Capítulo 14
ALGORITMOS Y MODELOS

Ya llevamos un buen recorrido a lo largo de las tecnologías inteligentes, sin embargo no hemos entrado en profundidad en sus métodos o, dicho de otro modo, no hemos aprendido por qué funcionan. Los siguientes capítulos, probablemente los más importantes del libro, quieren tapar estos huecos o dudas que a las mentes más ávidas y curiosas podría haber generado; y, al mismo tiempo, elevar a un nivel superior las "*tecnologías inteligentes*", mostrando las bases de su funcionamiento: ¿cómo se consigue que un algoritmo aprenda automáticamente a partir de ejemplos?

14.1 Introducción

En la actualidad, estamos rodeados de tecnología y datos que nos rodean en todas las áreas de nuestra vida. Desde redes sociales hasta servicios de *streaming*, pasando por motores de búsqueda y sistemas de recomendación, etc. el mundo digital está en constante evolución y crecimiento. Pero, ¿cómo es posible que estas plataformas y servicios puedan brindarnos experiencias personalizadas, precisas y eficientes? La respuesta radica en el Aprendizaje Automático y, más específicamente, en los **algoritmos y modelos** que impulsan esta tecnología.

El Aprendizaje Automático [*machine learning*] es una rama de la Inteligencia Artificial que se centra en desarrollar sistemas capaces de **aprender a partir de los datos**. En lugar de ser programados de manera explícita, estos sistemas pueden adaptarse y mejorar sus resultados a medida que se exponen a más información. El objetivo principal del Aprendizaje Automático es permitir a las máquinas "aprender" de manera similar a como lo haría un ser humano, identificando patrones, tomando decisiones y/o realizando predicciones.

Los algoritmos son el corazón del Aprendizaje Automático. Estos son conjuntos de instrucciones y reglas lógicas que permiten a las máquinas procesar y analizar los datos para extraer información útil. En otras palabras, los algoritmos son como "recetas" que guían a las máquinas en la forma en que deben procesar la información y tomar decisiones.

> ***Mi IA favorita dice**: Un algoritmo es un conjunto finito y ordenado de pasos o instrucciones que se sigue para resolver un problema específico o realizar una tarea. Los al-*

goritmos pueden implementarse en diversos contextos, desde cálculos matemáticos y procesamiento de datos hasta toma de decisiones y automatización, y son la base de muchos sistemas de computación y programas de software.

Por ejemplo, imagina que tienes una colección de fotografías y deseas crear un sistema de reconocimiento facial. El algoritmo de Aprendizaje Automático podría analizar todas las imágenes, identificar patrones en los rasgos faciales y aprender a distinguir diferentes personas. A medida que el sistema se expone a más imágenes, su precisión y capacidad para reconocer caras mejora, ya que el algoritmo aprende de cada imagen y refina su capacidad de identificación.

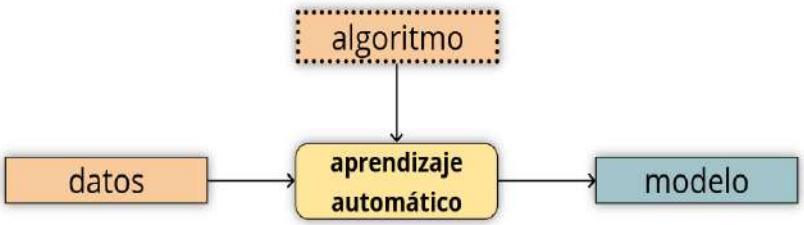


Figura 57: Relación entre datos, algoritmo y modelo en el aprendizaje automático.

Ya hemos hablado de modelos anteriormente, así que vamos a recordar el concepto. **Los modelos** son representaciones matemáticas de los datos y las relaciones que se encuentran dentro de ellos. En el contexto que nos ocupa, estos modelos son construidos por los algoritmos de Aprendizaje Automático y se utilizan para realizar predicciones o clasificaciones basadas en los datos de entrada. Los modelos pueden ser simples, como una línea recta en un gráfico, o más complejos, como redes neuronales con múltiples capas [*deep learning*].

