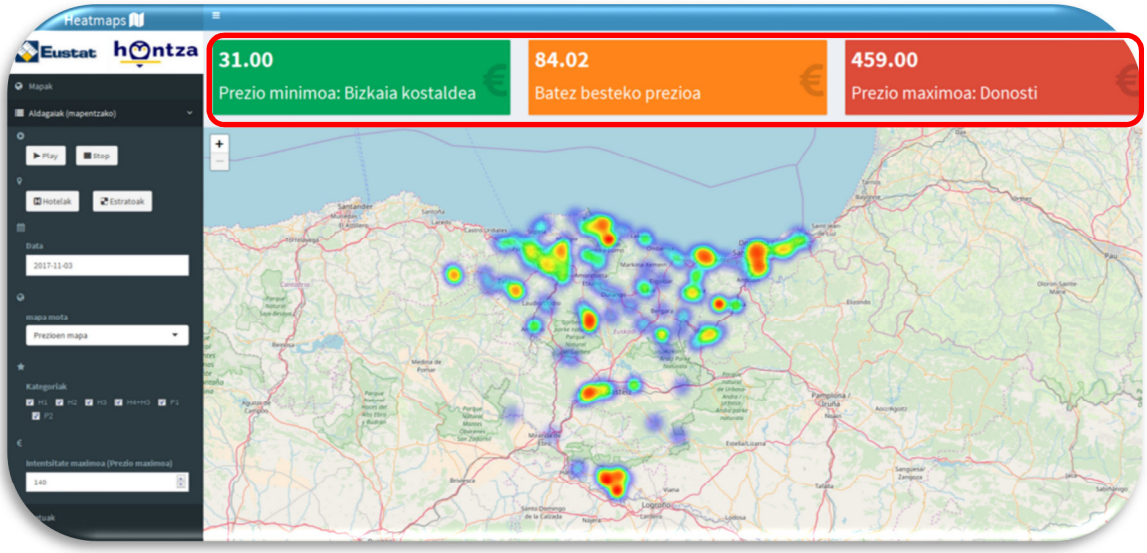
Finally, we can visualise the information provided by the spatial correlation indices. As we have observed, the correlation indicates whether there are data similar to or that stand out from the rest of the data. It should be noted that, in this case, we should go into detail to obtain information given that hotels in the Basque Country are generally similar to the hotels around them. For example, as we can see in the images below, we might think that there are no hotels that stand out on 3 November if we zoom out; however, if we zoom in on Donostialdea, we will see that there is a guesthouse in the Old Town that stands out. In this case, it is priced at 360 euros per night, whereas the guesthouses around it charge 50–100 euros.



8.3 Additional functionalities

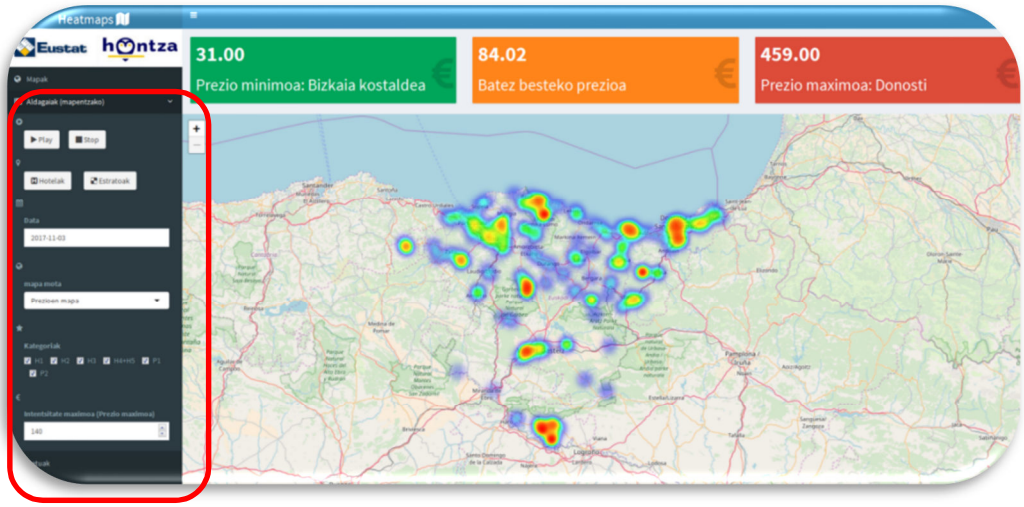
In order to obtain additional information, we have added a series of complementary functionalities to the heat map.

