VNiVERSiDAD
D SALAMANCA
CAMPUS DE EXCELENCIA INTERNACIONAL

800 AÑOS
VNiVERSiDAD
D SALAMANCA

FACVLTAD DE CIENCIAS

GRADO EN ESTADÍSTICA

TRABAJO DE FIN DE GRADO

MÉTODOS DE APRENDIZAJE AUTOMÁTICO APLICADOS AL DESARROLLO DE LA BIOINFORMÁTICA

Autora: Pilar Franco Martín

Tutor: Dr. José Antonio Castellanos Garzón

Salamanca, 2021

VNiVERSiDAD D SALAMANCA

CAMPUS DE EXCELENCIA INTERNACIONAL

FACVLTAD DE CIENCIAS

GRADO EN ESTADíSTICA

TRABAJO DE FIN DE GRADO

MÉTODOS DE APRENDIZAJE AUTOMÁTICO APLICADOS AL DESARROLLO DE LA BIOINFORMÁTICA

Autora: Pilar Franco Martín

Tutor: Dr. José Antonio Castellanos Garzón

Salamanca, 2021

FACULTAD DE CIENCIAS

GRADO EN ESTADÍSTICA

TRABAJO DE FIN DE GRADO

MÉTODOS DE APRENDIZAJE AUTOMÁTICO APLICADOS AL DESARROLLO DE LA BIOINFORMÁTICA

Autora: Pilar Franco Martín **Tutor**: Dr. José Antonio Castellanos Garzón

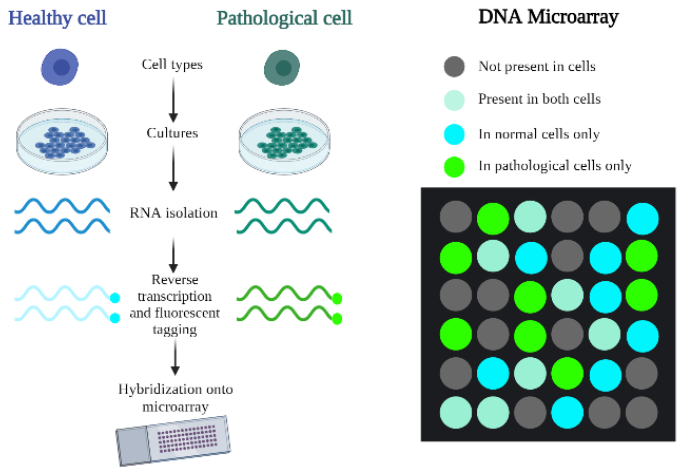
Salamanca, 2021

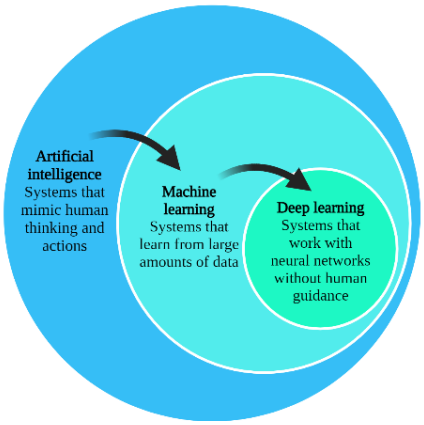
SUMMARY

Before we get into the project itself, it is convenient to introduce some basic concepts in order to have a better base. Big Data consists of a process that analyzes and interprets large volumes of data, both structured and unstructured. Big Data serves so that data stored remotely can be used by companies as the basis for their decision making. It is a tool used very frequently in statistics. Another important concept to be known might be data mining, which can be defined as the process of finding anomalies, patterns, and correlations in large data sets (Big Data) to predict outcomes.

As the purpose of this project is to link machine learning with bioinformatics, it is time to explain what bioinformatics consists of. Well, bioinformatics allows investigating, developing and applying computational tools to enable and improve the biological data handling. One of its applications is the management of the diagnostic technologies automation.

It is very common to use DNA microarrays in bioinformatics. Microarray technology is a developing technology to study the expression of many genes at the same time. It consists of placing thousands of gene sequences in specific places on a glass slide called chip. A sample containing DNA or gets in touch with the chip. Complementary base pairing between the sample and the gene sequences on the chip produces a measurable amount of light. The light-producing areas of the chip identify the genes which are expressed in that sample. This mechanism is the beginning of studies related to gene expression data, in which machine learning is used.





Artificial intelligence is the combination of algorithms proposed with the objective of creating machines with the same capabilities than humans. Machine learning is a discipline in the field of artificial intelligence that, though algorithms, provides computers with the ability to identify patterns in massive data to make predictions. Deep learning is a type of machine learning that trains a computer to perform tasks like humans do, such as speech recognition, image identification, or making predictions.

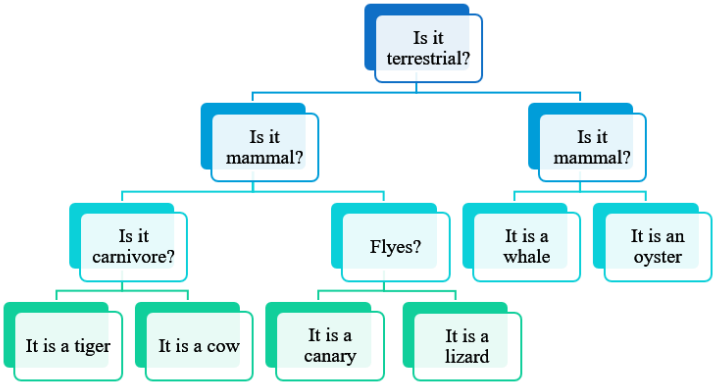
Machine learning techniques can be divided into supervised, unsupervised, semi-supervised and reinforcement learning.

The main difference between supervised and unsupervised learning is that in the supervised one we know the categories in which we want to divide the data. As its own name indicates, semi-supervised learning mixes parts of both machine learning types mentioned before. In problems solved by reinforcement learning, the solution is not known and the way to train the model is through positive or negative reinforcement depending on the results, causing a decision to be made. This approach is used when it is possible to assign a reward or a penalty but it is not known how to reach the result.

The next step is to explain some of the most important machine learning techniques.

To start, we can talk about decision trees. It is a technique that makes possible analysing sequential decisions based on the use of results and associated probabilities. Decision trees can be used to generate expert systems, binary searches and game trees. A tree can be considered a particular type of graph, a data arrangement made up of a set of hierarchically ordered nodes and vertices. We have different types of nodes: root, internal and terminal. All nodes have a single incoming vertex, except the root node. Trees lack loops (not like graphs).

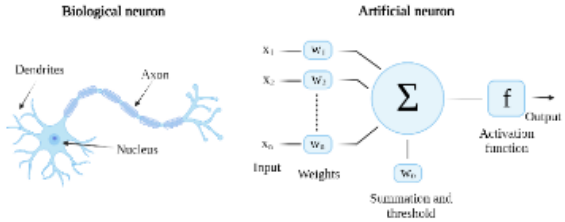
Survival trees are a class of classification and regression trees prepared to deal with censored data. The most important difference between a survival tree and a decision tree is the division criterion. In a decision tree, partitions are recursively when a threshold for each characteristic is decreed, but the interactions between the characteristics and the censored information from the model are not taken into account.



A decision forest is a set of decision trees that start randomly. The most important characteristic of this technique is that the trees that make up the forests are different from each other in a random way. This makes the system more general and robust.

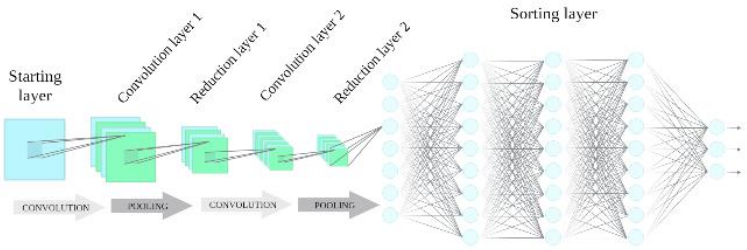
Association rules are algorithms that aim to discover connections in a set of items or attributes likely to happen at the same time. The most frequent algorithm of association rules is known as "*a priori*", which finds the most common relationships between items, performs iterations until the relationships do not have the minimum support. It is a robust technique as well as simple and provides a quite intuitive output.

Genetic algorithms are search mechanisms based on the natural selection and genetic laws. They combine the best adapted individuals survival along with genetic search operators such as mutation and crossing, hence they are comparable to a biological search. These algorithms are used successfully for a great variety of problems that do not allow an efficient solution through the application of conventional techniques. The hypotheses of a genetic algorithm are represented by bit strings, so that the crossover and mutation operators can handle them easily. Genetic operators join and mutate selected individuals from the population to become part of the offspring. The most common genetic operators are single point crossing, two point crossing, uniform crossing and mutation point. The adaptation function, also known as fitness, determines the pattern to follow to order potential hypotheses and to choose them through probability, with the aim of incorporating them into the next generation of the population. Being the task of learning classification rules, the adaptation function has an element that evaluates the hypotheses on the training data precision. This function has to measure the general performance of the rules, in addition to the individual precision.

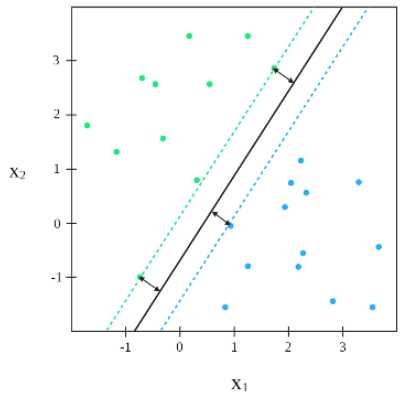


Artificial neural networks are artificial intelligence models inspired by the neurons and brain connections behaviour to solve problems. They are inspired by the nervous system and biological behaviour, creating a layered interconnection system of artificial neurons that collaborate to process input data and generate output. They are used for decision-making in business management, prediction, trend

recognition, pattern recognition and risk management (applied for example in fraud detection), home automation, autonomous vehicles and renewable energies, among other functions.



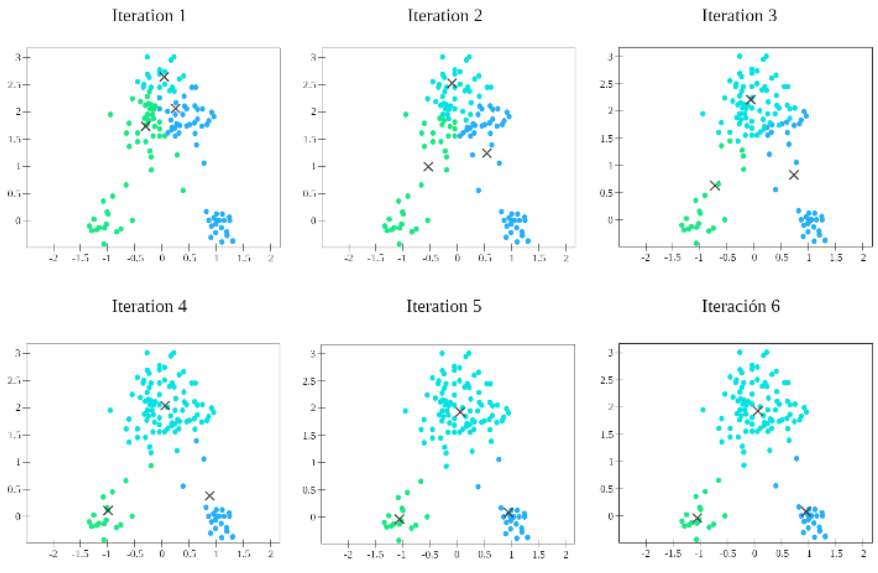
Convolutional neural networks is a deep learning algorithm that is designed to work with images, taking these as input, assigning importance (weights) to certain elements in the image in order to differentiate them from each other. This is one of the main algorithms that has contributed to the development and improvement of the computers field of view. Convolutional networks have several hidden layers, where the first ones can detect lines, curves and thus become specialized until they can recognize complex shapes such as a face, silhouettes, etc. The common tasks of this type of networks are: detection or categorization of objects, scene classification and image classification in general. The network takes as input the pixels of an image.



Support Vector Machines (SVM) are a machine learning technique that allow you to find the optimal way to classify among various classes. The optimal classification is done by maximizing the separation margin among the classes. The vectors which define the edge of this separation are the support vectors. When the classes are not linearly separable, we can use the kernel trick to add a new dimension where they are. Some success cases of support vector machines are: optical character recognition, face detection for digital cameras to focus properly, spam filters for email and image recognition on board satellites.

Clustering algorithms are within unsupervised learning, since clustering is not based on previously defined groups. It is a task whose main purpose is to achieve the grouping of unlabeled sets of objects, in order to build subsets of data known as clusters. Each cluster within a graph is formed by a collection of objects or data that in terms of analysis are similar to each other, but they have differential elements regarding other objects belonging to the data set and that can form an independent cluster.

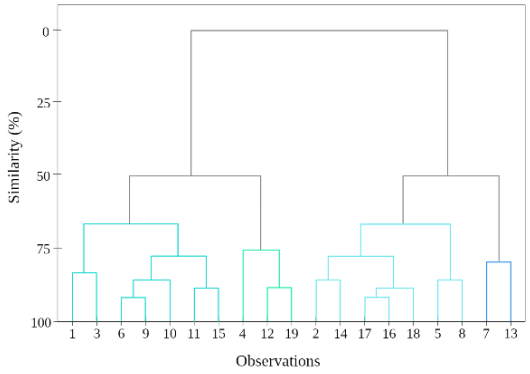
K-means is a clustering method that aims to generate a partition of a set of n observations into k groups. Each group is represented by the average of the points that compose it. The representative of each group is called the centroid. The number of groups to discover, k, is a parameter that must be set a priori. The clustering method starts with k randomly located centroids, and assigns each observation to the closest centroid. After being assigned, the centroids are moved to the average location of all the data assigned to it, and the points are reassigned according to



the new positions of the centroids. The objective of k-means is to group the observations in such a way that all those that are in the same group are the most similar to each other and that those belonging to different groups are the most dissimilar to each other. Distance measures, like the *Euclidean*, are used to measure similarity and dissimilarity. A measure of how well the centroids represent the members of your group is the sum of the squared errors. K-means, in each iteration, tries to reduce the value of the sum of the squared errors. The measure consists of the sum of the squared distances of each observation from the centroid of its group. The algorithm always ends, since it does not necessarily find the most optimal configuration, the one that corresponds to the minimum of the objective function. Finding a minimum of the function, even though it is not the absolute minimum, guarantees a grouping in which the groups are sparsely dispersed and are separated from each other. The algorithm is significantly sensitive to centroids that are initially randomly selected. This effect can be reduced by performing multiple runs of the method.

K nearest neighbors is one of the most basic and essential classification algorithms in machine learning. It belongs to the domain of supervised learning and finds intense application in pattern recognition, data mining, and intrusion detection. It is non-parametric, because it does not make explicit assumptions about the functional form of the data, avoiding mis-modeling the underlying distribution of the data. This algorithm does not explicitly learn a model. Instead, it chooses to memorize the training instances that are later used as "knowledge" for the prediction phase. Specifically, this means that only when a query is made to our database, that is, when we ask it to predict a label with an input, will the algorithm use the training instances to give an answer.

The hierarchical cluster algorithm groups data based on the distance between each one and looking for the data within a cluster to be the most similar to each other. In a graphical representation, the elements are nested in tree-shaped hierarchies.



Two types of hierarchical clustering can be defined depending on the direction in which the algorithm executes the clustering: agglomerative type (we begin to group from each individual element; at the beginning each point or data is in a separate cluster; at each step, the two closest clusters merge; these cluster mergers continue to occur in succession, producing a hierarchy of clustering results; at the end of the process, there is only a single cluster that brings together all the elements) ans divisible one (we start in reverse, from a single cluster that brings together all the data and we divide into smaller clusters).

The way to represent a hierarchical clustering is with a dendrogram. The vertical lines of the dendrogram illustrate the mergers (or splits) performed at each stage of clustering. We can see the distance, the different levels of associations between the individual data, and also the associations between clusters.

Bayesian networks provide a graphical representation for a set of random variables and for the relationships among them. The structure of the network allows the joint probability function of these variables to be specified as the product of conditional probability functions, generally simpler. This approach represents a good strategy to deal with problems related to uncertainty, where the conclusions cannot be built only from a previous knowledge about the problem.

Bayesian networks are among the most popular graphical models. The main difference, regarding other models, is that their arcs are directed and they represent conditional dependence between the variables. The name comes from the fact that much of the theory relevant to these types of networks is based on bayesian statistics.

Principal Component Analysis (PCA) is an unsupervised learning statistical method that simplifies the complexity of sample spaces with many dimensions while preserving their information. Suppose there is a sample with n individuals each with p variables $(X_1, X_2, \dots, X_p)$, that is, the sample space has p dimensions. PCA allows finding a number of underlying factors $(z < p)$ that explain approximately the same as the original p variables. Where before p values were needed to characterize each individual, now z values are enough. Each of these z new variables is called a principal component.

The goal of unsupervised learning is to extract information using predictors, for example, to identify subgroups. The main problem that unsupervised learning methods face is the difficulty in validating the results given that there is no response variable available to contrast them.

The PCA method, therefore, allows the information provided by multiple variables to be "condensed" into only a few components. This makes it a very useful method to apply after using other statistical techniques such as regression, clustering ... Even so, we must not forget that it is still necessary to have the value of the original variables to calculate the components.

The exponential increase in the amount of biological data constitutes one of the main challenges of computational biology, which requires the development of tools and methods able to transform these heterogeneous data into biological knowledge about the mechanism in question. These tools should provide insights in the form of testable models. Thanks to this abstraction, predictions of the system can be obtained.

Bioinformatics belongs to one of the newest areas of research. It has an infinity of tools designed to solve biological problems that require the management of large databases, distributed biological information, representations of difficult knowledge and the formulation of models about the functioning of cellular systems and predictions about their behaviour.

Machine learning can be used in branches such as proteomics, microarray studies, systems biology, the evolution and reconstruction of phylogenetic trees, text mining in computational biology... However, the main biological area in which machine learning is used is genomics.

In the genomics area, next-generation sequencing has quickly advanced the field by sequencing a genome in a short time. Thus, an active area of machine learning is applied to the identification of gene coding regions in a genome. These gene prediction tools involving machine learning would be more sensitive than typical homolog-based sequence searches.

Microarrays is a type of "lab on a chip" used to automatically collect data on large amounts of biological material. Machine learning can help in the analysis of this data, and has been applied to the identification of the expression pattern, the classification and the induction of the genetic network. This technology is especially useful to control the gene expression within a genome, helping to diagnose different types of cancer based on which genes are expressed. One of the main problems in this field is identifying which genes are expressed based on in the data collected. Furthermore, due to the large number of genes in which data is collected by microarrays, there is a large amount of irrelevant data to the expressed genetic identification task, further complicating this problem. Machine learning presents a possible solution to this problem, as various classification methods can be used to perform this identification.

Hospitals and healthcare providers are widely using machine learning and artificial intelligence to improve patient satisfaction, offer personalized treatments, make accurate predictions, and improve quality of life. It is also being used to streamline clinical trials and help speed up the drug discovery and delivery process.

After having researched on biology and, specifically, the branches of genomics and proteomics, having learned a series of machine learning techniques, and having combined both “worlds” to solve real problems; a series of conclusions can be drawn about the work carried out.

As seen, machine learning can be used in many fields of science. Specifically, in the bioinformatics sector it is becoming increasingly important and new algorithms are being developed to improve and streamline biological studies, which without the help of current statistics would take quite more time and money. In medicine and biology, a lot of work is done with genes to detect or classify diseases. Machine learning techniques, together with the statistical software available today, constitute a fundamental piece to make these genetic studies a simpler and faster task.

Finally, and as pointed out previously, we can say that machine learning techniques represent a great advance in solving bioinformatics problems. That is why, new methods should continue to be studied and existing ones improved, with the objective of solving difficulties that go along with us in the science world.