¿Cómo afrontaríamos el mismo problema sin Aprendizaje Automático? De manera tradicional: una persona o equipo de personas, a partir de unos datos de ejemplo y/o de unas especificaciones técnicas [un análisis de requerimientos] diseñan y codifican [y prueban] un sistema informático, que hace las veces de modelo.

Tomemos otro ejemplo para comprender mejor cómo funcionan. Supongamos que queremos **predecir** el precio de una vivienda en función de sus características, como el tamaño, la ubicación y el número de habitaciones, entre otras. El modelo de Aprendizaje Automático podría analizar un conjunto de datos de viviendas anteriores, aprender las relaciones entre las características y los precios, y luego utilizar ese conocimiento para predecir el precio de una nueva vivienda en función de sus características.

Todas las tecnologías inteligentes que hemos nombrado hasta ahora, y las que nos hemos dejado en el tintero, se basan en su mayor parte en técnicas de Aprendizaje Automático; bien con un enfoque estocástico o algorítmico [*machine learning*], bien mediante una arquitectura más compleja de redes neuronales artificiales [*deep learning*].

Figura 58: Esquematización del método tradicional de producción de sistemas informáticos.

14.2 Aprendizaje automático y ciencia de datos.

La ciencia de datos y el aprendizaje automático son dos disciplinas estrechamente relacionadas que se ocupan del análisis y la interpretación de datos para obtener información y tomar decisiones. Aunque están relacionadas, cada una tiene su propio enfoque y conjunto de técnicas.

> *La **ciencia de datos** (lo veremos con más detenimiento en el Capítulo 15) es un campo interdisciplinario que combina conocimientos de matemáticas, estadísticas, programación y dominio del área en cuestión, para extraer conocimiento a partir de los datos. Se enfoca en el proceso de recopilar, limpiar, analizar y visualizar grandes volúmenes de información con el objetivo de descubrir patrones, identificar tendencias y resolver problemas complejos. La ciencia de datos utiliza herramientas y técnicas estadísticas y matemáticas para explorar y comprender los datos.*

Por otro lado, como ya hemos dicho, el aprendizaje automático es una rama de la inteligencia artificial que se basa en la idea de que las máquinas pueden aprender automáticamente a partir de los datos sin ser programadas explícitamente. En lugar de seguir reglas predefinidas, los algoritmos de aprendizaje automático aprenden a través de la experiencia por medio de los datos de entrenamiento. El aprendizaje automático se utiliza para desarrollar modelos y algoritmos que pueden realizar tareas específicas, como clasificación, predicción, reconocimiento de patrones y toma de decisiones, entre otros objetivos.

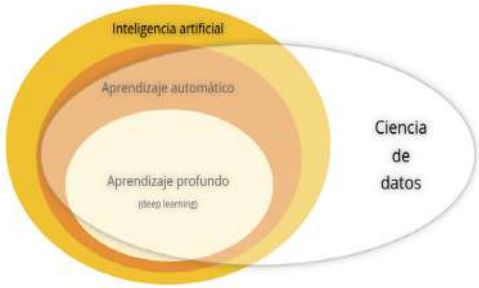


Figura 59: Relación entre inteligencia artificial y ciencia de datos.

Entonces, ¿cómo se relacionan la ciencia de datos y el aprendizaje automático? El aprendizaje automático es una parte integral de la ciencia de datos y una de las herramientas clave utilizadas para extraer conocimiento. La ciencia de datos proporciona el marco teórico y metodológico para abordar problemas complejos relacionados con los datos, y el aprendizaje automático ofrece las técnicas y algoritmos necesarios para analizar y extraer información útil de esos

datos. Un ejemplo claro de cómo se relacionan estas dos disciplinas sería el análisis predictivo:

> Supongamos que tenemos un conjunto de datos que contiene información demográfica y registros de compras de clientes. Usando técnicas de ciencia de datos, podemos explorar los datos, identificar variables relevantes y realizar un análisis estadístico para comprender el comportamiento de compra de los clientes.
>
> Una vez que tenemos una comprensión sólida de los datos, podemos aplicar técnicas de aprendizaje automático, como regresión[218] o clasificación, para desarrollar modelos. Estos modelos pueden predecir, por ejemplo, qué productos es más probable que compre un cliente en función de sus características demográficas y su historial de compras.