8.3.1 General data by day



The first thing we see when opening the application is a set of boxes appearing on the map. It displays a series of general data for a particular day, such as the minimum, average and maximum prices. The boxes change colour according to the values they show. We should recall that the value with the most intense shades of red is user-selected. The remaining box colours will also be chosen based on this colour. When it is equal to or greater than the maximum value, the box will be red. If it has values ranging between the maximum and the average, the box will be orange. Finally, if the value is lower than the average, the box will be green.

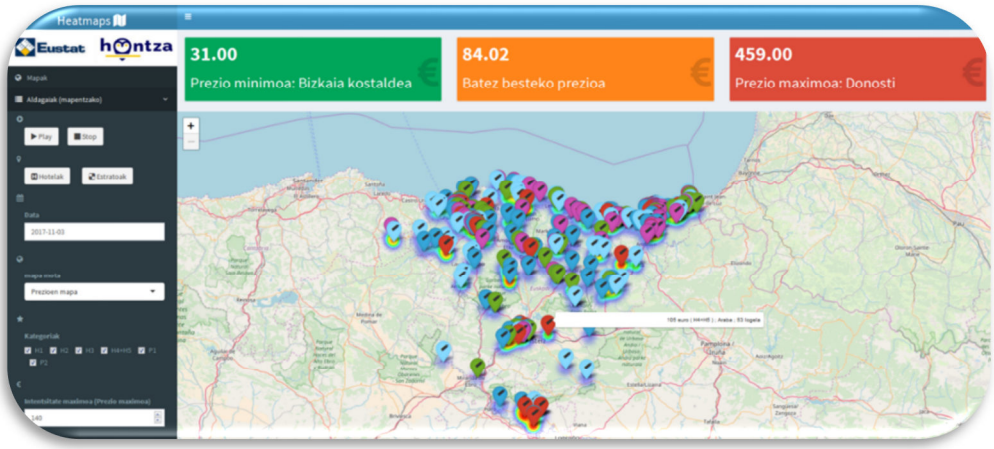
8.3.2 Map control menu



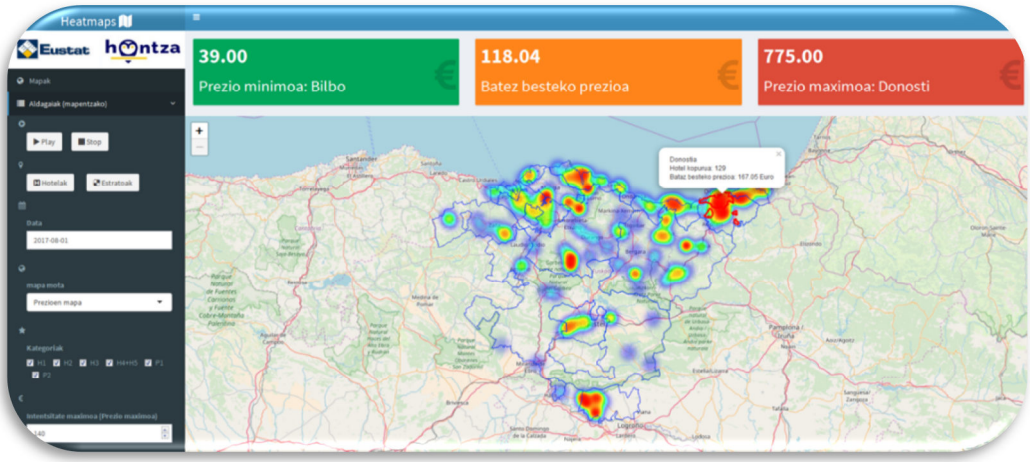
The application also has a section specifically for controlling the map interactively. Firstly, we can move the graphic using the “Play” and “Stop” buttons. When the map starts to move, the values of the boxes update as the map’s position is updated.

Under this option, we have the “Hotels” and “Strata” buttons. As we would expect, it gives us additional information on hotels and strata.

If we click on “Hotels”, a new point appears for each hotel, displaying the relevant information. As we can see in the image below, the points will have different colours according to the hotel category. For example, guesthouses will be purple, three-star hotels will be light blue, and four- and five-star hotels will be red. Furthermore, if we move the mouse over the points, we will see information on the hotels, such as their name, price, category, province and number of rooms.



If we click on “Strata”, the different limits that separate the strata in the autonomous region will be displayed. If we click on them, a small information sheet will appear for the day being analysed, including the name of the stratum, the number of hotels and the average price.



Lastly, we can see that we have another four boxes: “Date”, “Map type”, “Categories” and “Maximum intensity”. The first allows us to manually change the day being analysed. The “Map type” button allows us to choose from the different visualisation types, such as heat

map, trend map or spatial correlation. It is also possible to view the empty map as an additional function.