Índice de contenidos

Índice de figuras

Índice de ecuaciones

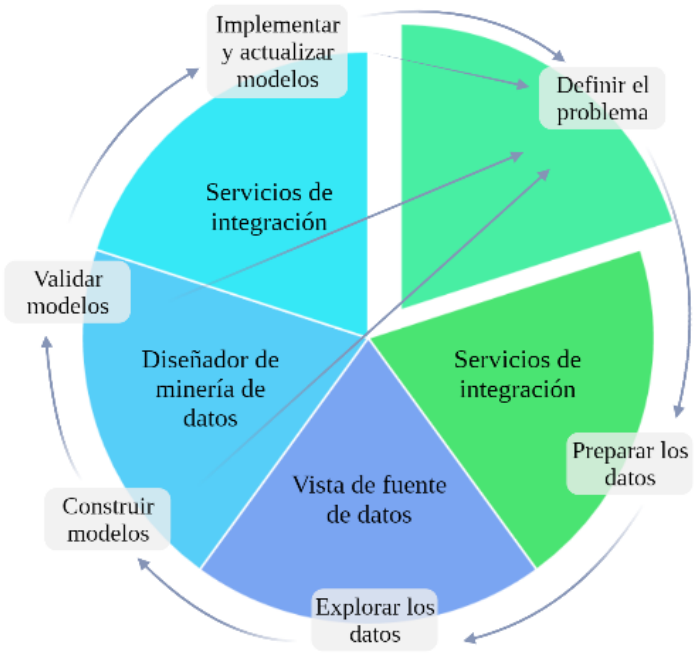
1. INTRODUCCIÓN

1.1. MINERÍA DE DATOS

Las empresas tienen una labor básica en su día a día, que es la elaboración de bases de datos. Éstas hacen posible elaborar estrategias para tener clientes nuevos o mantener a los que ya tenían de antes (Mitra & Acharya, 2005). Contamos con tal masificación de información, que en determinadas ocasiones resulta inviable mantenerla bien ordenada. Este problema se puede solventar fácilmente gracias a la *minería de datos*, que trata de encontrar patrones que se repiten a través de distintos algoritmos con base matemática o estadística (Olavide, 2006).

La minería de datos o *data mining* es un procedimiento que consiste en descubrir la información procesable de grandes conjuntos de datos. Emplea técnicas matemáticas y estadísticas para encontrar patrones y tendencias del conjunto de datos en cuestión, ya que dichos datos cuentan con relaciones demasiado enrevesadas o simplemente son muchos y no es posible recurrir a métodos tradicionales.

Figura 1. *Relaciones entre los pasos del proceso de minería de datos.*



Fuente: Elaboración propia con BioRender

A la hora de utilizar técnicas de minería de datos, es importante seguir una serie de pasos:

- Lo primero sería concretar el problema y buscar maneras de utilizar los datos para dar una respuesta para la cuestión.
- El segundo paso se basa en hacer una limpieza y criba de los datos con los que contamos.
- En tercer lugar habría que realizar una exploración de los datos mediante un análisis meramente descriptivo.
- Después, lo que hay que hacer es generar los modelos de minería de datos que se van a utilizar.
- El quinto paso consiste en inspeccionar y validar los modelos.

- Por último, hay que implementar los modelos que mejor se adapten a los datos y actualizarlos si fuera necesario.

1.2. BIOINFORMÁTICA

La *bioinformática* es una de las disciplinas de la ciencia que están cobrando más importancia actualmente; lo que se está comprobando precisamente este año, en el ámbito de la pandemia mundial en la que nos encontramos inmersos (López-Gartner et al., 2015). Esta disciplina tiene como objetivo explorar, desplegar y aplicar métodos informáticos y computacionales para que se puedan ordenar, analizar e interpretar datos biológicos. Se caracteriza por el hecho de estudiar información biológica con la ayuda de la ciencia de la computación y las matemáticas. Se trata de una disciplina totalmente nueva del campo de la biología, que fusiona *técnicas computacionales* con la comprensión de datos biológicos (Orozco & Jeferson, 2016).

El origen de este concepto se remonta a los años 60, de la mano de técnicas computacionales programadas para analizar la secuencia de las proteínas. Fue creciendo junto con el progreso de los conocimientos acerca de la *Biología Molecular*, el hallazgo del ADN y la evolución en métodos computacionales, entre otros avances. Actualmente, la percepción de la bioinformática es ligeramente distinta a la que se tenía entonces, ahora se trata de un área en pleno desarrollo ineludible para poder manejar la cantidad descomunal de datos generados por las llamadas tecnologías *"ómicas"* (proteómica, genómica…).

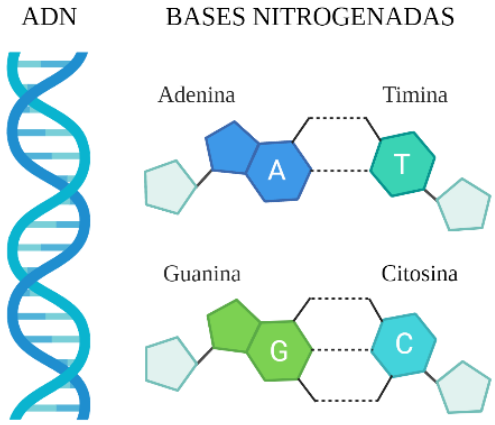
La bioinformática se basa en la *programación* en distintos lenguajes (R, Python, Java…) para diseñar algoritmos estadísticos que consigan extraer conocimiento a partir de los datos de los que tenemos constancia. Lo que se pretende conseguir con esta herramienta, en definitiva, es resolver grandes problemas vigentes hoy en día, como estudios de enfermedades tales como el cáncer, de una manera más dinámica y ordenada.

Los *microarrays* juegan un papel muy importante en el sector de la bioinformática, ya que constituyen un pilar fundamental a la hora de extraer datos biológicos y médicos en un primer momento. Por ello, se va a explicar más detenidamente cómo funciona la tecnología de estos pequeños chips (Mendoza Lombana, 2009).

1.2.1. MICROARRAYS DE ADN

El *Proyecto Genoma Humano* documentó nuestra secuencia genética y descubrió que el ADN de todos es idéntico al 99,9%. Para encontrar mutaciones, los investigadores usan *microarrays* para genotipar el ADN de los pacientes y determinar la secuencia exacta (A, T, C, o G). Para encontrar un tratamiento, los investigadores tienen que encontrar una causa, mutación o defecto en uno o más genes. Antes de la llegada de los *microarrays*, para resolver problemas complejos los científicos habrían tenido que realizar una investigación académica previa para enlaces genéticos con otros comportamientos parecidos a los que se pretendiera estudiar. Los investigadores usan los *microarrays* para escanear el conjunto del genoma y buscar parecidos genéticos entre un grupo de personas que comparten el problema. Todos los *microarrays* de *GeneChip (Affymetrix)* toman ventaja de la atracción química natural entre moléculas de ADN. Hay 4 moléculas o bases en cada cadena de ADN: A, G, T, C. C se empareja con G y A lo hace con T. Cuando una hebra de ADN encaja con otra, se dice que son complementarias.

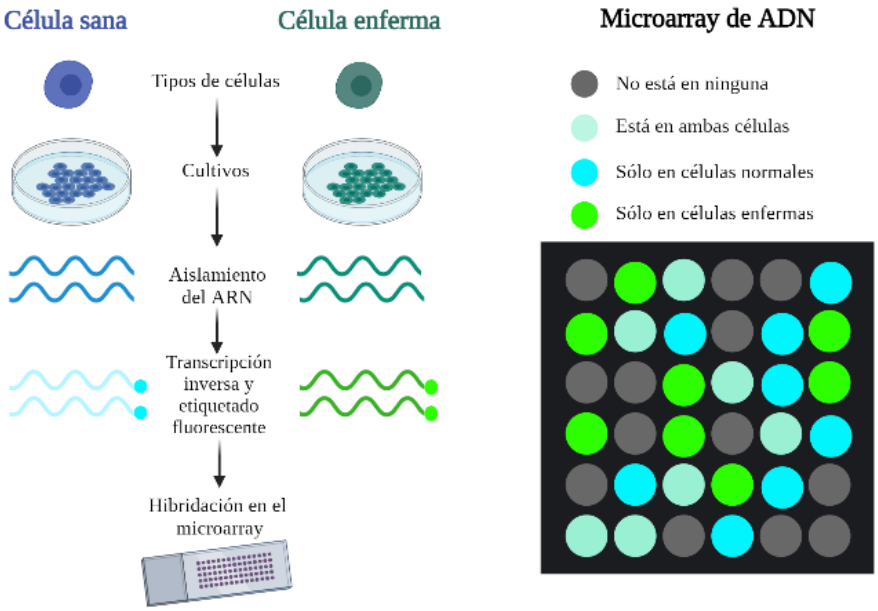
Figura 2. *Cadena de ADN con sus bases nitrogenadas.*



Fuente: Elaboración propia con BioRender

La superficie de un *array* de *Affymetrix* es como un tablero de ajedrez gigante. Cada cuadrado tiene un único tipo de hebra de ADN, llamado "*probe*". No hay sólo una hebra en cada cuadrado, sino millones de copias idénticas del mismo "*probe*". Ahora que tenemos un *probe* diseñado para medir ARN expresado, tenemos que extraer el ARN de una muestra biológica (sangre o saliva). Se hacen millones de copias de ese ARN para detectarlo mejor en el *array*. Mientras, unas moléculas de una sustancia química llamada biotina se unen a cada hebra. Estas moléculas de biotina actuarán como un pegamento molecular para las moléculas fluorescentes. Cuando los investigadores escaneen el *array* con un láser, las moléculas fluorescentes brillarán, mostrando dónde se ha pegado la muestra de ARN a las sondas (*probes*) de ADN en el *array*. Toda la muestra de ARN preparado se lava durante 14 o 16 horas. Enjuagamos el *array* para que los ARN desparejados sean eliminados. En un gen altamente expresado, la mayoría de moléculas de ARN se pegarán a la sonda y brillará bastante. El siguiente paso para los investigadores es usar técnicas adicionales mostrando las proteínas creadas por la función en cuestión.

Figura 3. *Microarray de ADN.*



Fuente: Elaboración propia con BioRender

1.3. INTELIGENCIA ARTIFICIAL

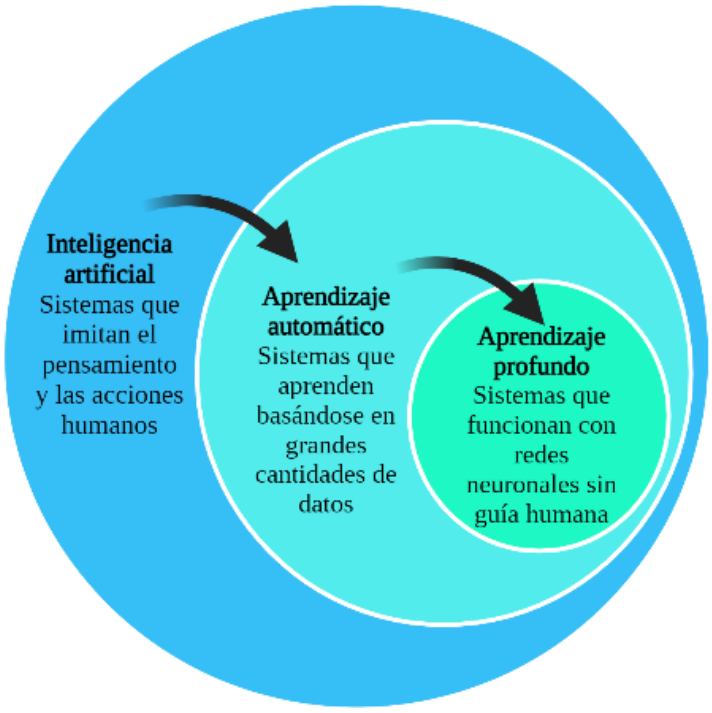
La *inteligencia artificial (IA)* es la capacidad de crear y programar máquinas con algoritmos matemáticos para que actúen como si se tratase de una mente humana y, con ello, sean capaces de aprender y tomar decisiones. Incluso pueden ser más eficaces que los humanos, puesto que son capaces de analizar muchos datos al mismo tiempo sin parar y cometen menos errores (Rouhiainen, 2018). El hecho de que los ordenadores aprendan y tomen sus propias decisiones hace que el sector de la inteligencia artificial esté creciendo exponencialmente.

Es un término que se definió en los 50; sin embargo, se ha dotado de gran importancia y conocimiento en los últimos años. Al principio se buscaba la solución de problemas y métodos simbólicos, lo que fue abriendo las puertas hacia la automatización y el razonamiento de los ordenadores para la toma de decisiones y búsquedas inteligentes que ayuden a mejorar las capacidades de las personas.

Quizás la idea que se tiene de la inteligencia artificial sea equivocada y ciertamente estremecedora. Esto es porque mucha gente piensa que únicamente consiste en crear robots que realicen las tareas manuales que antes hacían los humanos y por ello, nos van a arrebatar los trabajos o se van a volver en nuestra contra. En cambio, va mucho más allá y no se trata de automatización de robots basada únicamente en el hardware, sino que continúa siendo muy importante la investigación humana para configurar lo sistemas y no se puede prescindir así como así de la mente de las personas para realizar determinados trabajos.

En general, la inteligencia artificial no actúa por sí sola, sino que se aplica a productos y herramientas ya existentes con el fin de mejorarlos añadiendo un punto de inteligencia que no tenían antes. Esto se consigue a través de algoritmos de aprendizaje automático que se emplean para clasificar, recomendar productos…

Figura 4. *Inteligencia Artificial, Machine Learning y Deep Learning.*



Fuente: Elaboración propia con BioRender

La inteligencia artificial es un campo que contiene el área del aprendizaje automático, y este, a su vez, incluye el aprendizaje profundo (Cajamarca, 2019). Una máquina inteligente es un agente que entiende el ambiente que le rodea y realiza operaciones que maximicen sus probabilidades de éxito en un propósito determinado.

1.4. APRENDIZAJE AUTOMÁTICO

El *aprendizaje automático (AA)* o *machine learning* constituye uno de los sectores más destacables de la inteligencia artificial. Se puede definir como el área de la informática en la que los dispositivos electrónicos son capaces de aprender sin que ese sea su cometido inicial. Tenemos que agradecer al aprendizaje automático que muchos aparatos ofrecen un trato personalizado a cada usuario según la manera en que se han empleado, ya que obtienen experiencia y conocimientos a partir de ello (Su, 2005).

El *machine learning* emplea diferentes tipos de técnicas y algoritmos basados en las matemáticas y en la estadística con el propósito de detectar patrones en los datos con los que se está tratando. Un ejemplo concreto podría ser el del correo electrónico. Esta aplicación es capaz de distinguir los mensajes que no interesan de los que sí, agrupando aquellos menos importantes en la carpeta que conocemos con el nombre de *spam*. Esto se consigue a base de ver qué correos son los que abrimos y cuáles dejamos sin leer. Por lo tanto, la aplicación está fijándose en su entorno para elaborar patrones con los datos que tiene y así poder tomar la decisión de en qué carpeta introducir cada correo.

Los datos a partir de los cuales se construyen los modelos se denominan *datos de entrenamiento*, ya que el modelo entrena y aprende de ellos. Un modelo es una expresión matemática que describe la manera en que se relacionan unos atributos o características. Los métodos empleados pueden mostrar directamente el modelo que se pretende conseguir o, en su defecto, pueden encontrar datos parecidos o hallar patrones en dichos datos (Jimena Martínez, 2017).

El aprendizaje automático se puede dividir en tres clases (aprendizaje supervisado, aprendizaje no supervisado y aprendizaje por refuerzo) (Rouhiainen, 2018), de las que se hablará más adelante en el apartado Tipos de aprendizaje automático.

1.5. APRENDIZAJE PROFUNDO

El *deep learning* es un área de la inteligencia artificial, y a su vez, del aprendizaje automático, que se está volviendo cada vez más trascendente. Consiste en utilizar redes neuronales con el fin de facilitar a las máquinas tareas tales como el reconocimiento de voz o el procesamiento del lenguaje natural. El *aprendizaje profundo (AP)* ha contribuido a lograr avances en sectores tan dispares como la percepción de objetos y la traducción automática (Banafa, 2016).

Al contrario que los algoritmos tradicionales de aprendizaje automático, que tienen capacidad limitada de aprendizaje, los sistemas de aprendizaje profundo tienen un mejor rendimiento, ya que acceden a volúmenes más grandes de datos.

Las redes de aprendizaje profundo detectan estructuras complicadas en los datos que tratan.

Vamos a poner un ejemplo de un modelo de aprendizaje automático que se llama *"redes neuronales convolucionales"*. Se puede entrenar como millones de imágenes, como por ejemplo, que contengan gatos. Este tipo de red suele aprender de los píxeles que contienen las imágenes que adquiere. Clasifica conjuntos de píxeles que personaliza las características de un gato, como las patas, las orejas o la boca. Estas características en forma de píxeles asegurarían que hay un gato en la imagen.

En numerosas labores, la productividad de los sistemas de aprendizaje profundo domina considerablemente sobre la de los sistemas de aprendizaje automático. Esto no quiere decir que diseñar técnicas de aprendizaje profundo sea fácil; de hecho, para que el modelo sea útil se tienen que ajustar muchísimos parámetros.

El aprendizaje profundo ha tomado parte de una manera esencial en el hallazgo de exoplanetas, fármacos, en el diagnóstico de enfermedades… La escala de los retos a los que se enfrenta nuestra sociedad hoy en día requiere de un escalón más de inteligencia, que es más factible ahora que contamos con el *deep learning*.

1.6. BIG DATA

Hoy en día es fácil que todo el mundo haya escuchado alguna vez el término *Big Data*. Sin embargo, pocos son los que realmente entienden en qué consiste dicho avance tecnológico. El hecho de llegar y almacenar cantidades ingentes de información ha existido desde hace bastante tiempo. A pesar de ello, hasta los años 2000 no se acuñó el término de Big Data. Se trata de una de las principales áreas de ocupación para aquellos que trabajen en el mundo de las Tecnologías de la Información y la Comunicación (TIC) (Puyol Moreno, 2014). El *Big Data* abarca todos los sectores que podamos imaginar. Consiste en manejar bases de datos con volúmenes enormes y dificultad alta de análisis y comprensión.

Los datos han adquirido tal grado de valor, que la sociedad en la que vivimos será dirigida por ellos en algún momento. Esta es una de las razones por las que el *Big Data* está cobrando tanta importancia últimamente. Otra de las razones es que gran parte de los datos almacenados no se pueden procesar con sistemas tradicionales de computación, sino que requieren mecanismos más complejos. El hecho de tener la capacidad y los conocimientos necesarios para manejar grandes cantidades de datos correctamente supone un privilegio para las empresas en la sociedad de hoy, ya que les permiten tomar mejores decisiones y las hacen más competentes con respecto a otras que no son capaces de ello (Monleon-Getino, 2015).

Algunos autores entienden el concepto de *Big Data* como las llamadas *tres Vs*, refiriéndose al gran volumen de información que se tiene, a la inmensa variedad de datos que se pueden representar de formas muy diferentes (móviles, ordenadores, vídeo, GPS, coches, veletas, termómetros…) y a la velocidad de respuesta requerida para conseguir la información adecuada en el momento exacto (Puyol Moreno, 2014).

Muchas tecnologías que ayudan a estructurar, guardar y analizar los datos en profundidad giran en torno al concepto de *Big Data*.

Estas grandes cantidades de datos nos hacen entender lo que hacemos, lo que somos y lo que compramos; esto es así porque son capaces de predecir eventos futuros, como una catástrofe natural o una crisis económica.

La emergencia de *Big Data* supone buena oportunidad para conseguir ventajas en el ámbito comercial.

La desventaja del *Big Data* hasta hace unos años era que necesitaba una gran inversión de dinero para conseguir la capacidad computacional requerida. Este hándicap lo ha resuelto el *"Cloud Computing"*, que consiste en permitir el acceso a datos de todo el mundo a cualquiera con un método de pago. El mayor desafío para la inversión en *Big Data* es contar con personal cualificado para llevar a cabo estos trabajos y transformar los datos en decisiones y estrategias. Desde que apareció este término hasta hoy, su empleo se ha ido abaratando hasta el punto de permitir que el análisis de los datos no suponga un reto financiero difícil de superar, gracias a ciertos recursos computacionales.