La ciencia de datos y el aprendizaje automático se complementan mutuamente en el proceso de análisis de datos y toma de decisiones. La ciencia de datos proporciona el marco teórico y las herramientas para abordar problemas complejos relacionados con los datos, mientras que el aprendizaje automático ofrece las técnicas y los algoritmos para extraer información útil y desarrollar modelos predictivos o descriptivos. Ambas disciplinas son fundamentales en el mundo actual, donde los datos son abundantes y la capacidad de extraer conocimientos valiosos de ellos es esencial para resolver problemas y tomar decisiones informadas.

14.3 Algoritmos en aprendizaje automático

En el contexto del Aprendizaje Automático, los algoritmos desempeñan un papel fundamental al permitir que las máquinas aprendan a partir de los datos y realicen tareas específicas sin ser explícitamente programadas. Estos algoritmos son responsables de tomar decisiones basadas en patrones y características presentes en los datos de entrada. A continuación, explicaremos de nuevo los diferentes tipos de algoritmos de aprendizaje automático [§2.1] junto con ejemplos ilustrativos, por medio de la clasificación más usada.

14.3.1 Algoritmos supervisados

Los algoritmos supervisados se basan en conjuntos de **datos etiquetados**, es decir, conjuntos de datos en los que se conoce la respuesta deseada. Estos algoritmos utilizan estas etiquetas para aprender a predecir la salida correcta para nuevas instancias de datos.

En la Figura 60 podemos ver parte de los datos originales de un *dataset* clásico, el conjunto de datos recopilados de la flor *Iris*, la cual se expresa como tres especies relacionadas: *iris setosa*, *iris versicolor* e *iris virginica*. Estas tres especies son apenas diferenciables a simple vista, así que en 1936 se recogieron[219] 150 muestras y, para cada una de ellas, se etiquetó con su especie corres-

218 *Me adelantaré un poco: la regresión se refiere a un tipo de algoritmo supervisado que tiene como objetivo predecir una variable continua o cuantitativa. A diferencia de la clasificación, que asigna etiquetas categóricas a las entradas, la regresión busca modelar y entender la relación entre variables para predecir un resultado numérico. En general, ambos, regresión y clasificación, buscan predecir algo partiendo de los datos.*

219 *Artículo original: https://onlinelibrary.wiley.com/doi/epdf/10.1111/j.1469-1809.1936.tb02137.x*

pondiente. Por lo tanto, cada muestra consta de 5 columnas: longitud y anchura del sépalo, longitud y anchura del pétalo, especie a la que pertenece (ver Figura 61).

Table I

Iris setosa				*Iris versicolor*				*Iris virginica*			
Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width
5·1	3·5	1·4	0·2	7·0	3·2	4·7	1·4	6·3	3·3	6·0	2·5
4·9	3·0	1·4	0·2	6·4	3·2	4·5	1·5	5·8	2·7	5·1	1·9
4·7	3·2	1·3	0·2	6·9	3·1	4·9	1·5	7·1	3·0	5·9	2·1
4·6	3·1	1·5	0·2	5·5	2·3	4·0	1·3	6·3	2·9	5·6	1·8
5·0	3·6	1·4	0·2	6·5	2·8	4·6	1·5	6·5	3·0	5·8	2·2
5·4	3·9	1·7	0·4	5·7	2·8	4·5	1·3	7·6	3·0	6·6	2·1
4·6	3·4	1·4	0·3	6·3	3·3	4·7	1·6	4·9	2·5	4·5	1·7
5·0	3·4	1·5	0·2	4·9	2·4	3·3	1·0	7·3	2·9	6·3	1·8
4·4	2·9	1·4	0·2	6·6	2·9	4·6	1·3	6·7	2·5	5·8	1·8

Figura 60: Parte de la tabla original del dataset clásico Iris.