The “Categories” section allows us to filter the hotels by category and to view of those of interest. Finally, the “Maximum intensity” button enables us to select the value that defines the maximum intensity.

Bibliography

[1] Ander Juarez Mugarza. *Autokorrelazio espazialeko indizeetan oinarritutako ertz detekziorako metodoak*. (tesis de máster). Euskal Herriko Unibertsitatea/Universidad del Pais Vasco, Donostia/San Sebastian.

[2] Anselin, L. (1995). Local indicators of spatial association—LISA. Geographical analysis, 27(2), 93-115.

[3] Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[4] Canavos, G. *Probabilidad y Estadística*. McGraw Hill, 1994.

[5] Capp_e, O., Moulines, E., and Ryden, T. *Inference in Hidden Markov Models (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[6] Casella, G., and Berger, R. *Statistical Inference*. Duxbury Resource Center, June 2001.

[7] Chang, W., Cheng, J., Allaire, J., Xie, Y., and McPherson, J. *shiny: Web Application Framework for R*, 2017. R package version 1.0.5.

[8] Chen, E. Winning the Netix Prize: A Summary. http://blog.echen.me/2011/10/24/winning-the-netflix-prize-a-summary/.

[9] Cheng, J., Karambelkar, B., and Xie, Y. *leaflet: Create Interactive Web Maps with the JavaScript 'Leaet' Library*, 2018. R package version 2.0.0.

[10] Concepción, Morales, Eduardo René (2010). *Contribuciones a la segmentación de imágenes mediante algoritmos de clasificación automática*. (Tesis doctoral). Euskal Herriko Unibertsitatea/Universidad del Pais Vasco, Donostia/San Sebastian.

[11] de Lacalle, J. L. *tsoutliers: Detection of Outliers in Time Series*, 2017. R package version 0.6-6.

[12] Devore, J. L. *Probability and Statistics for Engineering and the Sciences*, 8th ed. Brooks/Cole, January 2011. ISBN-13: 978-0-538-73352-6.

[13] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[14] Grubbs, F. E. *Sample criteria for testing outlying observations*. Ann. Math. Statist. 21, 1 (03 1950), 27–58.

[15] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[16] Hyndman, R. J. *forecast: Forecasting functions for time series and linear models*, 2017. R package version 8.2.

[17] James, G., Witten, D., Hastie, T., and Tibshirani, R. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014.

[18] Komsta, L. *outliers: Tests for outliers*, 2011. R package version 0.14.

[19] Koren, Y., Bell, R., and Volinsky, C. *Matrix factorization techniques for recommender systems*. Computer 42, 8 (Aug 2009), 30–37.

[20] Karambelkar,B. and Schloerke, B. (2018). *leaflet.extras: Extra Functionality for 'leaflet' Package*. R package version 1.0.0. https://CRAN.R-project.org/package=leaflet.extras

[21] Kriesel, D. *A Brief Introduction to Neural Networks*. 2007.

[22] LESART, L. *Analyse statistique de la contiguïté*.1969.

[23] Moran, P. A. (1950). *Notes on continuous stochastic phenomena*. Biometrika, 37(1/2), 17-23.

[24] Moritz, S. *imputeTS: Time Series Missing Value Imputation*, 2017. R package version 2.5.

[25] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017.

[26] Rabiner, L. R. *Readings in speech recognition*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990, ch. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pp. 267-296.

[27] Ricci, F., Rokach, L., Shapira, B., and Kantor, P. B. *Recommender systems handbook*. Springer, New York; London, 2011.

[28] Robert J. Hijmans (2017). *geosphere: Spherical Trigonometry*. R package version 1.5-7.https://CRAN.R-project.org/package=geosphere

[29] Sievert, C., Parmer, C., Hocking, T., Chamberlain, S., Ram, K., Corvellec, M., and Despouy, P. *plotly: Create Interactive Web Graphics via 'plotly.js'*, 2017. R package version 4.7.1.

[30] Strang, G. *Linear algebra and its applications*. Thomson, Brooks/Cole, Belmont, CA, 2006.

[31] Wickham, H. *Reshaping data with the reshape package*. Journal of Statistical Software 21, 12 (2007), 1-20.

Organismo Autónomo del

EUSKO JAURLARITZA
GOBIERNO VASCO

Eustat
EUSKAL ESTATISTIKA ERAKUNDEA
INSTITUTO VASCO DE ESTADÍSTICA
www.eustat.es

www.eustat.es