1.7. ESTRUCTURA DEL TRABAJO

Una vez que ya se han introducido los conceptos y áreas básicas del aprendizaje automático, necesarios para entender el resto de apartados del trabajo, se puede proceder a explicar brevemente en qué consisten los apartados siguientes.

Lo primero que se ha de explicar son los Objetivos que se pretenden conseguir al elaborar este proyecto.

Después, conviene comentar los diferentes tipos de aprendizaje automático que existen (aprendizaje supervisado, no supervisado, semi-supervisado y aprendizaje por refuerzo), en qué consisten, cómo funcionan, por qué son importantes y en qué se diferencian unos de otros. Aquí no se va a hacer mucho hincapié en el mecanismo de los algoritmos de *machine learning*, ya que es un tema que se tratará con posterioridad. Esto se verá en el apartado de Tipos de aprendizaje automático.

Lo siguiente que se va a explicar son algunas de las Técnicas de aprendizaje automático más importantes hoy en día. Algunas que cabría destacar pueden ser los Árboles de decisión, los Algoritmos genéticos, las Reglas de asociación, las Redes neuronales artificiales, etc. En este apartado ya si se profundiza en el funcionamiento de los algoritmos, los modelos en los que se basan, las ecuaciones que utilizan y, en algunos casos, el código que se podría aplicar en programas como R o Python (caracterizados por su gran impacto en la rama de la estadística en los últimos años).

Teniendo siempre en cuenta los algoritmos explicados previamente, en el siguiente apartado (Aplicaciones del aprendizaje automático en la Bioinformática) se quiere dar a entender lo importante que resulta en aprendizaje automático en el campo de la bioinformática. En este momento se pretende explicar dónde se muestra significativo el aprendizaje automático dentro del mundo de la biología, la medicina…

Por último, tras haber trabajado bastante sobre los algoritmos de *machine learning* y sus aplicaciones en el área de la bioinformática, quedaría exponer una serie de conclusiones que se irán extrayendo en base a los conocimientos adquiridos a lo largo del presente trabajo.

2. OBJETIVOS

Lo primero que se pretende en este trabajo es comprender algunos conceptos básicos relacionados con el tema a tratar y desarrollarlos para que posteriormente resulte más sencillo aprender técnicas de aprendizaje automático. Algunos de estos conceptos clave son la minería de datos, la bioinformática, la inteligencia artificial o el aprendizaje profundo, entre otros.

Aunque, sobre todo, se busca llevar a cabo una investigación acerca de los distintos métodos de aprendizaje automático existentes a día de hoy. Para ello, se va a realizar una búsqueda bibliográfica de las técnicas que resultan más influyentes en los procesos de extracción de conocimiento.

También se quiere dar a entender lo importantes que resultan los avances en el sector de la inteligencia artificial y del aprendizaje automático para las diferentes ramas de la bioinformática, con el objetivo de aprender a obtener e interpretar información relevante de datos biológicos. La idea principal de este objetivo es centrarnos en la selección de genes en datos de *microarrays* de ADN, a través de un ejemplo práctico.

3. TIPOS DE APRENDIZAJE AUTOMÁTICO

Existen múltiples maneras de dividir las técnicas de aprendizaje automático según diferentes criterios. En este caso se ha elegido una clasificación en función de si el aprendizaje es supervisado o no y en qué medida lo es. Con ello obtenemos cuatro tipos de aprendizaje automático: supervisado, no supervisado, semi-supervisado y por refuerzo.

3.1. APRENDIZAJE SUPERVISADO

En las técnicas de *aprendizaje supervisado* se emplean etiquetas para señalar las diferentes clases que forman los datos. Este modelo de aprendizaje consiste en preparar un sistema para conseguir información que posibilite la clasificación de las muestras en base a sus categorías a posteriori. Las máquinas de vector soporte (SVM), las redes neuronales o el algoritmo de k-vecinos más próximos son algunas de las técnicas que son abordadas por este tipo de aprendizaje automático.

Estas técnicas se han empleado exitosamente en la identificación de patrones sobre los datos de *microarrays* de expresión génica, aunque seguramente el estudio más habitual dentro de este campo es el de *"expresión diferencial"*. Este análisis está basado en elegir aquellos genes que poseen una expresión significativamente grande o pequeña, entre dos categorías definidas con anterioridad (habitualmente se trata de una muestra de pacientes frente a otra de controles).

Hoy en día se conocen distintos algoritmos de expresión diferencial, siendo uno de los más nombrados SAM (*Significance Analysis of Microarrays*), que funciona con al paquete *siggenes* de R. Esta función utiliza un test para cada gen de la matriz de expresión, dándole un valor de *R-fold* y un *p-valor*. Después se hace una corrección del *p-valor*, que suele ser a través del método FDR (*False Discovery Rate*). Éste no es el único método que se emplea en Bioinformática, también sirven el de *Bonferroni* o el de *Hochberg*. Hay que decidir qué punto de corte escoger sobre el p-valor corregido (habitualmente entre 0,01 y 0,05), consiguiendo después un conjunto de genes que resultan estadísticamente significativos en cuanto a su expresión génica. Esto quiere decir que el grupo de genes que obtenemos posee una expresión diferente en cada una de las categorías de las que se parte inicialmente (Alberto Risueño Pérez, 2012).

El aprendizaje automático enfocado desde el punto de vista supervisado tiene el propósito de establecer una función que prediga el valor de un objeto de entrada correcto tras haber aprendido de los datos que se usaron como entrenamiento. La salida de la función puede ser un número —como en el caso de la regresión— o una etiqueta de clase —si nos encontramos en un problema de clasificación— Partiendo de los datos de entrada, debemos ser capaces de generalizar determinadas situaciones, que no han sido vistas en el aprendizaje (Jimena Martínez, 2017).

Los datos en el aprendizaje supervisado tienen atributos extras que se tratan de predecir (Grado et al., 2018).

El aprendizaje supervisado se divide a su vez en dos tipos, según los datos de entrada, de salida y lo que se pretenda conseguir:

- Problemas de *clasificación* (identificación de dígitos, diagnósticos o detección de identidad).
- Problemas de *regresión* (predicciones meteorológicas, de expectativa de vida, de crecimiento…).

3.2. APRENDIZAJE NO SUPERVISADO

El *aprendizaje no supervisado* intenta ofrecer información teniendo en cuenta los datos de muchas muestras sin que se les haya asignado una categoría o etiqueta antes. Este tipo de aprendizaje automático consiste en hallar parecidos y diferencias entre las muestras, por lo que es crucial medir la distancia entre ellas.

Este aprendizaje cuenta con diferentes beneficios en el campo de la biomedicina; por ejemplo, agrupando pacientes con perfiles parecidos, basándose en un principio en una población que sea supuestamente homogénea.

Asimismo se ha empleado exitosamente para descubrir relaciones entre genes basándose en sus perfiles de expresión génica, desvelando la existencia de grupos que se relacionan con funciones biológicas distintas.

Los métodos de agrupamiento jerárquico constituyen una rama de las técnicas no supervisadas que simbolizan las diferentes muestras, de tal forma que las muestras que se encuentran más cerca en una estructura conocida como dendrograma se parecen más entre sí. Estas técnicas se usan bastante para obtener una representación de similitudes y diferencias que hay entre las muestras del estudio en cuestión. Además, se pueden compaginar dos dendrogramas dando lugar a un mapa bidimensional que recibe el nombre de mapa de calor o *heatmap*. Esta nueva estructura mejora el reconocimiento visual de las distintas muestras gracias a la diferencia de colores que se emplean. Igualmente hay bastantes más métodos que separen las muestras en grupos (por ejemplo, *k-medias*) y otros de agrupamiento difuso.

En esta perspectiva ajustamos el modelo a las observaciones. Difiere del aprendizaje supervisado, principalmente, en que tan sólo necesita instancias y los datos no llevan etiquetas de ningún tipo (Jimena Martínez, 2017).

3.3. APRENDIZAJE SEMI-SUPERVISADO

También existe un tipo de aprendizaje automático, que combina los dos que se han explicado anteriormente y recibe el nombre de *aprendizaje semi-supervisado*. Un ejemplo común sería utilizar un método de agrupamiento de variables no supervisado (*clustering* jerárquico) en una matriz que contenga sólo variables significativas previamente seleccionadas mediante una técnica de aprendizaje supervisado. Esta clase de aproximación reduce el tipo de variables sólo a las que son significativas, apoyándose en las categorías conocidas *a priori*, y logra que la técnica de agrupamiento no supervisado clasifique las muestras (paciente-control) correctamente y haga posible inspeccionar con más exactitud el agrupamiento de las variables (de los genes si estamos tratando un problema de expresión génica).

3.4. APRENDIZAJE POR REFUERZO

El *aprendizaje automático por refuerzo* resulta atrayente dentro del mundo del aprendizaje automático. Por una parte, emplea un sistema de *feedback* y progreso que tiene una cierta similitud con el aprendizaje supervisado con descenso de gradiente. Por otra parte, no se suelen usar conjuntos de datos para solventar problemas de aprendizaje automático por refuerzo. Trata de combatir los problemas en un campo más amplio de la inteligencia artificial.

Vamos a poner un ejemplo. Imagínate que estás en la situación de jugar tu primera partida de chinchón y no sabes cuáles son las normas ni el funcionamiento del juego. La carta que tienes levantada para coger es el as de oros. Tienes dos opciones: coger el as de oros y cambiarla por una de tus cartas o

levantar una del montón y ver si te sirve. Decides levantar una carta del montón y, entonces, el siguiente jugador coge el as de oros y en la siguiente ronda cierra la partida y pierdes. No entiendes qué ha pasado, pero sí que el as de oros es una buena carta que la próxima vez cogerías antes que otras. Aquí se estaría aplicando aprendizaje automático por refuerzo, aprendiendo a escoger lo correcto en base a los resultados de situaciones anteriores parecidas.

Este tipo de aprendizaje automático funciona cuando no se sabe si una decisión es buena o mala antes de tomarla. Lo que sí se puede hacer es aprender del resultado después de tomar la decisión para escoger la opción correcta en ocasiones futuras.

Un concepto que habría que conocer en el área del aprendizaje automático por refuerzo sería *"agente"*, que es la capacidad de actuar en el propio entorno. Los agentes escogen una tarea que afecta al entorno, examinando el *"estado del medio ambiente"*, que tiene la información requerida para escoger una opción. La composición de las elecciones que realiza el agente denota su comportamiento. En definitiva, se trata de aprender una conducta óptima según experiencias pasadas.

Esto se consigue por medio de una puntuación que recibe cada acción. Como es de esperar, una positiva indica una buena decisión y una negativa denota una mala elección. Puede haber muchas opciones buenas, malas, o simplemente tener opciones sin puntuación. La meta final es elegir la mejor opción basándonos en la puntuación.

Tenemos tres motivos por los que se deberían considerar los resultados en vez de las acciones. En primer lugar, se tiene más información del resultado que del valor de las acciones. Realmente, lo que intentamos aprender es el valor de las acciones, ya que si lo conociésemos no haría falta recurrir al aprendizaje automático por refuerzo. En segundo lugar, el valor de determinadas acciones no tiene por qué ser siempre el mismo si el estado no cambia porque el azar también juega su papel. Por último, el hecho de premiar acciones en vez de resultados puede llevar a equívoco ("Aprendizaje automático por refuerzo," 2019).

4. TÉCNICAS DE APRENDIZAJE AUTOMÁTICO

4.1. ÁRBOLES DE DECISIÓN

Se trata de una técnica empleada desde hace bastante tiempo; sin embargo, se ha dado a conocer con más intensidad en los últimos años. Probablemente este hecho se deba a que ahora se pueden crear conjuntos de árboles distintos, en vez de uno solo. Se pueden utilizar para razonar la manera en que se ha llegado a la predicción en cuestión.

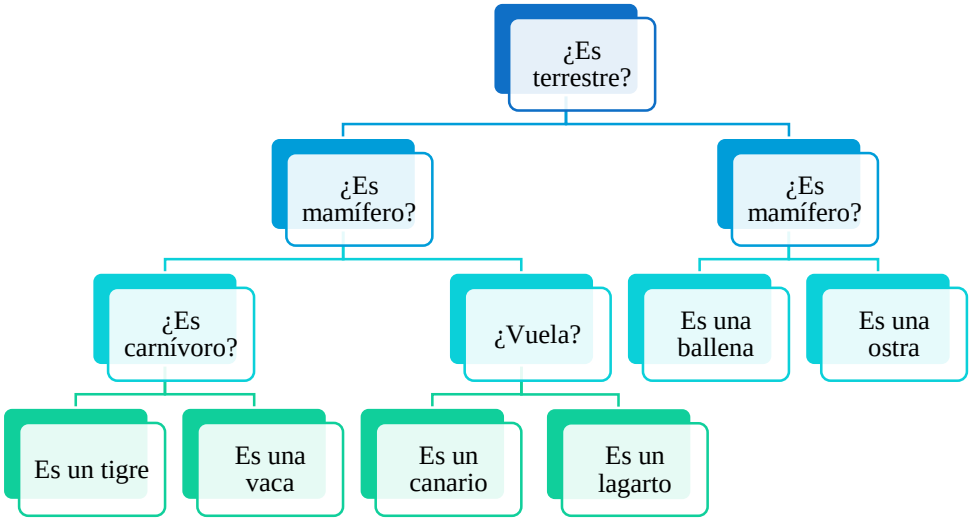
Un ejemplo de su uso puede ser en esquemas de clasificación para predecir mapas de contactos inter-residuales de proteínas. También se pueden emplear para clasificar láminas de borde en láminas centrales en un conjunto de proteínas. Esta perspectiva resulta eficaz porque hace posible localizar las zonas de interacción proteína – proteína (Santiesteban-Toca, Casañola-Martin, & Aguilar-Ruiz, 2014).

Un árbol se puede considerar un tipo de grafo particular, una disposición de datos constituida por un conjunto de nodos y vértices ordenados jerárquicamente. Tenemos distintos tipos de nodos: raíz, internos y terminales. Todos los nodos tienen un solo vértice entrante, excepto el nodo raíz. Los árboles carecen de bucles (no como los grafos).

Un árbol de decisión determina una propiedad que no se conoce a partir de una muestra, mediante sucesivas preguntas acerca de las propiedades que sí sabemos. Las respuestas que se van ofreciendo marcan la siguiente pregunta a realizar. Estas preguntas escalonadas se representan junto con sus respuestas a través de un grafo en forma de árbol (López González Tutorizado, Ángel, Bayarri, & Azpitarte, 2016).

A continuación, se muestra un ejemplo de árbol de decisión sencillo. Para saber de qué animal se trata lo primero que se pregunta es si es terrestre. La siguiente pregunta es si es mamífero, independientemente de que sea terrestre o no. En el caso de no ser terrestre sólo tenemos dos nodos terminales (ballena y ostra); mientras que si lo es contamos con cuatro nodos terminales, dependiendo el animal de si es carnívoro o vuela.

Figura 5. *Ejemplo de árbol de decisión*



Fuente: Elaboración propia

4.1.1. ÁRBOLES DE SUPERVIVENCIA

Los árboles de supervivencia son una clase de árboles de clasificación y regresión preparados para tratar con datos censurados. La idea elemental de un árbol es ir separando los datos en base a una norma particular y los elementos que se parecen se deberán situar en el mismo nodo.

La diferencia más importante entre árbol de supervivencia y árbol de decisión es el criterio de división. En un árbol de decisión se hacen particiones recursivamente al decretar un umbral para cada característica, pero no se tienen en cuenta las interacciones entre las características ni la información censurada del modelo. Existen dos estratos en los que se pueden dividir los criterios de división para la supervivencia:

- Maximizar la heterogeneidad entre nodos, en cuyo caso se minimiza la función de pérdida usando el criterio de homogeneidad en el nodo.
- Minimizar la homogeneidad en los nodos, donde se utilizan estadísticas de prueba de rango logarítmico para medidas de heterogeneidad entre nodos.

La selección del árbol final constituye una parte importante de la creación de un árbol de supervivencia. Esto se puede ir haciendo hacia delante o hacia atrás. No obstante, un conjunto de árboles es capaz de eludir el problema de la selección final de árboles y con un rendimiento mayor que si sólo hubiese un árbol (Bellot, 2020).

4.1.2. BOSQUES DE DECISIÓN

Un bosque de decisión es un conjunto de árboles de decisión que comienzan aleatoriamente. La característica más importante de esta técnica es que los árboles que constituyen los bosques son distintos entre sí de forma aleatoria. Esto hace que haya una de-correlación entre las predicciones de cada árbol y hace más general y robusto el sistema.

Cada árbol se entrena al margen de los demás y, en la medida de lo posible, en paralelo. En la fase de test, cada muestra se introduce al mismo tiempo en todos los árboles hasta llegar a los nodos terminales.

Ecuación 1. *Combinación de las predicciones en un bosque de decisión mediante promediado*

$$p(c|\boldsymbol{V}) = \frac{1}{T}\sum_{t=1}^{T} p_t(c|\boldsymbol{V})$$

La ***Ecuación 1*** se emplea para combinar las predicciones de los árboles. $p_t(c|\boldsymbol{V})$ representa la distribución de probabilidad que se ha conseguido a posteriori por el t-ésimo árbol. Con la ***Ecuación 2*** se pretende conseguir lo mismo, pero multiplicando las salidas y usando los datos normalizados (López González Tutorizado et al., 2016).

Ecuación 2*. Combinación de las predicciones en un bosque de decisión mediante productorio*

$$p(c|\boldsymbol{V}) = \frac{1}{Z}\prod_{t=1}^{T} p_t(c|\boldsymbol{V})$$

4.2. REGLAS DE ASOCIACIÓN

Las reglas de asociación son algoritmos que tienen como objetivo descubrir conexiones en un conjunto de ítems o atributos propensos a ocurrir a la vez. La palabra *transacción* en este contexto se refiere a los grupos de eventos asociados entre sí, como la cesta de la compre o las páginas web que visita una persona.

Una transacción puede estar constituida por uno o varios ítems. Cada uno de los posibles conjuntos de ítems es un *ítemset*.

Un ejemplo básico de regla de asociación podría ser $X \Rightarrow Y$, que traducimos en "X implica Y". En este caso X e Y son los *ítemsets*. La parte izquierda de la regla de asociación se llama antecedente y la derecha, consecuente (Rodrigo, 2018).

4.2.1. *A PRIORI*

Se trata del algoritmo más frecuente de reglas de asociación, encuentra las relaciones más comunes entre los ítems, realiza iteraciones hasta que las relaciones no cuentan con el soporte mínimo, es una técnica robusta a la par que sencilla y proporciona una salida bastante intuitiva.

No requiere asignar los atributos de cada lado de la regla, sino que se forman automáticamente. Hay distintas formas del algoritmo para poder manejar diferentes tipos de datos, necesita que se especifique el soporte mínimo y el máximo de reglas con las que se va a contar.

Consiste en buscar conjuntos que tengan cardinalidad de 1 a k y emplea dichos conjuntos para crear las reglas de asociación, mediante iteraciones. "*A priori*" se asume que todo subconjunto de un conjunto frecuente ha de ser frecuente también.

En base al principio de poda, si alguno de los conjuntos es infrecuente, no hay que crear superconjuntos.

4.3. ALGORITMOS GENÉTICOS

El principal problema al que se enfrentan los algoritmos genéticos es localizar la mejor hipótesis de un espacio, es decir, la que optimiza una medida numérica que predefinimos para el problema (adaptación *fitness* de la hipótesis). Si el objetivo del aprendizaje es aproximar una función desconocida contando con un conjunto para entrenar entradas y salidas, la adaptación sería la precisión de la hipótesis sobre el conjunto de entrenamiento, es decir, el porcentaje de éxitos al predecir la salida. A veces, el aprendizaje se puede tomar como un juego, en cuyo caso la adaptación se puede medir como porcentaje de partidas ganadas.