Fuente: "Annals of eugenics", Volume 7, issue 2. Sep. 1936.

El problema que se busca solucionar es: ¿a partir de esos datos, es posible generalizar y aprender (esto es, crear un modelo) de forma que podamos asignar una especie a una flor sólo con recolectar sus datos de longitud y anchura de sépalo y pétalo?

Los algoritmos supervisados de aprendizaje automático, necesitan la solución (la etiqueta) para aprender la relación que hay entre las **columnas de características** (también llamadas variables independientes) y la **columna etiqueta** (también llamada variable dependiente, ver Figura 61).

sepal length (cm)	sepal width (cm) ▲	petal length (cm)	petal width (cm)	species
5.0	2.0	3.5	1.0	versicolor
6.2	2.2	4.5	1.5	versicolor
6.0	2.2	4.0	1.0	versicolor
6.0	2.2	5.0	1.5	virginica
6.3	2.3	4.4	1.3	versicolor
5.5	2.3	4.0	1.3	versicolor
5.0	2.3	3.3	1.0	versicolor
4.5	2.3	1.3	0.3	setosa
5.5	2.4	3.8	1.1	versicolor
5.5	2.4	3.7	1.0	versicolor
4.9	2.4	3.3	1.0	versicolor
6.7	2.5	5.8	1.8	virginica

Figura 61: Datos tabulados de dataset parcial Iris.

La idea en general es:

1. Entrenar con datos de la forma $X_1, X_2, \ldots, X_n \rightarrow Y$ y obtener un modelo con una tasa de error conocida y asumible.

2. Una vez tengamos un modelo, inferir a partir de nuevos X_1, X_2, ... su Y correspondiente.

Los algoritmos supervisados se pueden a su vez dividir en dos, según sea la naturaleza de su etiqueta, esto es, de lo que queremos predecir:

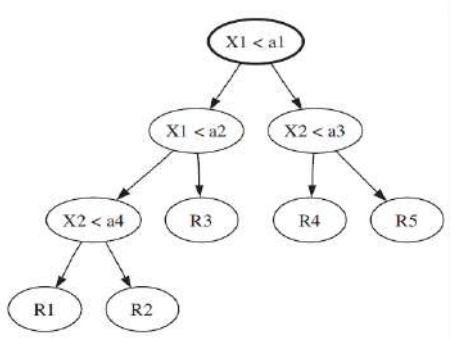
- Si es un número continuo, por tanto toma valores infinitos, estamos prediciendo un valor; por tanto estamos ante una **regresión**[220].
- Si la etiqueta toma valores finitos, como las especies *Iris*, estamos ante un problema de **clasificación**.

A continuación veremos algunos ejemplos de algoritmos supervisados.

Regresión lineal

La regresión lineal es un algoritmo utilizado para predecir una variable continua basada en la relación lineal entre las variables de entrada y la de salida. Por ejemplo, supongamos que queremos predecir el precio de una casa en función de su tamaño. Utilizando un conjunto de datos históricos que incluya el tamaño y el precio de diferentes casas, el algoritmo de regresión lineal aprenderá a trazar una línea que mejor se ajuste a los puntos y permita predecir el precio de una casa nueva en función de su tamaño.

Figura 62: Ejemplo de árbol de decisión



Árboles de decisión

Los árboles de decisión se utilizan para predecir una salida discreta[221] mediante una secuencia de preguntas basadas en las características de la entrada, formando así una especie de árbol invertido. Cada nodo del árbol representa una pregunta y las ramas corresponden a las posibles respuestas o nuevas preguntas.

Por ejemplo, supongamos que queremos clasificar correos electrónicos como "*spam*" o "*no spam*" en función de ciertas características, como la presencia de palabras clave específicas. Un árbol de decisión puede hacer preguntas como "¿El correo electrónico contiene la palabra 'oferta'?" y, en función de las respuestas (sí/no en este caso), llegar a una clasificación final (es o no es *spam*).