A pesar de que los detalles de implementación cambian en los distintos algoritmos genéticos, la estructura suele ser la misma. En dicha estructura el algoritmo funciona con iteraciones, renovando un conjunto de hipótesis denominado población. La población se evalúa en base a una función de adaptación en cada una de las iteraciones. Se origina una población nueva que proviene de la selección de los sujetos que más se han adaptado. Otros individuos se eligen para establecer un nuevo conjunto sobre el que aplicar operaciones genéticas, tales como el cruce o la mutación.

Las entradas del algoritmo que se muestra en la ***Figura 6*** contienen una función de adaptación con el fin de evaluar los aspirantes a hipótesis, un umbral que defina el grado de aceptación, el tamaño de la

población y los parámetros requeridos para definir cómo progresa la población, es decir, la fracción de la población que se remplazará en cada una de las generaciones y la tasa de mutación.

Cada iteración origina una nueva generación de hipótesis, basada en la población actual de hipótesis. Lo primero es seleccionar un número de hipótesis en la población $((1 - r) \times p)$ para introducirlo en la prole. La probabilidad de una hipótesis $h_i \in pob$ de ser elegida viene dada por la siguiente ecuación:

Ecuación 3. *Probabilidad de una hipótesis de ser seleccionada*

$$\Pr(h_i) = \frac{adaptación(h_i)}{\sum_{j=1}^{p} adaptación(h_j)}$$

Cuando ya tenemos elegidos a los miembros de la población que van a formar parte de la prole, los demás componentes de la misma se seleccionan con el operador de cruce. Dicho operador selecciona $(r \times p)/2$ pares de hipótesis de la población y da lugar a dos descendientes para cada par, combinando trozos de los dos padres. Los padres se escogen a través de probabilidades con la $\Pr(h_i)$. Ahora la prole tiene un tamaño p, por lo que falta elegir m elementos de la población para emplear la operación mutación. Estos individuos se escogen al azar y las transformaciones que se realizan sobre ellos son aleatorias. En ocasiones, se evita que las mejores hipótesis puedan mutar, practicando de esta manera un cierto elitismo.

Este algoritmo genético efectúa una búsqueda por barrido (*beam*), paralela y aleatoria de hipótesis que cuentan con un buen desempeño conforme a la función de adaptación (A. G. Hern, 2004).

Figura 6. *Prototipo de un algoritmo genético*

```
        umbral: adaptación suficiente para parar;
        p: número de hipótesis en la población;
        s: fracción de la población a subsistir;
        m: tasa de mutación;
    static:    pob: población;
        pobAux: caché para recalcular población;
    output: maxh: la hipótesis con mejor adaptación;
    pob ← generadorHipótesis(p);
    foreach h in pob do
        adaptación(h);
    endforeach
    while    maxAdaptación(pob) < umbral do
        pobAux ← selección ((1 − r) × p, pob);
        pobAux ← pobAux u cruce ((r × p) / 2, pob);
        pobAux ← mutar (m, pobAux);
        pob ← pobAux;
        foreach h in pob do
            adaptación(h);
        endforeach
    endwhile
    return maxh ← maxAdaptación(pob);
endfun
```

4.3.1. REPRESENTACIÓN DE HIPÓTESIS

Las hipótesis de un algoritmo genético se representan con cadenas de bits, de manera que los operadores de cruce y mutación las puedan manejar sencillamente.

Vamos a poner un ejemplo. Tomamos el atributo cielo con sus tres posibles valores (soleado, nublado, lluvia). Una forma de codificar una restricción sobre este atributo sería emplear una cadena de 3 bits, donde cada posición de la cadena concuerda con un valor concreto. Cuando se pone un 1 en una determinada posición, significa que el cielo toma el valor de esa posición. Por ejemplo, la cadena 011 representa la restricción $cielo = nublado \cup cielo = lluvia$. Ahora vamos a tener en cuenta otro atributo (viento) que puede ser fuerte o débil.

$$(cielo = nublado \cup lluvia) \cap (viento = fuerte)$$

La condición anterior se puede representar a través de una cadena de bits de la siguiente manera:

cielo	*viento*
011	*10*

También se pueden representar post-condiciones, es decir, los que pasaría en un determinado ámbito si se dan unas condiciones definidas previamente. Por ejemplo, si $viento = fuerte \rightarrow jugartenis = si$ sería de la siguiente manera con bits:

cielo	*viento*	*jugartenis*
111	*10*	*10*

Cuando diseñamos un código binario, conviene tener en cuenta que todas las cadenas de bits sintácticamente válidas representan una hipótesis bien definida. Por ejemplo, la cadena 111 10 11, no impone restricciones sobre jugartenis. Para evitarlo, se podría emplear un solo bit (A. G. Hern, 2004).

4.3.2. OPERADORES GENÉTICOS

Los operadores genéticos unen y mutan a los individuos seleccionados de la población para formar parte de la prole (hijos). En la siguiente imagen podemos ver algunos de los operadores genéticos más comunes. Se trata de versiones idealizadas de las operaciones genéticas de la evolución biológica.

Figura 7. *Algunos operadores genéticos*

	padres	máscara	hijos
Cruce en un solo punto:	11101001000 00001010101	11111000000 →	11101010101 00001001000
Cruce en dos puntos:	11101001000 00001010101	00111110000 →	11001011000 00101000101
Cruce uniforme:	11101001000 00001010101	11111000000 →	11101010101 00001001000
Punto de mutación:	11101001000	→	11101011000

Fuente: Elaboración propia

Del bit que se encuentra en la posición *i* de un padre se obtiene el bit de la posición *i* de cada hijo. La *máscara* es la cadena que marca cuál es el padre que va a aportar el bit *i*. Cuando la máscara lleva un 1 en la posición *i*, el hijo toma el bit de esa posición de un padre; por otra parte, si lleva un 0, el hijo toma el bit del otro padre. Por tanto, habría dos posibilidades de descendencia, dependiendo de qué padre escogemos como 1 y cuál como 0.

El operador de cruce en un solo punto da lugar a dos descendientes, partiendo de dos cadenas de bits. Se cogen bits de cada uno de los padres y se transmiten a los hijos. La máscara en este caso está formada por *n* posiciones de unos y el resto ceros hasta concluir la cadena. La cadena se completa cuando los hijos tienen el mismo número de bits que los padres. El punto de cruce, *n*, se escoge de forma aleatoria.

El cruce en dos puntos requiere dos puntos de cruce en la máscara: un n_0 para ceros y un n_1 para unos. Después habría que terminar la cadena con ceros otra vez.

En el cruce uniforme, la máscara se crea de una forma totalmente aleatoria y los bits son generados independientemente de los otros (A. G. Hern, 2004).

4.3.3. FUNCIÓN DE ADAPTACIÓN Y SELECCIÓN

La función de adaptación, también conocida como *fitness*, determina la pauta a seguir para ordenar las hipótesis potenciales y para elegirlas mediante probabilidad, con el objetivo de incorporarlas en la próxima generación de la población. Siendo el cometido aprender reglas de clasificación, la función de adaptación cuenta con un elemento que evalúa la precisión de las hipótesis sobre los datos de entrenamiento. Esta función tiene que medir el desempeño general de las reglas, además de la precisión individual.

Existe un método de selección proporcional a la adaptación o selección de ruleta en el que la probabilidad de que una hipótesis se elija viene dada por el radio de su adaptación y la adaptación del resto de individuos de la población en cuestión.

Otro método de selección conocido es la selección por torneo, en el que las hipótesis se escogen de la población por azar. Por esto, la hipótesis más adaptada se elige con una probabilidad p y con una probabilidad $(1 - p)$ se escoge la menos adaptada. Por consiguiente, se consigue una población son diversidad (A. G. Hern, 2004).

4.4. REDES NEURONALES ARTIFICIALES

Desde principios del siglo XX se ha comenzado a llevar a cabo modelos computacionales que han tratado de imitar la conducta del cerebro humano. A pesar de que se han planteado muchos, todos emplean una estructura de red en la que los nodos o neuronas son procesos numéricos que comprenden estados de otros nodos en función de sus uniones (Bellot, 2020).

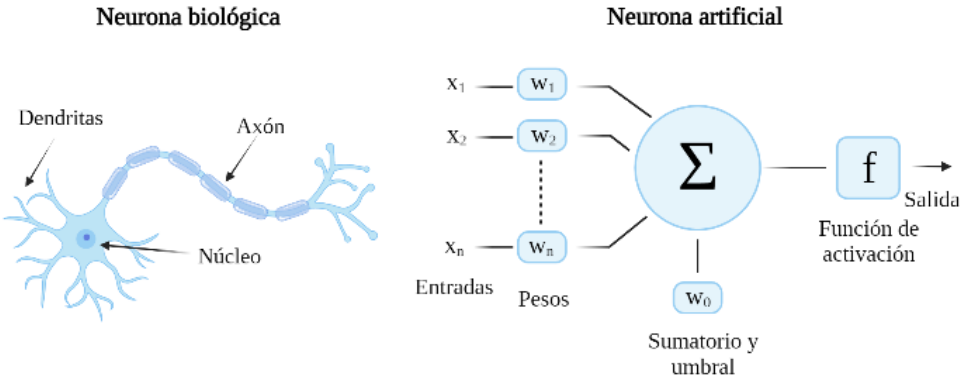
Las Redes Neuronales Artificiales (RNA) se han dado a conocer por lo sencillo que es usarlas e implementarlas y la capacidad de aproximar cualquier función matemática. Se pueden emplear para obtener patrones y encontrar tramas de difícil apreciación para las personas.

Se trata de conjuntos de elementos de cálculo simples, normalmente adaptativos, conectados internamente en masa de forma paralela y que cuentan con una disposición jerárquica que les hace posible la interacción con algún sistema de la misma forma en que lo lleva a cabo el sistema nervioso humano.

Las RNA consisten en imitar la actividad que realizan las redes neuronales biológicas. Las neuronas del cerebro humano están constituidas por dendritas, núcleo (o soma) y axón. La función de las dendritas es recibir los impulsos nerviosos emitidos por otras neuronas. Dichos impulsos son procesados por el núcleo y transmitidos por el axón hacia neuronas adyacentes.

En la ***Figura 8*** podemos ver las estructuras de una neurona humana y de una neurona artificial y así compararlas de una forma más visual.

Figura 8. *Partes de una neurona humana vs partes de una neurona artificial*



Fuente: Elaboración propia con BioRender

El impulso nervioso de las neuronas artificiales viene dado por la suma de las entradas multiplicadas por sus pesos. El valor que se obtiene es procesado dentro de la célula a través de una función de activación que devuelve un valor que se manda como salida de la neurona.

Al igual que las neuronas cerebrales, las artificiales también se pueden disponer de tal forma que se obtenga una red neuronal con neuronas interconectadas entre sí en diferentes niveles o capas.

La primera capa se conoce como *capa de entrada*, la última como *capa de salida* y las que se encuentran entre ambas se denominan *capas ocultas* (porque no sabemos los valores de entrada y de salida de las mismas).

El concepto de *Deep Learning*, del que se ha hablado en el apartado Aprendizaje profundo, surge del hecho de emplear muchas capas ocultas en las redes neuronales.

Para entrenar una red hay que ajustar los pesos de las entradas de todas sus neuronas para que las respuestas se ajusten lo máximo posible a los datos. Según se van ajustando los pesos, el error se reduce. Para que una red neuronal pueda identificar y generalizar es imprescindible contar con bastantes imágenes con las que entrenar e incluyendo la mayor variabilidad que se pueda (Olivera, 2019).

Puede haber una función que actúe como umbral en las conexiones y en cada neurona, de manera que la señal debe ser mayor que un límite determinado para poder transmitirse a otra neurona.

Para que una red neuronal aprenda y funcione exitosamente, se requiere un número muy elevado de iteraciones.

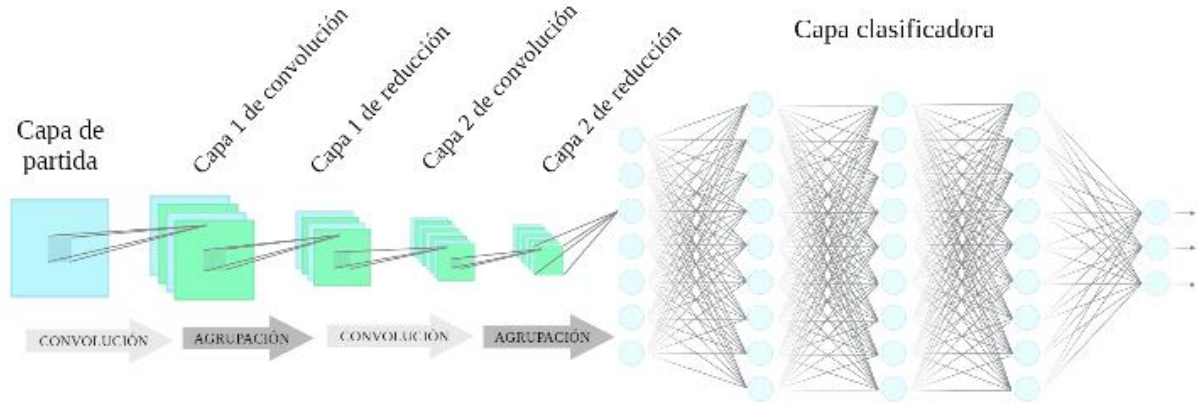
Se trata de una técnica que se usa para resolver problemas difíciles para la programación ordinaria, por lo que se puede considerar un método útil para solucionar el problema de detección automática de órganos en medicina (López González Tutorizado et al., 2016).

4.4.1. Redes Neuronales Convolucionales

Las redes neuronales convolucionales son un tipo de red neuronal artificial en el que las neuronas se relacionan con campos receptivos de una forma parecida a lo que ocurre con las neuronas en la corteza visual primaria de un cerebro humano. Se trata de una variante del *perceptrón multicapa*. Como se emplea en matrices de dos dimensiones (o tres si hablamos de detección de órganos), son bastante útiles para problemas de visión artificial.

Están formadas por muchas capas de filtros convolucionales de las dimensiones que sea. Tras cada capa se incluye una función para llevar a cabo un mapeo causal no lineal, normalmente *Max-pooling*. Tratándose de redes de clasificación, al inicio se tiene la fase de extracción de características, formada por neuronas convolucionales. Al finalizar la red hay neuronas de perceptrón simples para hacer la clasificación final sobre las características que se han extraído. La fase en la que se extraen las características se parece al proceso de estimulación de las células de la corteza visual. Dicha fase está compuesta por capas alternas de neuronas convolucionales y neuronas de reducción de muestreo. A medida que avanza el progreso de los datos en esta fase, se reduce la dimensionalidad, de modo que las neuronas más lejanas pierden sensibilidad a las perturbaciones en los datos de entrada.

Figura 9. *Esquema de una Red Neuronal Convolucional*



Fuente: Elaboración propia con BioRender

Las redes neuronales convolucionales se consideran una de las mejores formas de confrontar la detección automática de órganos, debido a que se han estado obteniendo unos resultados muy positivos en los últimos años de estudio sobre este tema (López González Tutorizado et al., 2016).

4.5. MÁQUINAS DE VECTORES DE SOPORTE

Las *máquinas de vectores de soporte* (SVM) constituyen un algoritmo de aprendizaje automático de tipo supervisado. Su principal objetivo es localizar un plano que divida los grupos dentro de los datos lo mejor posible. La selección del plano maximiza el margen entre los puntos que se encuentran más próximos entre sí en el plano; dichos puntos son los vectores de soporte (Abril, 2018).

Se trata de un algoritmo de clasificación que se ha usado bastante desde que se creó en los 90 con el fin de resolver diferentes problemas de clasificación. En el área de la farmacogenómica se ha empleado principalmente para predecir desenlaces partiendo de variables *ómicas* y clínicas (A. Hern, 2020).

Una máquina de vector de soporte elabora un hiperplano óptimo como superficie de decisión, para que el margen entre los dos grupos se amplíe al máximo. Los vectores de soporte se refieren a un subconjunto de las observaciones de entrenamiento que se emplean como soporte para la localización óptima de la superficie de decisión (Bellot, 2020).

El entrenamiento se puede dividir en dos fases:

- Transformar los datos de entrada en un espacio dimensional, procedimiento conocido como truco *kernel*.
- Enfrentarse a un problema de optimización cuadrática ajustado al hiperplano óptimo con la meta de clasificar las características que se han transformado en dos clases.

4.5.1. CLASIFICACIÓN DE PROBLEMAS LINEALMENTE SEPARABLES

Se trata de un método que consiste en generar un hiperplano en un espacio de p dimensiones (haciendo alusión a los p predictores que hay en un conjunto de datos) a fin de dividir un conjunto de n observaciones maximizando el margen entre los dos puntos más próximos al hiperplano.

Se tiene un conjunto de datos con unos vectores $X_1, X_2, \dots, X_n$ con p elementos o predictores y unas variables respuesta $y_1, y_2, \dots, y_n \in \{-1, 1\}$ para cada observación. -1 es una de las clases y 1 la restante. El hiperplano ha de separar los puntos acorde a la clase y_i a la que corresponden, como se observa en las siguientes ecuaciones:

Ecuación 4. *Hiperplano para y=1*

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} > 0, \qquad si\ y_i = 1$$

Ecuación 5. *Hiperplano para y=-1*

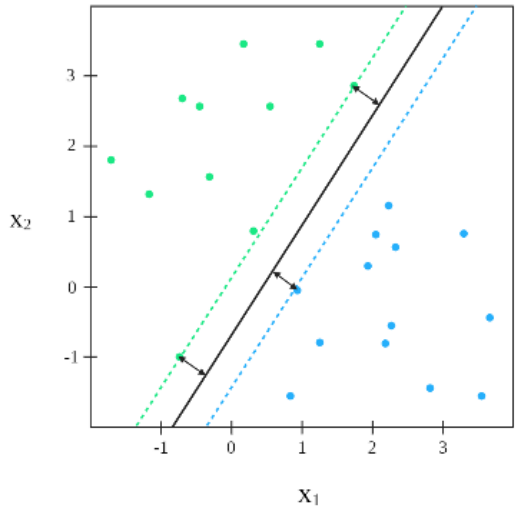
$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} < 0, \qquad si\ y_i = -1$$

Ecuación 6. *Hiperplano para* $1 < i < n$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) > 0\ \forall\ i\ \in \{1, \dots, n\}$$

Los coeficientes de la ecuación son los β_i y x_{ij} hace referencia al componente *j* del vector de predictores X_i, lo que significa que aquello que se encuentra por encima o por debajo del hiperplano tomará los valores 1 y -1, respectivamente. En la ***Figura 10*** los puntos verdes representan los vectores cuyo resultado de la ecuación anterior es menor que 0 y los azules se corresponden con los resultados mayores que 0 (Santos, 2019).

Figura 10. *Hiperplano en un espacio de 2 dimensiones*



Fuente: Elaboración propia

Se puede definir margen como la distancia entre el punto más cercano al hiperplano y el propio hiperplano. Esta idea resulta útil a la hora de elegir el mejor hiperplano para dividir los datos, ya que si un conjunto de datos se puede separar mediante un hiperplano, hay un número infinito de hiperplanos capaces de dividirlo. De esta manera, el hiperplano de máximo margen es el que divide los puntos de manera que el valor del margen M sea máximo. La ecuación del hiperplano anterior puede generalizarse de siguiente manera:

Ecuación 7. *Hiperplano general*

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M \; \forall \; i \; \in \{1, \ldots, n\}$$

4.5.2. CLASIFICACIÓN DE PROBLEMAS LINEALMENTE NO SEPARABLES

En algunos casos no se puede elaborar un hiperplano que divida rigurosamente el conjunto de observaciones según los espacios que haya. Más concretamente, la ecuación $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip})$ carece de solución para valores de M mayores o iguales que 0. Con el objetivo de solucionar este problema, se emplean clasificadores que posibiliten la clasificación equivocada de determinados puntos para mejorar la clasificación de los puntos y obtener robustez en cuanto a pequeñas alteraciones en observaciones individuales; es decir, para reducir la varianza (V, 2013). Estos clasificadores reciben el nombre de margen suave o clasificador de soporte vectorial y hacen que algunos de los puntos se sitúen en el lado erróneo del margen o del hiperplano. El problema de maximización de M para los clasificadores de soporte vectorial se cambia de la siguiente forma:

Ecuación 8. *Hiperplano para problemas linealmente no separables*

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M(1 - \varepsilon_i) \; \forall \; i \; \in \{1, \ldots, n\}$$

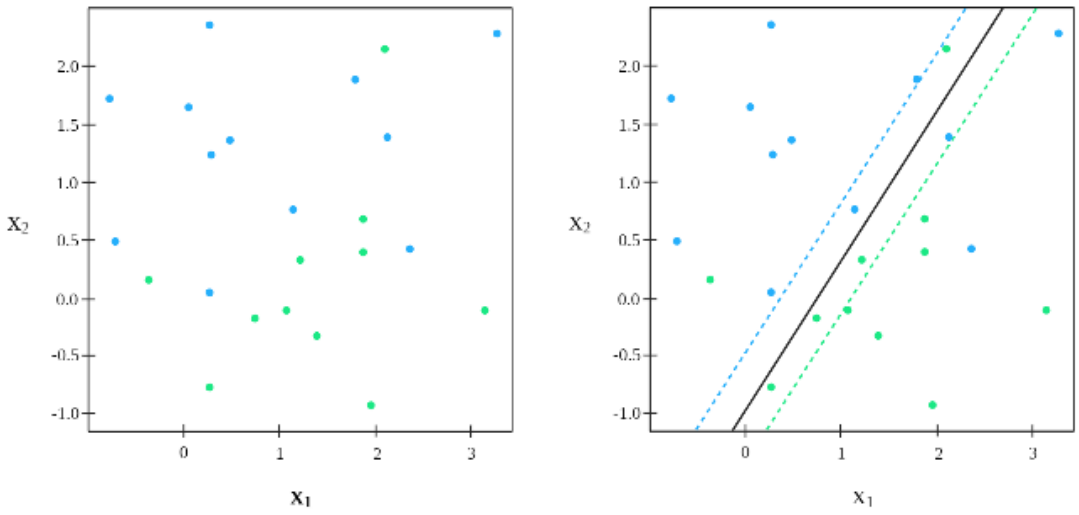
Ecuación 9. *Error en problemas linealmente no separables*

$$\varepsilon_i \geq 0 \sum_{i=1}^{n} \varepsilon_i \leq C$$

El error ε_i señala la ubicación del punto i en relación con el margen. Si $\varepsilon_i = 0$, i se encuentra en la orilla correcta del margen, si $0 < \varepsilon_i < 1$, i está en el lado incorrecto y si $\varepsilon_i > 1$, i se halla en el lado incorrecto del hiperplano. C es un valor no negativo para ajustar, que acota los valores de ε_i. Si $C = 0$, todos los $\varepsilon_{i,\dots,n} = 0$, si bien, si $C > 0$, quiere decir que no puede haber más de C observaciones en el margen incorrecto del hiperplano (G, James, Witten D, Hastie T, 2013).

Cabría destacar que no todos los puntos afectan a la posición del hiperplano y la elaboración del margen. De hecho, la ubicación del hiperplano sólo depende de los puntos del margen y de si violan el margen o el hiperplano. Al cambiar la posición de los puntos que no se encuentran en las posiciones mencionadas anteriormente a una posición no mencionada, la localización del hiperplano no va a variar. Por esta razón, los puntos de posiciones mencionadas se llaman vectores de soporte (V, 2013).

Figura 11. *Representación de un conjunto de puntos que no pueden separarse*



Fuente: Elaboración propia

4.6. ALGORITMOS DE AGRUPAMIENTO O *CLUSTERING*

Se trata de un conjunto de técnicas que se pueden clasificar dentro del aprendizaje no supervisado, ya que el agrupamiento no está basado en grupos definidos previamente (Villazana, Arteaga, Seijas, & Rodriguez, 2012).

Dividir un conjunto de datos en grupos es una tarea importante para conocer el comportamiento de la población en cuestión. Para llevar a cabo esta labor se emplean los llamados algoritmos de agrupamiento (Pascual, Pla, & Sánchez, 2007).

"El Clustering es una tarea que consiste en agrupar un conjunto de objetos (no etiquetados) en subconjuntos de objetos llamados Clusters. Cada Cluster está formado por una colección de objetos

que son similares (o se consideran similares) entre sí, pero que son distintos respecto a los objetos de otros Clusters." (Moya, 2016).

Gracias al uso del *clustering* el sistema es capaz de analizar los datos, llevar a cabo la tarea que se le encomiende y localizar errores en el funcionamiento. En este caso, su labor es segmentar datos en grupos de dimensiones parecidas basándose en características para favorecer el proceso.

Los métodos de *clustering* resultan complicados porque en función de los criterios que se usan para diseñar el *cluster*, será efectivo o no para lograr la meta que tenemos. Lo primero que hay que hacer es definir el número de *clusters* que tenemos que elaborar.

Después habría que determinar las formas y similitudes que deben tener los miembros de cada grupo y fijar el centro desde donde se empezará el agrupamiento, así como especificar un margen de error.

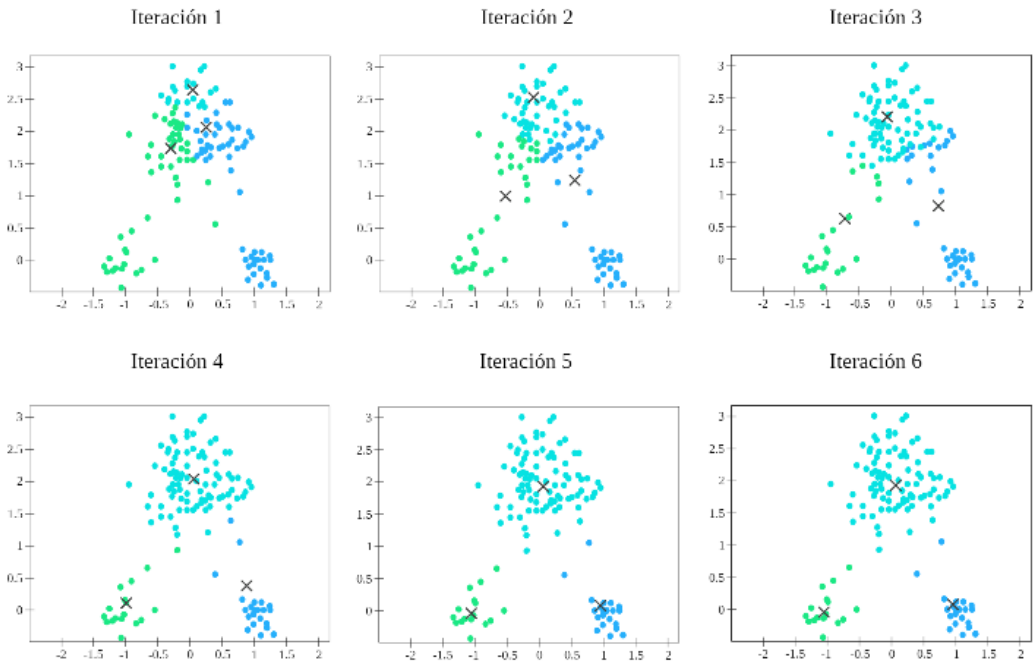
Lo ideal es crear un bucle con muchas iteraciones para tener todas las combinaciones de errores que pueda haber en el modelo.

4.6.1. ALGORITMO DE *K-MEANS*

Es un algoritmo que divide los N individuos en K particiones (K es un valor arbitrario). Cada objeto se posicionará en el *cluster* que tenga la media más parecida a la suya. Se establecen K centros al azar y, posteriormente, se sitúan los objetos en el centro más próximo a ellos. Después, se vuelve a calcular el centro como la media de los puntos, se asignan otra vez los objetos al centro más cercano, y así sucesivamente hasta obtener convergencia (Wikipedia, 2020).

Conviene repetirlo varias veces con valores diferentes, ya que depende de la primera asignación y puede proporcionar un resultado u otro. Para resolver este problema se ha diseñado un algoritmo llamado *K-means++*, que elige unos centros mejores (Arthur & Vassilvitskii, 2006).

Figura 12. *Ejemplo de algoritmo de k-medias*



Fuente: Elaboración propia

Vamos a analizar la ***Figura 12***. En la primera iteración se ven tres cruces que representan los tres centros iniciales. Se asignan los puntos al centro más próximo. Tras este paso, se calcula la media de los puntos y se actualiza el centroide. Se continúa con estos pasos hasta que los puntos no varíen.

La desventaja más notable de este algoritmo es que hay que fijar el número de clusters desde el principio y los resultados cambian en función de este número (Sanz, 2016).

4.6.2. ALGORITMO DE *K-NEAREST NEIGHBOURS*

Se trata de un algoritmo que se basa en instancia (no aprende un modelo, sino las instancias de entrenamiento que se emplean como base de conocimiento para la predicción) y de tipo supervisado dentro del aprendizaje automático. Se emplea para clasificar muestras (variables discretas) o para predecir (variables continuas). Principalmente se usa para clasificar valores buscando los puntos más parecidos que se han aprendido en la fase de entrenamiento y suponiendo nuevos puntos en base a esa clasificación (Bagnato, 2018).

En el caso de *K-Nearest Neighbours* la *K* representa el número de puntos vecinos que se consideran en las cercanías para clasificar los n grupos que ya se conocen previamente, ya que es de tipo supervisado (Merkle, 2020).

Tiene una ventaja clara y es que es fácil de implementar. Sin embargo, funciona mejor en conjuntos de datos pequeños y con pocas *features*, ya que emplea todos los datos para entrenar cada uno de los puntos y requiere mucha memoria.

Lo primero que hay que hacer es calcular la distancia entre el ítem que queremos clasificar y el resto de ítems del conjunto de datos de entrenamiento. Después se escogen los K individuos más cercanos y, finalmente, se realizará una votación de mayoría entre los *K* puntos: los de una clase que dominen decidirán su clasificación final (Isabel, Herrera, Hoyo, & Martinez, 2020).

La manera más común de medir la distancia entre los puntos es empleando la conocida distancia Euclídea:

Ecuación 10. *Distancia Euclídea*

$$d_E(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

P_1 y P_2 son los puntos cuya distancia se quiere medir y (x_1, y_1) y (x_2, y_2) son las coordenadas cartesianas.

4.6.3. *CLUSTERING* JERÁRQUICO

Este algoritmo asocia los datos en función de la distancia entre los *clusters* y buscando que los datos de un *cluster* sean los que más se parecen (González, 2019).

Si representamos gráficamente, los elementos estarían anidados jerárquicamente en forma de árbol. Este tipo de agrupamientos se denominan dendrogramas y resultan de gran interés para distintos campos de aplicación. Los árboles jerárquicos proporcionan una vista de los datos en distintos niveles de abstracción. La consistencia de las soluciones de agrupamiento en diferentes niveles de detalle permite

extraer particiones planas de diferentes niveles de detalle durante el análisis de datos, lo que las hace ideales para la exploración y visualización interactivas (Duda, R.O., Hart, P.E., and Stork, 2001).

Los algoritmos jerárquicos constituyen una estructura donde los individuos se asocian en subconjuntos cada vez más grandes hasta que todos ellos son del mismo conjunto. Así, se muestran las relaciones de cercanía que hay entre los elementos.

Los procedimientos básicos pueden ser de dos tipos: *aglomerativos* (se crean clusters individuales, es decir, con un elemento cada uno, y en cada una de las iteraciones se juntan los dos *clusters* más cercanos, y así hasta que solamente quede un *cluster*) y *divisivos* (partiendo de un *cluster* que abarca todos los demás, en cada iteración se elige un cluster y se divide y el proceso termina cuando tengamos tantos clusters como individuos) (Wesley, 2006).

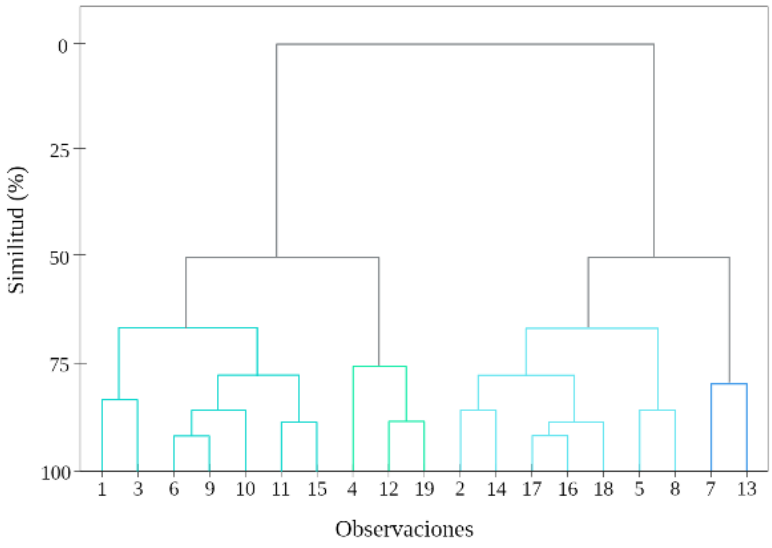
4.6.3.1. DENDROGRAMA

Un dendrograma es un diagrama de árbol que va enseñando los subconjuntos que se crean al formar conglomerados de individuos y su grado de parecido. Dicho grado se mide en el eje *Y* y los individuos en el eje *X* (Schutte, 2006).

La forma más común de representar un *clustering* jerárquico es a través de dendrogramas. Estos esquemas reflejan el orden en que se han asociado los *clusters* y el grado de cercanía que poseen. Los nodos hojas representan los individuos en solitario, el nodo raíz corresponde al *cluster* que abarca todos los demás y el resto de nodos son los *clusters* que se van creando (Wesley, 2006).

La clasificación final se lleva a cabo a partir de la poda del dendrograma. Esto significa que hay que trazar una línea a lo largo de la figura para determinar la clasificación final. Se pueden comparar distintas posibilidades para elegir la que tenga más sentido (Schutte, 2006).

Figura 13. *Dendrograma con distancia Euclídea*



Fuente: Elaboración propia

4.7. REDES BAYESIANAS

Las redes bayesianas modelan fenómenos a través de un conjunto de variables y las dependencias entre ellas. Con este modelo se puede realizar inferencia bayesiana (estimar la probabilidad de variables desconocidas según las variables conocidas). Estos modelos pueden tener diferentes aplicaciones en clasificación, predicción, diagnóstico… Asimismo, pueden proporcionar información útil sobre cómo son las relaciones entre las variables del dominio, que en ocasiones se interpretan como una relación de causa-efecto.

Al principio, estos modelos se crearon "a mano" y se fundamentaban en un conocimiento experto. En los últimos tiempos se han descubierto distintas técnicas para aprender basándonos en los datos.

Las redes bayesianas son representaciones gráficas de dependencias para razonamiento de probabilidad en las que los nodos son variables aleatorias y los arcos las relaciones de dependencia directa entre las variables (Sucar, 2006).

"Una Red Bayesiana es un modelo probabilístico que relaciona un conjunto de variables aleatorias mediante un grafo dirigido, son redes gráficas sin ciclos en el que se representan variables aleatorias y las relaciones de probabilidad que existan entre ellas que permiten conseguir soluciones a problemas de decisión en casos de incertidumbre." (Rivera Lozano, 2011).

Una red bayesiana consiste en representar dependencias para un razonamiento probabilístico. Los nodos se corresponden con las variables aleatorias y los arcos con las relaciones de dependencia directa entre variables (Tié, Félix, 2007).

Es una herramienta informática con la que se pueden crear distintos modelos dependiendo del diseñador y de las condiciones del comportamiento de las variables. Dicha herramienta permite que el proceso vaya hacia atrás (*backward*) y hacia adelante (*forward*) (María Teresa Ortiz, 2015).

Las redes bayesianas se pueden considerar una solución de independencia, ya que permiten encontrar relaciones de dependencia o independencia entre conjuntos de variables.

4.7.1. INFERENCIA BAYESIANA

Los modelos bayesianos simulan situaciones de incertidumbre si no se sabe si la hipótesis enunciada es verdadera o falsa en un determinado rango de variación.

Todos los modelos bayesianos tienen la asignación de probabilidad como medida de creencia de una hipótesis, por lo tanto, la inferencia es un procedimiento para reajustar las medidas de creencia al conocer axiomas nuevos.

La inferencia bayesiana emplea observaciones y evidencias para afirmar que una suposición sea verdadera. Se trata de observar la evidencia y calcular una estimación en función del grado de creencia que plantea la hipótesis. Por ello, si se cuenta con una cantidad grande de datos los resultados serán más satisfactorios.

Se usa con frecuencia para la toma de decisiones, principalmente cuando existen parámetros desconocidos (Tié, Félix, 2007).

Es capaz de percibir aquellas situaciones en que la dinámica del mercado tiene una estrategia firme de cambios, ya que se basa en un método que utiliza datos histórico. Esto ofrece un indicador bastante bueno sobre las condiciones futuras del mercado.

Aplicando los modelos de inferencia bayesiana se pueden reconocer diferentes tipos de patrones de transición. La inferencia sobre este tipo de redes hace posible su uso para desarrollar clasificadores. Para ello se necesita una red bayesiana cuyas variables se relacionen entre sí en el grafo. Se verificará la clasificación infiriendo la probabilidad posterior de cada valor de la clase sobre el grafo. Después se escoge el valor que haga máxima esta probabilidad (Fernández & Britos, 2004).

4.7.2. TIPOS DE REDES BAYESIANAS

Vamos a clasificar las redes bayesianas en función del tipo de variables utilizadas, que pueden ser continuas o discretas.

- Redes bayesianas continuas:

  Tienen infinitos valores posibles. Es difícil establecer exactamente las probabilidades condicionadas para cada uno de los valores de las variables, por lo que dichas probabilidades vienen dadas por una función de probabilidad.

  Casi todas las variables reales son de este tipo. *"Una red Bayesiana cuyas variables sean todas continuas y estén todas representadas mediante funciones normales lineales, tiene una distribución normal multivariada."* (Rivera Lozano, 2011). Estas variables hay que convertirlas en discretas por la gran cantidad de datos con la que se cuenta.

- Redes bayesianas dinámicas:

  Afectan a procesos que cuentan con una variable aleatoria en cada intervalo de tiempo. El proceso se puede entender como una serie de procesos en un momento del tiempo (Chow, 1968).

  Las probabilidades condicionales del presente modelo no varían con el tiempo, sino que se repiten las etapas y las relaciones entre las etapas.

4.7.3. APLICACIONES DE LAS REDES BAYESIANAS

Algunas de las áreas en las que se emplean las redes bayesianas son medicina, genética, procesos de producción, depuración de programas de inteligencia artificial, entre otras; todas ellas orientadas a la resolución de problemas y a la determinación de probabilidades que reduzcan el riesgo. En el estudio y tratamiento de datos, las redes bayesianas previenen el riesgo operacional; de tal manera que contribuye a tomar decisiones en situaciones de emergencia.

Las redes bayesianas constituyen una herramienta muy importante para la rama financiera por su aportación en las condiciones de probabilidad de inferencia.

También se emplean con el objetivo de representar conocimiento en técnicas de razonamiento, análisis de deudores del sistema financiero, predicción de ventas, explotación de información… Las redes bayesianas hacen posible aprender acerca de las relaciones de dependencia y combinar nuevos datos con conocimiento.

4.8. ANÁLISIS DE COMPONENTES PRINCIPALES

Probablemente, el *Análisis de Componentes Principales* (ACP) sea la técnica estadística multivariante más popular y antigua. Se utiliza en prácticamente todas las disciplinas científicas. Su origen se remonta a *Pearson* y *Cauchy* (*Pearson*, 1901), aunque fue *Hotelling* (*Hotelling*, 1933) quien lo formalizó y también quien acuñó el término de *componente principal*.

El ACP analiza datos representando observaciones descritas por diferentes variables dependientes, que en general están correlacionadas entre sí. Su objetivo es extraer información importante de los datos y expresarla como un conjunto nuevo de variables ortogonales que reciben el nombre de *componentes principales*. Además, representa el patrón de similitud de las observaciones y las variables mostrándolas como puntos en mapas (Saporta & Keita, 2009).

4.8.1. REQUISITOS PREVIOS

Hay que preprocesar los datos antes de analizarlos. Casi siempre, las columnas de la matriz X se centrarán para que la media de cada columna sea 0. Cuando las variables se miden en diferentes unidades, se suelen estandarizar a la norma unitaria.

La matriz X tiene los siguientes valores de descomposición (Hervé Abdi, 2007):

Ecuación 11. *Descomposición de la matriz X*

$$X = P\Delta Q^T$$

P es la matriz $I \times L$ de los vectores singulares de la izquierda, Q la matriz $J \times L$ de los vectores singulares de la derecha y Δ es la matriz diagonal de valores singulares. Hay que tener en cuenta que Δ^2 es igual a la matriz diagonal de los autovalores de $X^T X$ y XX^T.

La *inercia de una columna* se define como la suma de sus elementos al cuadrado y se calcula de la siguiente manera:

Ecuación 12. *Inercia de una columna*

$$\gamma_j^2 = \sum_i^I x_{i,j}^2$$

La suma de todos los γ_j^2 es la *inercia total* de los datos, que también equivale a la suma de los valores singulares al cuadrado.

El *centro de gravedad de las filas g* (también llamado centroide o baricentro) es el vector de las medias de cada columna de X.

La *distancia Euclídea* de la i-ésima observación del vector g se calcula a través de la siguiente ecuación:

Ecuación 13. *Distancia Euclídea en el vector g*

$$d_{i,g}^2 = \sum_j^J \left(x_{ij} - g_j\right)^2$$

La suma de todos los $d_{i,g}^2$ es igual a la *inercia total*.

4.8.2. OBJETIVOS DEL ANÁLISIS DE COMPONENTES PRINCIPALES

Los objetivos del Análisis de Componentes Principales son:

- Extraer la información más importante de los datos.
- Reducir el tamaño del conjunto de datos, manteniendo solo la información importante.
- Simplificar la descripción del conjunto de datos.
- Analizar la estructura de las observaciones y las variables.

Para alcanzar estas metas, el ACP calcula nuevas variables llamadas *componentes principales*, que se obtienen como combinaciones lineales de las variables originales. La primera componente principal ha de tener la mayor varianza posible. La segunda se calcula bajo la restricción de ortogonalidad con la primera componente y se calcula de la misma manera. Los valores de estas nuevas variables para las observaciones se llaman *puntuaciones factoriales* y se interpretan geométricamente como las *proyecciones* de las observaciones sobre las componentes principales.

4.8.2.1. ENCONTRAR LAS COMPONENTES

En el ACP, las componentes se obtienen de la *Descomposición en Valores Singulares* (DVS) de la matriz X. Específicamente, con $X = P\Delta Q^T$. La matriz de puntuaciones factoriales de $I \times L$, denotada por F, se obtiene:

Ecuación 14. *Matriz F de puntuaciones factoriales*

$$F = P\Delta$$

La matriz Q da los coeficientes de las combinaciones lineales usados para calcular las puntuaciones factoriales. Dicha matriz también se puede interpretar como una matriz de proyección, ya que multiplicar X por Q proporciona los valores de las proyecciones de las observaciones en las componentes principales. Esto se puede mostrar de la siguiente manera:

Ecuación 15. *Combinación de ecuaciones*

$$F = P\Delta = P\Delta Q^T Q = XQ$$

Las componentes también se pueden representar geométricamente por la rotación de los ejes originales. Por ejemplo, si X representa dos variables, la longitud de una palabra (Y) y el número de líneas de su definición en el diccionario (W), entonces el ACP representa los datos con dos factores ortogonales. En este contexto, la matriz X se puede interpretar como sigue:

Ecuación 16. *Producto de la matriz de puntuaciones factoriales*

$$X = FQ^T, \quad F^T F = \Delta^2 \quad y \quad Q^T Q = I$$

Sabemos que la matriz Q es una matriz de proyección que transforma la matriz de datos original en puntuaciones factoriales. Esta matriz también se puede emplear para calcular las puntuaciones factoriales para observaciones que no fueran incluidas en el ACP. Dichas observaciones se llaman *suplementarias* o *ilustrativas*. Las observaciones necesarias para calcular el ACP se llaman *activas*. Las puntuaciones factoriales para las observaciones suplementarias se obtienen primero posicionando estas observaciones en el espacio del ACP y entonces proyectándolas sobre las componentes principales. Un vector x_{sup}^T se proyecta así:

Ecuación 17. *Proyección del x_{sup}^T en el ACP*

$$f_{sup}^T = x_{sup}^T Q$$

Si los datos se has preprocesado (centrado o normalizado), las observaciones suplementarias deberían someterse al mismo preprocesamiento antes de calcular sus puntuaciones factoriales.

4.8.3. Interpretación del Análisis de Componentes Principales

4.8.3.1. La contribución de una observación a una componente

El autovalor asociado a una componente es igual a la suma de los cuadrados de las puntuaciones factoriales para esta componente. Por consiguiente, la importancia de una observación para una componente se puede obtener por la proporción de las puntuaciones factoriales al cuadrado de esta observación por el autovalor asociado con esta componente. Dicha proporción es la *contribución* de la observación a la componente, que se calcula así:

Ecuación 18. *Contribución de una observación i a la componente l*

$$ctr_{i,l} = \frac{f_{i,l}^2}{\sum_i f_{i,l}^2} = \frac{f_{i,l}^2}{\lambda_l}$$

El valor de una contribución se encuentra entre 0 y 1 y, para una componente dada, la suma de las contribuciones de todas las observaciones es igual a 1. Resulta útil basar la interpretación de una componente en las observaciones cuya contribución es mayor que el porcentaje de contribución.

Las puntuaciones factoriales de las observaciones suplementarias no se utilizan para calcular los autovalores. Por ello, sus contribuciones no se suelen calcular.

4.8.3.2. Coseno al cuadrado de una componente con una observación

El coseno al cuadrado muestra la importancia de una componente para una observación dada. Indica la contribución de una componente a la distancia al cuadrado de la observación al origen. Se corresponde al cuadrado del coseno del ángulo de un triángulo rectángulo formado por el origen, la observación y su proyección en la componente y se calcula de la siguiente forma:

Ecuación 19. *Cuadrado del coseno de un triángulo rectángulo*

$$cos_{i,l}^2 = \frac{f_{i,l}^2}{\sum_l f_{i,l}^2} = \frac{f_{i,l}^2}{d_{i,g}^2}$$

La distancia entre el origen y la observación se calcula como el sumatorio de las puntuaciones factoriales al cuadrado de esta observación. Las componentes con un valor alto del coseno, contribuyen en una proporción relativamente grande a la distancia total, por lo que dichas componentes resultan importantes para la observación.

4.8.3.3. CARGA: CORRELACIÓN DE UNA COMPONENTE Y UNA VARIABLE

La correlación entre una componente y una variable estima la información que comparten. En el marco de referencia del ACP, esta correlación se denomina *carga*. La suma de los coeficientes de correlación al cuadrado entre una variable y todas las componentes es igual a 1. Por tanto, las cargas al cuadrado son más fáciles de interpretar que las cargas sin elevar al cuadrado.

Las variables se pueden graficar como puntos en el espacio de componentes, empleando sus cargas como coordenadas. Esta representación difiere del gráfico de observaciones: Las observaciones se representan con sus proyecciones y las variables con sus correlaciones. Cuando los datos se representan perfectamente con solo dos componentes, la suma de los cuadrados de las cargas es 1, lo que implica que las cargas se situarán en el llamado *círculo de correlaciones*. Cuando se requieren más de dos componentes para representar los datos, las variables se localizarán dentro del círculo. Cuanto más cerca esté una variable del círculo de correlaciones, mejor se reconstruirá la variable de las dos primeras componentes. Y, cuanto más cerca esté una variable del centro del gráfico, menos importante será para las dos primeras componentes.

4.8.4. INFERENCIA ESTADÍSTICA: EVALUAR LA CALIDAD DEL MODELO

4.8.4.1. MODELO DE EFECTO FIJO

Los resultados del ACP hasta ahora corresponden a un modelo de efecto fijo. En este contexto, el ACP es descriptivo y la cantidad de varianza explicada de X por una componente indica su importancia.

Para un modelo de efecto fijo, la calidad del modelo de ACP usando las primeras M componentes se obtiene calculando la matriz estimada $\hat{X}^{[M]}$, que es la matriz X reconstruida con las primeras M componentes. La fórmula para esta estimación se obtiene combinando otras anteriores para finalmente tener que $X = FQ^T = XQQ^T$.

Entonces, la matriz $\hat{X}^{[M]}$ se construye usando $X = FQ^T = XQQ^T$, manteniendo solo las M primeras componentes:

Ecuación 20. *Matriz estimada*

$$\hat{X}^{[M]} = P^{[M]}\Delta^{[M]}Q^{[M]T} = F^{[M]}Q^{[M]T} = XQ^{[M]}Q^{[M]T}$$

$P^{[M]}, \Delta^{[M]}$ y $Q^{[M]}$ representan las matrices P, Δ y Q con solo sus M primeras componentes.

Para evaluar la calidad de la reconstrucción de X con M componentes, se evalúa la similitud entre X y $\hat{X}^{[M]}$. Se pueden usar varios coeficientes para esta tarea (H. Abdi, 2007). El coeficiente de correlación al cuadrado a veces se emplea como el coeficiente R_V (Dray, 2008). El coeficiente más conocido es la *suma de cuadrados residual* (SCE) y se calcula como sigue:

Ecuación 21. *Suma de cuadrados residual*

$$SCE_M = \left\| X - \hat{X}^{[M]} \right\|^2$$

Cuanto menor sea la suma de cuadrados residual, mejor será el modelo del ACP. En el caso del modelo de efecto fijo, cuanto más grande es M, mejor estimación de $\hat{X}^{[M]}$ se obtiene.

4.8.4.2. MODELO DE EFECTO ALEATORIO

En la mayoría de aplicaciones, el conjunto de observaciones representa una muestra de una población más grande. En este caso, la meta es estimar el valor de las observaciones nuevas de esta población. Esto corresponde a un modelo de efecto aleatorio. Para estimar la capacidad de generalización del modelo de ACP, no se pueden emplear procedimientos paramétricos estándar. Por consiguiente, la actuación del modelo de ACP se evalúa utilizando técnicas de remuestreo basadas en computación, como puede ser la validación cruzada (donde los datos se dividen en conjuntos de aprendizaje y conjuntos de prueba). Una técnica popular de validación cruzada es el *jackknife* (Oaks & Sage, 2010), donde cada observación se elimina del conjunto y las observaciones restantes constituyen el conjunto de aprendizaje.

4.8.4.3. ¿CUÁNTAS COMPONENTES HAY QUE CONSIDERAR?

A menudo, solo se necesita extraer la información importante de una matriz de datos. En este caso, el problema es averiguar cuántas componentes hay que tener en cuenta (Peres-Neto & Jackson, 2005). Lo primero es tener un gráfico con los autovalores de acuerdo a su tamaño ("*screeplot*") y ver si hay un punto en dicho gráfico, conocido como codo, en el que la pendiente pasa de ser empinada a plana y conservar solo aquellas componentes que se sitúan antes del codo (Cattell, 1966).

Otra forma de elegir el número de componentes es quedarnos sólo con las que tienen los autovalores más grandes que la media.

4.8.5. ROTACIÓN

Tras determinar el número de componentes y con el objetivo de facilitar la interpretación, el análisis implica una rotación de las componentes (Hervé Abdi, 2003). Los ejes nuevos siempre explicarán menos inercia que las componentes originales, ya que las rotaciones se llevan a cabo en un subespacio.

Los dos tipos de rotación más utilizados son la ortogonal y la oblicua, que se van a explicar a continuación.

4.8.5.1. ROTACIÓN ORTOGONAL

Una rotación ortogonal viene especificada por una matriz de rotación, llamada R, donde las filas representan los factores originales y las columnas, los rotados. En la intersección de la fila m y la columna n tenemos el coseno del ángulo entre el eje original y el nuevo: $r_{m,n} = \cos\theta_{m,n}$. Una matriz de rotación tiene la propiedad de ser ortonormal. La rotación *varimax*, desarrollada por *Kaiser* (Kaiser, 1958), es el método de rotación más conocido. Para *varimax*, una solución simple significa que cada componente tiene un número pequeño de cargas grandes y un número grande de cargas pequeñas. Esto simplifica la interpretación porque, después de una rotación *varimax*, cada variable original tiende a asociarse a una de las componentes y cada componente representa sólo un número pequeño de variables. Formalmente *varimax* busca factores tales que maximicen la varianza de las cargas al cuadrado:

Ecuación 22. *Maximización de la varianza de las cargas al cuadrado*

$$v = \sum (q_{j,l}^2 - q_l^{-2})^2$$

4.8.5.2. ROTACIÓN OBLICUA

Con rotación oblicua, los ejes nuevos son libres de tomar cualquier posición en el espacio de las componentes, pero el grado de correlación permitido entre los factores es pequeño porque dos componentes altamente correlacionados se interpretan mejor como un único factor. Por ello, este tipo de rotaciones relajan la restricción de ortogonalidad para hacer más sencilla la interpretación (Thurstone, 1956).

Dentro de las rotaciones oblicuas, la más rápida y simple, conceptualmente, es la *promax*. El primer paso es definir la matriz diana, casi siempre obtenida como el resultado de una rotación *varimax*. Después, hay que calcular un ajuste de mínimos cuadrados de la solución *varimax* para la matriz diana. Las rotaciones *promax* se interpretan mirando las correlaciones entre los ejes rotados y las variables originales.

5. APLICACIONES DEL APRENDIZAJE AUTOMÁTICO EN LA BIOINFORMÁTICA

Los métodos desarrollados en el apartado anterior (Técnicas de aprendizaje automático) son los más utilizados en el campo de la bioinformática, y más concretamente en el área de la genómica. La bioinformática es uno de los sectores de la ciencia que más recurre a las técnicas de aprendizaje automático para resolver sus problemas y descubrir nuevos avances.

El incremento exponencial de la cantidad de datos biológicos plantea dos problemas, que resulta interesante conocer:

- La extracción de información útil a partir de dichos datos.
- Constituye uno de los principales retos de la biología computacional, que necesita el desarrollo de herramientas y métodos capaces de transformar esos datos heterogéneos en conocimiento biológico sobre el mecanismo en cuestión. Estas herramientas deberían aportar conocimientos en forma de modelos comprobables. Gracias a esta abstracción se podrán obtener predicciones del sistema.

Existen varios dominios biológicos en los que aplicar técnicas de aprendizaje automático para extraer conocimiento, que se pueden clasificar de la siguiente manera:

- La *genómica* es uno de los dominios más importantes de la bioinformática. A partir de las secuencias del genoma, se puede extraer la ubicación y estructura de los genes (Mathé, Sagot, Schiex, & Rouzé, 2002). La identificación de elementos reguladores (Aerts, Loo, Moreau, & De Moor, 2004) y genes de ARN no codificantes (Carter, Dubchak, & Holbrook, 2001) también se están llevando a cabo desde el punto de vista computacional. Además, la información de la secuencia se emplea para la función genética y la predicción de la estructura secundaria del ARN.
- Las proteínas transforman la información que contienen los genes en vida. Juegan un papel muy importante en el proceso de la vida y su estructura tridimensional resulta clave en su funcionalidad. La principal aplicación de los métodos computacionales en *proteómica* es la predicción de la estructura de las proteínas. Las proteínas son macromoléculas muy complejas con miles de átomos y límites. Esto hace que la predicción de la estructura de la proteína constituya un problema combinatorio complicado, en el que se requieren técnicas de optimización. En este ámbito se emplean técnicas de aprendizaje automático para la predicción de la función de las proteínas.
- Los ensayos de *microarrays* son el dominio más conocido donde se recopilan datos experimentales complejos. Estos datos plantean dos problemas:
    - Los datos deben procesarse previamente de tal forma que los algoritmos de aprendizaje automático los empleen adecuadamente.
    - El análisis de los datos, que depende de lo que se esté buscando. En el caso de los datos de microarrays, las aplicaciones más típicas son la identificación de patrones de expresión, la clasificación y la inducción de redes genéticas.
- *Biología de sistemas* es otro dominio en el que la biología y el aprendizaje automático trabajan juntos. Resulta complicado modelar los procesos vitales que tienen lugar en el interior de la célula. Por esto, los métodos computacionales son increíblemente útiles para modelar redes biológicas (Bower & Bolouri, 2001), en especial redes genéticas, redes de transducción de señales y vías metabólicas.
- La *evolución* y la reconstrucción de árboles filogenéticos emplean técnicas de aprendizaje automático. Los árboles filogenéticos son diagramas que representan las relaciones evolutivas entre organismos. Antes se construían en base a características morfológicas, metabólicas… Actualmente, gracias a la gran cantidad de secuencias genómicas disponibles, los algoritmos de

construcción de árboles filogenéticos se basan en la comparación entre diferentes genomas (Baldi, Brunak, & Bach, 2001). Dicha comparación se lleva a cabo a través de alineación de múltiples secuencias y técnicas de optimización.

- Un efecto secundario de la aplicación de técnicas computacionales a la creciente cantidad de datos es un incremento de las publicaciones disponibles. Esto proporciona una nueva fuente de información valiosa, donde se requieren técnicas de *minería de textos* para la extracción de conocimiento. Por tanto, la minería de textos se está volviendo cada vez más interesante en biología computacional y se está aplicando en la notación funcional, la predicción de la ubicación celular y el análisis de interacción de proteínas (Krallinger & Erhardt, 2005).

Además de las aplicaciones mencionadas, también se emplean técnicas computacionales para resolver otro tipo de problemas, como el diseño eficiente de PCR o el análisis de imágenes biológicas.

La bioinformática pertenece a una de las áreas dentro de la investigación más nuevas en el terreno de la biología molecular. Cuenta con infinidad de herramientas diseñadas para resolver problemas biológicos que requieren el manejo de grandes bases de datos, información biológica distribuida, representaciones de conocimiento difícil y la formulación de modelos acerca del funcionamiento de sistemas celulares y predicciones sobre su comportamiento.

Las técnicas bioinformáticas se encargan de tratar procesos experimentales, ordenar resultados, analizar los mismos, crear modelos para diseñar hipótesis y proponer experimentos nuevos y ayudar a extraer información biológica.

La bioinformática resulta imprescindible en el proceso de I+D de la industria farmacéutica, entre otras cosas, porque contribuye a adaptar los costes de los ensayos clínicos a tamaños de poblaciones adecuados (Garzón, 2020).

Como se ha comentado antes, la genómica es una de las ramas de la bioinformática en la que más repercute el empleo de técnicas de aprendizaje automático. Es por este motivo que se va a incidir un poco más en esta área que en otras menos relevantes.

5.1. APRENDIZAJE AUTOMÁTICO EN GENÓMICA

En este siglo se ha comenzado a descubrir la secuencia del genoma humano (Lander et al., 2001). Se han secuenciado organismos en el reino animal y vegetal (Nature, 2000) y se han elaborado más de sesenta proyectos de secuenciación del genoma eucariota.

Hoy en día, se cuenta con tecnologías que sirven para buscar enfermedades degenerativas, como por ejemplo, el cáncer (Guyon & De, 2003). Una de las tecnologías más presentes actualmente son los *microarrays* de ADN, que cuantifican la expresión génica de muchos genes al mismo tiempo (Moreno & Clin, 2004). Diversas investigaciones muestran que los perfiles de expresión génica ofrecen información con el fin de diferenciar un tipo de cáncer dentro de tejido con una morfología parecida y elaborar un diagnóstico mejor y sugerir terapias para combatir dicha enfermedad (Alon et al., 1999). La selección de genes es comúnmente asociada a la clasificación de tumores, cuya meta es encontrar un conjunto de genes destacables de un *microarray*. No se trata de un trabajo sencillo, ya que este pequeño chip cuenta con genes informativos y poco relevantes también (ruido). Los *biomarcadores* son necesarios para elaborar pruebas de diagnóstico. Los métodos basados en *filtros* y *wrapper* resultan bastante útiles a la hora de resolver el problema de la selección y clasificación de genes. Este tipo de método origina un pre-procesamiento que se basa en la puntuación que da una técnica de filtrado para realizar la limpieza del *microarray*.

La tecnología de los *microarrays* maneja cantidades ingentes de datos genéticos que pertenecen a diferentes enfermedades. Las grandes dimensiones de los *microarrays* ocasionan un problema de precisión de análisis y dificultad computacional (Pérez-Rubido, 2013).

Para resolver el problema de la gran cantidad de datos que posee un microarray y la presencia de ruido se han desarrollado técnicas de minería de datos y de aprendizaje automático que sirven para extraer información importante explorando el *microarray* (Zhang, Ding, & Li, 2008).

La herramienta más empleada consiste en clasificar características para diferencias entre distintas clases de muestras de tejido de enfermedades (Zhang et al., 2008).

Los algoritmos de aprendizaje automático que se fundamentan en métodos de filtros o *wrapper*, descartan la información menos importante del microarray con una puntuación (valor del gen) (Yu et al., 2010).

En la actualidad se usan algoritmos híbridos que mezclan distintas técnicas de filtros y *wrapper*, que se emplean para elegir y extraer genes que se consideran importantes para diagnosticar una enfermedad (Montiel, 2016).

Un tema importante en el análisis de datos de *microarrays* de expresión génica es el descubrimiento de genes capaces de diferenciar entre muestras de diferentes poblaciones, es decir, genes que son relevantes para determinados procesos. Estos genes se llaman genes informativos, biomarcadores o genes expresados diferencialmente (*DEGs*). El descubrimiento de DEGs también es útil para compañías farmacéuticas que quieren identificar genes que pueden ser atacados por fármacos. En los últimos años, se ha puesto mucho empeño en el desarrollo de metodologías para el descubrimiento de DEGs. El problema aún es desafiante y emergen nuevos algoritmos alternativos a los ya existentes. A pesar del amplio rango de enfoques propuestos para solucionar este problema, muchos algoritmos comparten elementos comunes que difieren solo en los detalles de cada uno (Lazar et al., 2012).

La predicción y descubrimiento de enfermedades y la reconstrucción de redes reguladoras de genes de datos de expresión génica son aplicaciones típicas de *microarrays* de expresión génica (Penfold & Wild, 2011). Las soluciones a estos problemas requieren técnicas de aprendizaje automático, como clasificación supervisada, *clustering* y regresión. La aplicación directa de estos métodos en datos de grandes dimensiones normalmente es ineficiente (Somorjai, Dolenko, & Baumgartner, 2003). Por ello, sería ideal seleccionar un pequeño subconjunto de genes/características que sea discriminativo entre los subgrupos de muestras. La selección de características consiste en identificar el conjunto de genes cuyos niveles de expresión son indicativos de una característica (clínica/biológica) que interesa particularmente. Seleccionar los genes más informativos consiste en identificar el subconjunto de genes más discriminativo, a través de toda la población de muestras. Esta definición solo es válida para los problemas de clasificación que están claramente identificados de antemano. En los últimos años se han propuesto diferentes estrategias para la selección de genes: filtrado, envoltorio, técnicas integradas y más recientemente técnicas de conjunto (Yang, Yang, Zhou, & Zomaya, 2016).

Las *técnicas de filtrado (filter techniques)* evalúan el poder discriminativo de los rasgos basándose sólo en propiedades intrínsecas de los datos. Por regla general, estos métodos estiman una puntuación de relevancia y se emplea un umbral para seleccionar los rasgos/genes que mejor puntúan. Las técnicas de filtrado no necesariamente se usan para construir predictores. Incluso cuando el subconjunto de rasgos no es óptimo, es preferible debido a su escalabilidad estadística y computacional. Hay dos estrategias de filtrado.

- En la *estrategia de clasificación*, se seleccionan los genes que se encuentran en la cima en el ranking de acuerdo a índices de relevancia estimados con una función de puntuación predefinida. Lo primero sería cuantificar la diferencia de expresión entre diferentes grupos de muestras en orden decreciente mediante una función de puntuación. Después, habría que estimar la significación estadística de las puntuaciones (se suele cortar en 0,05). Posteriormente, se seleccionan los genes que más cambian. Por último, quedaría validar el subconjunto de genes seleccionado en el paso anterior (Guyon & Elisseefl, 2006).
- La *estrategia de búsqueda espacial* consiste en adoptar una estrategia de optimización que propondrá el subconjunto de genes más informativo y menos redundante. Primero, hay que

definir una función de coste para optimizar. En segundo lugar, se busca un subconjunto de genes que optimice la función de coste mediante un algoritmo de optimización. Por último, como en el caso anterior, se valida el subconjunto de genes seleccionado.

Las *técnicas de envoltorio (wrapper techniques)* seleccionan el subconjunto de rasgos más discriminante minimizando la predicción del error de un clasificador particular. Estos métodos dependen del clasificador que se use y se critican principalmente por su gran demanda computacional.

Las *técnicas de integración (embedded techniques)* representan una clase diferente de métodos en el sentido de que permitan las interacciones con el algoritmo de aprendizaje pero el tiempo computacional es más corto que en los métodos de envoltorio.

Las *técnicas de ensamble (ensemble techniques)* representan una clase relativamente nueva de métodos de selección de genes. Se han propuesto para hacer frente a los problemas de inestabilidad observados en muchas técnicas cuando hay pequeñas perturbaciones en el entrenamiento. Estos métodos se basan en diferentes estrategias de submuestreo (Haury, Gestraud, & Vert, 2011).

Los métodos más frecuentes para el descubrimiento de genes de expresión diferencial son *Significance Analysis Microarrays* (SAM), Análisis de la Varianza (ANOVA), la prueba *t de Student*, el test de *Welch*, el *Fold-Change* o el *Rank Product*, que se usan en estudios comparativos.

5.1.1. EJEMPLO: SELECCIÓN Y CLASIFICACIÓN DE GENES CON UN MÉTODO HÍBRIDO FILTRO/*WRAPPER*

En la primera fase de la selección se plantea un método mixto que mezcla diferentes técnicas de filtro con el objetivo de solucionar el problema del ruido en los *microarrays*. En la segunda fase se emplea un modelo de selección y clasificación usando búsqueda gravitacional y *k-Nearest Neighbor*.

5.1.1.1. *MICROARRAYS* DE ADN

Los datos de un microarray se representan con una matriz, cuyas filas son los genes y cuyas columnas son las muestras. En cada celda se encuentra un valor de expresión génica, representando la intensidad del gen en cada muestra (Huang, Tao, Li, & On, 2012).

Ecuación 23. *Matriz de expresión génica*

$$x_{ij} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1n_m} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2n_m} \\ x_{31} & x_{32} & x_{33} & \cdots & x_{3n_m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n_g1} & x_{n_g2} & x_{n_g3} & \cdots & x_{n_gn_m} \end{bmatrix}$$

Las x son los datos genómicos, ng (gen, fila) los genes dentro de la matriz y nm (muestras, columnas) las muestras en la matriz.

5.1.1.2. PRE-PROCESAMIENTO

Los datos de los *microarrays* suelen estar en distintas escalas numéricas y/o distribuciones de probabilidad. Por esta razón, hay que tipificar los datos, es decir, transformarlos de tal forma que sus valores oscilen entre cero y uno, a pesar de que cada uno provenga de escalas o distribuciones diferentes. Una opción para llevar a cabo esta parte del pre-procesamiento es normalizar en base a una técnica de min-máx (Martinez, Martinez, & Solka, 2017):

Ecuación 24. *Tipificación con min-max*

$$X' = \frac{X - Min(X)}{Max(X) - Min(X)'}$$

X es la base de datos de la que partimos y $Min(X)$ y $Max(X)$ los datos mínimo y máximo de la base de datos, respectivamente. Por último, X' es la base de datos ya tipificada.

Una vez que tenemos los datos normalizados y continuando en la fase de pre-procesamiento, lo próximo sería hacer la primera selección de genes con técnicas de filtrado. Se quiere conseguir un ranking de los genes del *microarray* con cada método, estipulando un valor de pertenencia que ayude diferenciar los genes importantes de los que no lo son. Algunos de los métodos que se pueden utilizar en este paso son (Alberto et al., 2017):

-BSS/WSS: Se trata de un método para seleccionar genes que consiste en calcular la razón de la suma de cuadrados entre clases y la suma de cuadrados dentro de las clases. j es el gen en cuestión.

Ecuación 25. *Razón de BSS y WSS*

$$\frac{BSS(j)}{WSS(j)} = \frac{\sum_i \sum_k I(y_i = k)\left(\bar{x}_{kj} - \bar{x}_{.j}\right)^2}{\sum_i \sum_k I(y_i = k)\left(x_{ij} - \bar{x}_{kj}\right)^{2'}}$$

$\bar{x}_{.j}$ indica la expresión media del gen j mediante todas las muestras y $\bar{x}_{kj}$ representa la expresión media del gen j en todas las muestras para la clase k.

-Relación señal a ruido: Consiste en reconocer los patrones de expresión génica que tengan máxima diferencia en la expresión media entre dos grupos y la varianza mínima de expresión en cada grupo.

Ecuación 26. *Relación señal a ruido*

$$SNR = \left|\frac{\mu_1 - \mu_2}{\sigma_1 - \sigma_2}\right|$$

μ_1 y μ_2 son las expresiones medias de las clases 1 y 2, respectivamente; mientras que σ_1 y σ_2 son las desviaciones típicas de las muestras de cada clase.

-Información mutua: Se puede definir como medida de la dependencia mutua entre dos variables aleatorias, es decir, medida de la entropía de una variable aleatoria X (en este caso gen A), por el conocimiento de la variable aleatoria Y (en este caso gen B) (Marinescu, 2011). La entropía, a su vez,

es una reducción de la incertidumbre. Se necesita elegir dos genes A y B al azar con distribuciones distintas y distribución de probabilidad conjunta. La información mutua se halla de la siguiente manera:

Ecuación 27. *Información mutua*

$$I(A;B) = \sum_{a_i}\sum_{b_j} P(a_i, b_j) log \frac{P(a_i, b_j)}{P(a_i) \cdot P(b_j)}$$

$P(a_i, b_j)$ es la probabilidad conjunta de los genes A y B, $P(a_i)$ es la probabilidad de A y $P(b_j)$ es la probabilidad de B.

En la segunda etapa se puede implementar un algoritmo que seleccione genes basándose en una búsqueda gravitacional y un clasificador KNN.

5.1.1.3. BÚSQUEDA GRAVITACIONAL

La *búsqueda gravitacional* es un algoritmo que propuso *Rashedi* en 2009 (Xiang, Han, Duan, Qiang, & …, 2015) y que se basa en la ley gravitacional de *Newton*, que dice que la fuerza de gravedad que hay entre dos cuerpos es directamente proporcional al producto de sus masas e inversamente proporcional a la distancia entre ellos elevada al cuadrado. Las soluciones de la población del algoritmo de búsqueda gravitacional se relacionan entre ellas mediante la gravedad. La masa de cada agente indica la calidad con la que cuenta. Los agentes son objetos y se mueven hacia otros más pesados por la gravedad que crean. El agente cuya masa es la más pesada constituye la solución más acertada del algoritmo (Sajedi & Razavi, 2017).

Ecuación 28. *Constante de gravedad en la iteración t*

$$G(t) = G_{0^{e^{-\alpha t/T}}}$$

G_0 y α se inicializan al empezar la búsqueda y sus valores se van reduciendo en cada iteración. T son las iteraciones totales.

Ecuación 29. *Fuerza de gravedad en base a la ley de Newton*

$$F = G\frac{M_{aj} \cdot M_{pi}}{R^2}$$

Este algoritmo cuenta con tres masas diferentes: masa gravitacional activa (M_{aj}), masa gravitacional pasiva (M_{pi}) y masa inercial.

Ecuación 30. *Aceleración del objeto i*

$$a_i = \frac{F_{ij}}{M_{ii}}$$

Según la segunda ley de *Newton*, al aplicar una fuerza sobre un objeto, éste se desplaza con una aceleración que depende de la fuerza que se aplica (F_{ij}) y de la masa del objeto (M_{ii}).

En las próximas ecuaciones se observa cómo se actualizan las velocidades y posiciones de los agentes:

Ecuación 31. *Actualización de velocidad*

$$V_i(t+1) = rand_i \cdot V_i(t) + a_i(t)$$

Ecuación 32. *Actualización de posición*

$$X_i(t+1) = rand_i \cdot V_i(t) + a_i(t)$$

5.1.1.4. CLASIFICADOR *K-NEAREST NEIGHBOURS*

Como se ha explicado en el apartado Algoritmo de *K-Nearest Neighbours*, se trata de un algoritmo de clasificación basado en la hipótesis de que los individuos de una población comparten características con los de su entorno, por lo que observando a los vecinos más cercanos de un individuo se puede extraer información sobre el mismo.

Sea $x^1, x^2, \dots, x^n$ una muestra con una función de densidad $f(x)$ desconocida. $f(x)$ se estima partiendo de un elemento central de la muestra x que aumenta hasta abarcar k elementos con distancia Euclídea parecida, donde k es un valor definido de forma arbitraria (Suguna & Thanushkodi, 2010). Entonces, se tiene:

Ecuación 33. *Función de densidad*

$$f(x) = \frac{k/n}{V_k(x)}$$

$V_k(x)$ es el volumen de un elipsoide centrado en x, cuyo radio es la distancia Euclídea entre x y el *k-ésimo* vecino más cercano.

5.1.1.5. ALGORITMO HÍBRIDO GSA/KNN

1) Se fijan los valores iniciales de G_0, α, ε y el número de iteraciones t.
2) Se genera la población inicial de forma aleatoria, con una distribución uniforme. Cuenta con N agentes asociados a cada gen del *microarray*.

3) Se introduce el clasificador KNN en la función de coste del algoritmo con el fin de evaluar a los agentes de la población. Para comprobar el error del clasificador se usa validación cruzada.
4) La constante de gravedad se va actualizando con la fórmula de la Ecuación 15.
5) La fuerza cuando el agente j actúa sobre el agente i en un tiempo determinado, se calcula de la siguiente manera:

Ecuación 34. *Fuerza en un tiempo t*

$$F_{ij}^{d}(t) = G(t)\frac{M_{pi}(t) \cdot M_{aj}(t)}{R_{ij}(t) + \varepsilon}(x_j^d(t) - x_i^d(t))$$

M_{aj} es la masa gravitacional activa del agente j, M_{pi} es la masa gravitacional pasiva del agente i y $G(t)$ es la constante de gravedad en el momento t.

6) La fuerza que actúa sobre el agente i en cada iteración se calcula:

Ecuación 35. *Fuerza sobre i en t*

$$F_i^d(t) = \sum_{j \in Kbest, j \neq i} rand_j F_{ij}^d(t)$$

$Kbest$ son los K primeros agentes con masa más grande.

7) Se calcula la masa inercial de la siguiente forma:

Ecuación 36. *Masa inercial* m_i

$$m_i(t) = \frac{fit_i - worst(t)}{best(t) - worst(t)}$$

Ecuación 37. *Masa inercial* M_i

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}$$

8) Se calcula la aceleración del agente i mediante la Ecuación 17, la velocidad con la Ecuación 18 y la posición a través de la Ecuación 19. Se va aumentando el número de iteraciones hasta que se cumplen los criterios de paro.

6. CONCLUSIONES

Después de haber investigado sobre biología y, en concreto, las ramas de la genómica y la proteómica, de haber aprendido una serie de técnicas de aprendizaje automático, y de haber combinado ambos "mundos" para resolver problemas reales; se pueden extraer una serie de conclusiones sobre el trabajo llevado a cabo.

El aprendizaje automático constituye una parte importante de la Inteligencia artificial que, a su vez, engloba el Aprendizaje profundo, área que ha aportado numerosos avances que hacen del *machine learning* una herramienta aún más útil a la hora de buscar soluciones. Dicho instrumento va de la mano de la Minería de datos, que también emplea algoritmos para tratar con cantidades ingentes de datos y, con ello, se da pie a mencionar al Big Data del que tanto se está hablando en los últimos años. Cada vez hay mayor capacidad de almacenamiento de datos, lo cual nos lleva a pensar lo importante que resulta desarrollar nuevas técnicas más robustas que nos ayuden a analizar cantidades cada vez más grandes de información.

El aprendizaje automático se puede clasificar en supervisado, no supervisado y por refuerzo, como se muestra en Tipos de aprendizaje automático. El aprendizaje supervisado se diferencia del no supervisado en que se saben a priori las categorías en que se quieren dividir los datos. Por otro lado, en el aprendizaje no supervisado se busca descubrir nuevos patrones o resultados y no la reproducción de un resultado ya conocido. En los problemas resueltos mediante aprendizaje por refuerzo no se conoce la solución y la manera de entrenar el modelo es a través de refuerzos positivos o negativos según los resultados, haciendo que se tome una decisión. Este enfoque se emplea cuando es posible asignar una recompensa o una penalización pero no se sabe cómo llegar al resultado.

Todas las técnicas del presente estudio se emplean con frecuencia en diversas áreas de la ciencia. Sin embargo, siempre hay unas que destacan más sobre las demás.

Uno de los algoritmos más importantes puede ser las Redes neuronales artificiales. Como su propio nombre indica, se trata de unas estructuras que simulan el funcionamiento de una neurona biológica. Son capaces de crear patrones, reconocer información o resolver enigmas complejos, siendo de gran utilidad para muchos sectores. Tienen la capacidad de transmitir datos a gran velocidad entre dos puntos y, con el paso del tiempo, pueden llegar a funcionar de manera autónoma.

Otras técnicas que cabe destacar son los Algoritmos de agrupamiento o Clustering, que se pueden englobar dentro del aprendizaje no supervisado, debido a la falta de grupos definidos con anterioridad. Consiste en agrupar un conjunto de objetos similares en *clusters*. Dentro de este grupo de técnicas, las más destacables son el Algoritmo de k-means y el Algoritmo de K-Nearest Neighbours.

Es imprescindible resaltar la importancia de las técnicas de reducción de la dimensionalidad. Dentro de este grupo, la más conocida y utilizada es el Análisis de Componentes Principales. Este método reduce la dimensión del espacio al encontrar nuevos vectores que maximizan la variación lineal de los datos con los que se trabaja. Lo positivo es que, a pesar de reducir el número de variables drásticamente, no se pierde demasiada información gracias a las correlaciones fuertes de los datos.

Como se ha visto, el aprendizaje automático se puede emplear en muchos campos de la ciencia. Concretamente, en el sector de la bioinformática está cobrando cada vez más importancia y están desarrollándose nuevos algoritmos para mejorar y agilizar los estudios biológicos, que sin la ayuda de la estadística actual, llevarían muchísimo más tiempo y dinero. En medicina y biología se trabaja mucho con los genes para detectar o clasificar enfermedades. Las técnicas de aprendizaje automático, junto con los *software* estadísticos con lo que se cuenta hoy en día, constituyen una pieza fundamental para hacer de estos estudios genéticos una tarea más sencilla y rápida. Estas herramientas estadísticas generalmente trabajan en base a los datos recogidos de experimentos que se realizan con Microarrays de ADN. Uno de los usos más frecuentes es el que hace referencia a la expresión diferencial de los genes. Para llevar

a cabo este tipo de tareas se puede emplear un algoritmo que recibe el nombre de SAM (*Significance Analysis Microarrays*) y que se implementa en el programa estadístico R. Consiste en ver qué genes son los que más se expresan (los más importantes) en valor absoluto en dos o más categorías diferentes, generalmente pacientes y controles. De esta manera se puede comprobar cómo afectan unos genes determinados a una enfermedad y con esos resultados estudiar los genes que más cambian ya en el laboratorio.

Finalmente, y como se ha puntualizado anteriormente, se puede decir que las técnicas de aprendizaje automático suponen un gran avance para resolver problemas bioinformáticos. Es por esto que se deben seguir estudiando nuevos métodos y mejorando los ya existentes, con el objetivo de solucionar dificultades que nos acompañan en el mundo de la ciencia.

7. BIBLIOGRAFÍA

Abdi, H. (2007). RV coefficient and congruence coefficient. *Encyclopedia of Measurement and Statistics*.

Abdi, Hervé. (2003). Multivariate Analysis. In *researchgate.net*. Retrieved from http://www.utdallas.edu/

Abdi, Hervé. (2007). Singular Value Decomposition (SVD) and Generalized Singular Value Decomposition (GSVD). In *m2multimedia.u-bourgogne.fr*. Retrieved from http://www.utd.edu/

Abril, H. M. (2018). *Clasificadores para el reconocimiento automático de células blásticas en leucemias agudas linfoides y mieloides*.

Aerts, S., Loo, P. Van, Moreau, Y., & De Moor, B. (2004). A genetic algorithm for the detection of new cis-regulatory modules in sets of coregulated genes. *BIOINFORMATICS APPLICATIONS NOTE*, *20*(12), 1974–1976. https://doi.org/10.1093/bioinformatics/bth179

Alberto, L., Montiel, H., Edgardo, C., Pérez, C., Gabriel, J., Ruiz, R., … Ixtepec, C. (2017). *Selección y clasificación de genes cancerígenos utilizando un método híbrido filtro / wrapper Cancerous Genes Selection and Classification Using a Hybrid Filter / Wrapper Method*. *136*, 85–97.

Alberto Risueño Pérez. (2012). *Bioinformática aplicada a estudios del transcriptoma humano: análisis de expresión de genes, isoformas génicas y ncRNAs en muestras sanas y en cáncer*.

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., & Levine, A. J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. In *Cell Biology* (Vol. 96). Retrieved from www.pnas.org.

Aprendizaje automático por refuerzo. (2019). Retrieved from https://sitiobigdata.com/2019/10/27/aprendizaje-automatico-por-refuerzo-parte-4/

Arthur, D., & Vassilvitskii, S. (2006). *k-means++: The Advantages of Careful Seeding*.

Bagnato, J. I. (2018). Algoritmo k-Nearest Neighbor | Aprende Machine Learning. Retrieved April 19, 2021, from https://www.aprendemachinelearning.com/clasificar-con-k-nearest-neighbor-ejemplo-en-python/

Baldi, P., Brunak, S., & Bach, F. (2001). *Appendix A Statistics A.1 Decision Theory and Loss Functions*.

Banafa, A. (2016). ¿Qué es el aprendizaje profundo?

Bellot, N. (2020). *automático para el estudio del análisis de supervivencia en pacientes infectados por el VIH*.

Bower, J., & Bolouri, H. (2001). *Computational modeling of genetic and biochemical networks*.

Cajamarca, M. (2019). Inteligencia Artificial, Aprendizaje Automático y Aprendizaje Profundo. Retrieved from https://planetachatbot.com/inteligencia-artificial-aprendizaje-automático-y-aprendizaje-profundo-862ca9790bb9

Carter, R. J., Dubchak, I., & Holbrook, S. R. (2001). A computational approach to identify genes for functional RNAs in genomic sequences. *Nucleic Acids Research*, *29*(19), 3928–3938. https://doi.org/10.1093/nar/29.19.3928

Cattell, R. B. (1966). The scree test for the number of factors. *Multivariate Behavioral Research*, *1*(2), 245–276. https://doi.org/10.1207/s15327906mbr0102_10

Chow, C. (1968). Approximating discrete probability distributions with dependence trees. *Ieeexplore.Ieee.Org*. Retrieved from https://ieeexplore.ieee.org/abstract/document/1054142/?casa_token=bF1-

zBBIThYAAAAA:qUooiB2yruzpbt4MFRpFk09d1r1HlcxDQe9qKSpDP_uVkGlrjks60WO-cPEy9950KJYfksyaeg

Dray, S. (2008). On the number of principal components: A test of dimensionality based on measurements of similarity between matrices. *Elsevier*. Retrieved from https://www.sciencedirect.com/science/article/pii/S0167947307002939?casa_token=UlbZMGIEW00AAAAA:EawEmTakYZu5JftXIRnK3o6-wQRkFzJCAXoCPVODqqFuLGBGmg9K_orRR8sv9NXMYklF2WxSQQ

Duda, R.O., Hart, P.E., and Stork, D. G. (2001). *Pattern Classification*.

Fernández, E. J., & Britos, P. (2004). *CLASIFICADORES BAYESIANOS*. Retrieved from https://ri.itba.edu.ar//handle/123456789/2531

G, James, Witten D, Hastie T, T. R. (2013). *An introduction to statistical learning*.

Garzón, J. A. C. (2020). *DNA-Microarray Data Mining*. 1–30.

González, L. (2019). Clustering Jerárquico - Agrupar elementos con minería de datos. Retrieved April 20, 2021, from https://estrategiastrading.com/clustering-jerarquico/

Grado, T. D. E. F. I. N. D. E., Boyarshinov, V., Sociedad Argentina de Informática e Investigación Operativa (SADIO), Basyarudin, Silvestre, L. J., de las Heras Mínguez, G., … González, F. A. (2018). Uso de algoritmos automático aplicado a bases de datos genéticos. *Высшей Нервной Деятельности*, *10*(2), 44. Retrieved from http://openaccess.uoc.edu/webapps/o2/bitstream/10609/65426/6/rgagoTFM0617memoria.pdf%0Ahttps://www.managementsolutions.com/sites/default/files/publicaciones/esp/machine-learning.pdf%0Ahttp://sedici.unlp.edu.ar/handle/10915/41722%0Ahttps://books.google.ca/

Guyon, I., & De, A. M. (2003). An Introduction to Variable and Feature Selection André Elisseeff. In *Journal of Machine Learning Research* (Vol. 3). Retrieved from https://www.jmlr.org/papers/volume3/guyon03a/guyon03a.pdf?ref=driverlayer.com/web

Guyon, I., & Elisseefl, A. (2006). An introduction to feature extraction. *Studies in Fuzziness and Soft Computing*, *207*, 1–25. https://doi.org/10.1007/978-3-540-35488-8_1

Haury, A. C., Gestraud, P., & Vert, J. P. (2011). The influence of feature selection methods on accuracy, stability and interpretability of molecular signatures. *PLoS ONE*, *6*(12). https://doi.org/10.1371/journal.pone.0028210

Hern, A. (2020). *Fabián Alberto Hernández Tarapués*.

Hern, A. G. (2004). *Aprendizaje Automático : Algoritmos genéticos*. 1–13.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational*. Retrieved from https://psycnet.apa.org/record/1934-00645-001

Huang, Q., Tao, D., Li, X., & On, A. L.-I. T. (2012). Parallelized evolutionary learning for detection of biclusters in gene expression data. *Ieeexplore.Ieee.Org*. Retrieved from https://ieeexplore.ieee.org/abstract/document/5728798/?casa_token=CTUR-BevJawAAAAA:KGa3YCVLDA4ouUKzBcEk2N9HMTZmPAIADswHP8z8dkiN2Lf6dSTkc8dQd1J-VGKDKLpMLK5a7A

Isabel, M., Herrera, L., Hoyo, C. V., & Martinez, D. F. (2020). *Evaluación de análisis de clustering jerárquico en datos moleculares de alta dimensión* . Retrieved from http://hdl.handle.net/10609/120648

Jimena Martínez, M. (2017). *Desarrollo de Métodos Analíticos y de Predicción para Informática Molecular Basados en Técnicas de Aprendizaje Automático y Visualización*. Retrieved from http://repositoriodigital.uns.edu.ar/bitstream/123456789/3727/1/Tesis doctoral - María Jimena Martínez.pdf

Kaiser, H. (1958). The varimax criterion for analytic rotation in factor analysis. *Springer*. Retrieved

from https://idp.springer.com/authorize/casa?redirect_uri=https://link.springer.com/content/pdf/10.1007/BF02289233.pdf&casa_token=7GP62_1QDpoAAAAA:J5IYHdDdKdSUoh8utvMRjpHsBC-hFcvjMHoapOzVMxUHDAIoJxEjWVjUcEzLpoATq5-x-6KEZs1aYqvy

Krallinger, M., & Erhardt. (2005). Text-mining approaches in molecular biology and biomedicine. *Elsevier*. Retrieved from https://www.sciencedirect.com/science/article/pii/S1359644605033763?casa_token=mKN2PXB2Rw0AAAAA:zAIGTwMigGU55D_QHyEAPFhAFtRLfDailOxHb5XT0J67cDmuBOVcN-xucxEqypnZ2Wtv6x9CoQ

Lander, E., Linton, L., Birren, B., Nusbaum, C., Nature, M. Z.-, & 2001, U. (2001). Erratum: initial sequencing and analysis of the human genome: international human genome sequencing consortium (nature (2001) 409 (860-921)). *Profiles.Wustl.Edu*. Retrieved from https://profiles.wustl.edu/en/publications/erratum-initial-sequencing-and-analysis-of-the-human-genome-inter

Lazar, C., Taminau, J., Meganck, S., Steenhoff, D., Coletta, A., Molter, C., … Nowé, A. (2012). A survey on filter techniques for feature selection in gene expression microarray analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *9*(4), 1106–1119. https://doi.org/10.1109/TCBB.2012.33

López-Gartner, G., Agudelo-Valencia, D., Castaño, S., Isaza, G. A., Castillo, L. F., Sánchez, M., & Arango, J. (2015). Identification of a putative ganoderic acid pathway enzyme in a Ganoderma Australe transcriptome by means of a Hidden Markov model. *Advances in Intelligent Systems and Computing*, *375*, 107–115. https://doi.org/10.1007/978-3-319-19776-0_12

López González Tutorizado, R., Ángel, :, Bayarri, A., & Azpitarte, R. L. (2016). *Detección Automática de Órganos en adquisiciones de Tomografía Computarizada con Métodos de Machine Learning*.

María Teresa Ortiz, F. G. (2015). 7.3 Algoritmo hacia adelante-hacia atrás (forward-backward) | Estadística Multivariada. Retrieved May 18, 2021, from https://est-mult.netlify.app/algoritmo-hacia-adelante-hacia-atrás-forward-backward.html

Marinescu, D. (2011). *Classical and quantum information*. Retrieved from https://books.google.es/books?hl=es&lr=&id=He-L5CLq1PUC&oi=fnd&pg=PP1&dq=Dan+C.+Marinescu,+Gabriela+M.+Marinescu,+%22Classical+and+Quantum+Information%22,Academic+Press+2012&ots=47NVO5FOFH&sig=-ZpUnHX_cl1a_RFIB8uEqULssYg

Martinez, W., Martinez, A., & Solka, J. (2017). *Exploratory data analysis with MATLAB*. Retrieved from https://books.google.es/books?hl=es&lr=&id=PD0PEAAAQBAJ&oi=fnd&pg=PP1&dq=Martínez,+W.+L.,+Martinez,+A.+R.:+Exploratory+Data+Analysis+with+MATLAB®.+A+CRC++Press+Company.+Boca+Ratón+London+New+York+Washington,+D.C+(2005)&ots=oXeeitlun2&sig=d8UUynnabCoTTc3XHfsNBp8mwMA

Mathé, C., Sagot, M. F., Schiex, T., & Rouzé, P. (2002, October 1). Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Research*, Vol. 30, pp. 4103–4117. https://doi.org/10.1093/nar/gkf543

Mendoza Lombana, S. P. (2009). *BIOINFORMÁTICA Historia*.

Merkle. (2020). El algoritmo K-NN y su importancia en el modelado de datos. Retrieved April 19, 2021, from https://www.merkleinc.com/es/es/blog/algoritmo-knn-modelado-datos

Mitra, S., & Acharya, T. (2005). *Data mining: multimedia, soft computing, and bioinformatics*. Retrieved from https://books.google.es/books?hl=es&lr=&id=VPeOaKNfDlgC&oi=fnd&pg=PR14&dq=S.+Mitra+and+T.+Acharya.+Data+mining:+multimedia,+soft+computing+and+bioinformatics.+John+

Wiley+%26+Sons,+2003&ots=icURuy5_Wd&sig=M-ORCXY17gAIIpkOI31k0zAqThU

Monleon-Getino, A. (2015). El impacto del Big-data en la Sociedad de la Información. Significado y utilidad. *Historia y Comunicacion Social*, *20*(2), 427–445. https://doi.org/10.5209/rev-HICS.2015.v20.n2.51392

Montiel, L. (2016). Hybrid algorithm applied on gene selection and classification from different diseases. *IEEE*. Retrieved from https://ieeexplore.ieee.org/abstract/document/7437242/?casa_token=fu-pK6jB8nEAAAAA:_z5xBc4j5a_FMyeqzMBFUNsArlXGDn597IWUBjwxtqnXwtAJR28XWTf_xQeUnJ2ujQGrmvmwpA

Moreno, V., & Clin, X. S. (2004). Uso de chips de ADN (microarrays) en medicina: fundamentos técnicos y procedimientos básicos para el análisis estadístico de resultados. *Academia.Edu*. Retrieved from https://www.academia.edu/download/45978831/es-revista-medicina-clinica-2-pdf-13057538-S300.pdf

Moya, R. (2016). ¿Que es el Clustering? - Jarroba. Retrieved April 16, 2021, from https://jarroba.com/que-es-el-clustering/

Nature, C. F. (2000). Analysis of the genome sequence of the flowering plant Arabidopsis thaliana. *Hsrc.Himmelfarb.Gwu.Edu*. Retrieved from https://hsrc.himmelfarb.gwu.edu/smhs_pharm_facpubs/354/

Oaks, T., & Sage, C. (2010). Barycentric Discriminant Analysis (BADIA). In *researchgate.net*. Retrieved from https://www.researchgate.net/publication/267302779

Olavide, U. P. De. (2006). *Minería de Datos : Conceptos y Tendencias*. *29*(29), 11–18.

Olivera, O. G.-O. (2019). Redes Neuronales artificiales: Qué son y cómo se entrenan | Xeridia. Retrieved April 13, 2021, from https://www.xeridia.com/blog/redes-neuronales-artificiales-que-son-y-como-se-entrenan-parte-i

Orozco, S., & Jeferson, A. (2016). Aplicación de la inteligencia artificial en la bioinformática, avances, definiciones y herramientas* Aplication of Artificial Intelligence in Bioinformatics, advances, definitions and tools. *UGCiencia*, 159–171.

Pascual, D., Pla, F., & Sánchez, S. (2007). *Algoritmos de agrupamiento*.

Pearson, K. (1901). LIII. On lines and planes of closest fit to systems of points in space . *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), 559–572. https://doi.org/10.1080/14786440109462720

Penfold, C. A., & Wild, D. L. (2011). How to infer gene networks from expression profiles, revisited. *Royalsocietypublishing.Org*, *1*(6), 857–870. https://doi.org/10.1098/rsfs.2011.0053

Peres-Neto, P., & Jackson, K. S. (2005). How many principal components? Stopping rules for determining the number of non-trivial axes revisited. *Elsevier*. Retrieved from https://www.sciencedirect.com/science/article/pii/S0167947304002014?casa_token=JXoz7PbbL7oAAAAA:U1H8O3KOC9cet1kXOcE68_Vt255ns3pk-3YLcrR_0RzAYA7B7KXxkcIWRvwj2pcPHgS_QqPBwA

Pérez-Rubido, R. (2013). Una revisión a algoritmos de selección de atributos que tratan la redundancia en datos microarreglos. *Revista Cubana de Ciencias*. Retrieved from http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992013000400002

Puyol Moreno, J. (2014). Una aproximación a Big Data. *Revista de Derecho de La UNED (RDUNED)*, (14), 471. https://doi.org/10.5944/rduned.14.2014.13303

Rivera Lozano, M. (2011). *EL PAPEL DE LAS REDES BAYESIANAS EN LA Chow, C., Theory, C. L.-I. transactions on I., & 1968, undefined. (n.d.). Approximating discrete probability distributions with dependence trees. Ieeexplore.Ieee.Org. Retrieved from https://ieeexplore.ieee.org/abstra*.

Rodrigo, J. A. (2018). Reglas de asociación y algoritmo Apriori con R. Retrieved April 20, 2021, from https://www.cienciadedatos.net/documentos/43_reglas_de_asociacion

Rouhiainen, L. (2018). *Inteligencia artificial 101*. 352. Retrieved from https://planetadelibrosar0.cdnstatics.com/libros_contenido_extra/40/39307_Inteligencia_artificial.pdf

Sajedi, H., & Razavi, S. F. (2017). DGSA: discrete gravitational search algorithm for solving knapsack problem. *Operational Research*, *17*(2), 563–591. https://doi.org/10.1007/s12351-016-0240-2

Santiesteban-Toca, C. E., Casañola-Martin, G. M., & Aguilar-Ruiz, J. S. (2014). Las técnicas de aprendizaje automático en la predicción de estructura de proteínas: un enfoque desde la bioinformática. *Afinidad*, *71*(567), 219–227.

Santos, J. B. (2019). *Métodos de aprendizaje automático para detección de anomalías*. 1–106.

Sanz, E.-J. B.-H. (2016). *Algoritmos de clustering y aprendizaje automático aplicados a Twitter*.

Saporta, G., & Keita, N. N. (2009). *Principal Component Analysis: application to Statistical Process Control*. https://doi.org/10.1002/9780470611777.ch1ï

Schutte, D. (2006). The dendrogram technique as a tool to development questionnaires. *Journals.Co.Za*. Retrieved from https://journals.co.za/doi/abs/10.10520/EJC51478

Somorjai, R. L., Dolenko, B., & Baumgartner, R. (2003). Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, caveats, cautions. *BIOINFORMATICS*, *19*(12), 1484–1491. https://doi.org/10.1093/bioinformatics/btg182

Su, A. (2005). *Aprendizaje Automático*.

Sucar, L. E. (2006). *Redes Bayesianas*.

Suguna, N., & Thanushkodi, K. (2010). An improved k-nearest neighbor classification using genetic algorithm. *International Journal*. Retrieved from www.IJCSI.org

Thurstone, L. L. (1956). Multiple factor analysis. *Chicago: Univer. of Chicago Press*.

Tié, Félix, D. (2007). *REDES BAYESIANAS Y RIESGO OPERACIONAL Universidad de A Coruña*.

V, V. (2013). *The nature of statistical learning theory*. *Springer science & business media*.

Villazana, S., Arteaga, F., Seijas, C., & Rodriguez, O. (2012). *Estudio Comparativo entre Algoritmos de Agrupamiento Basado en SVM and Fuzzy C-Means based Clustering Applied to Arrhythmic ECG Signals: A Comparative Study*.

Wesley, A. (2006). *An Introduction to Search Engines and Web Navigation.* Retrieved from https://academic.oup.com/comjnl/article-abstract/49/4/500/353884

Wikipedia. (2020). K-medias. Retrieved April 19, 2021, from https://es.wikipedia.org/wiki/K-medias

Xiang, J., Han, X., Duan, F., Qiang, Y., & … X. X. (2015). A novel hybrid system for feature selection based on an improved gravitational search algorithm and k-NN method. *Elsevier*. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494615000642?casa_token=v972aVyiMJgAAAAA:3rjDE4_GTMIH_JwqKoPoMzo5LNSnBaUgDQjP67_j02BdxJzTRpy7dnded5lfPhrb-tspYqawwcv-

Yang, P., Yang, Y. H., Zhou, B. B., & Zomaya, A. Y. (2016). A review of ensemble methods in bioinformatics: * Including stability of feature selection and ensemble feature selection methods. In *ingentaconnect.com*. Retrieved from https://www.ingentaconnect.com/content/ben/cbio/2010/00000005/00000004/art00006

Yu, G., Feng, Y., Miller, D. J., Xuan, J., Hoffman, E. P., Org, E., … Shih, I.-M. (2010). Matched Gene

Selection and Committee Classifier for Molecular Classification of Heterogeneous Diseases Ie-Ming Shih. In *Journal of Machine Learning Research* (Vol. 11). Retrieved from http://www.cbil.ece.vt.edu/software.htm.

Zhang, Y., Ding, C., & Li, T. (2008). Gene selection algorithm by combining reliefF and mRMR. *BMC Genomics*, *9*(SUPPL. 2). https://doi.org/10.1186/1471-2164-9-S2-S27