Regresión logística

La Regresión Logística es un algoritmo de aprendizaje supervisado utilizado principalmente para

220 *El término "regresión" proviene originalmente de estudios en genética y biología realizados por Francis Galton en el siglo XIX. Galton estaba investigando la relación entre padres e hijos en lo relativo a su altura. Observó un fenómeno que él llamó "regresión hacia la media", que significa que los hijos de padres con alturas extremas (tanto altos como bajos) tendían a tener alturas más cercanas a la media de la población. La "regresión" en el contexto que nos ocupa no implica necesariamente una "regresión hacia la media" como en el estudio original de Galton. En lugar de ello, se refiere a la tarea de predecir una variable continua. Este uso del término se ha vuelto estándar en estadísticas y aprendizaje automático.*

221 *También se pueden usar para regresión. Pero lo habitual es usarlos para clasificación.*

problemas de clasificación binaria (dos clases), aunque también se puede adaptar para clasificación multiclase[222]. Es una extensión del modelo de Regresión Lineal que utiliza la función logística (o *sigmoide §5.2*) para modelar la probabilidad de que una entrada pertenezca a una determinada clase. Este modelo es especialmente útil cuando se quiere estimar una probabilidad que puede traducirse en una decisión binaria (por ejemplo, si un correo electrónico es *spam* o no).

Figura 63: Ejemplo de aprendizaje automático a partir de datos usando árboles de decisión.
Fuente: www.saedsayad.com; autor: Dr. Saed Sayad.

Random forest (bosques aleatorios)

Los Bosques Aleatorios es un método de ensamblaje que combina múltiples árboles de decisión para crear un modelo más fuerte y robusto. Cada árbol se construye utilizando un subconjunto aleatorio de las características y un subconjunto aleatorio de las muestras del conjunto de datos. Las predicciones de los árboles individuales se combinan para dar una respuesta final, lo cual generalmente mejora la precisión y reduce el riesgo de sobreajuste. Este algoritmo se utiliza tanto para tareas de clasificación como de regresión y es conocido por su flexibilidad y robustez.

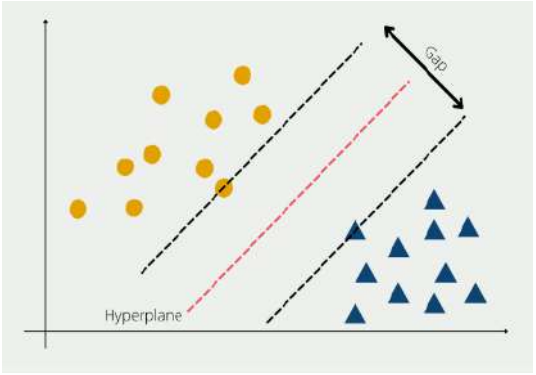


Figura 64: Explicación de SVM
Fuente: https://databasecamp.de/en/ml/svm-explained

Máquinas de soporte vectorial (SVM)

Las Máquinas de Soporte Vectorial son algoritmos que buscan encontrar un hiperplano (o conjunto de hiperplanos en espacios de alta dimensión) que mejor separe las diferentes clases de datos. Están diseñadas para clasificación y también se pueden utilizar para tareas de regresión. La principal ventaja de SVM es su eficacia en espacios de alta dimensión y su habilidad para encontrar límites de decisión complejos. Se pueden utilizar diferentes *kernels*[223] para transformar el espacio de características y encontrar un hiperplano que maximice el margen entre clases.

222 *Por ejemplo, el dataset Iris es un problema de clasificación de tres clases: satosa, virginica y versicolor.*

SVM es un gran clasificador, y su funcionamiento, como gran parte de los conceptos de estamos viendo, son muy simples [pero con un complejo desarrollo matemático].

Un clasificador binario aprende a partir de los datos de entrenamiento a asignar a cada entrada una etiqueta [en la Figura 64, la entrada posee 2 dimensiones y pretende clasificar cada punto como círculo o triángulo].

SVM calcula, a partir de los ejemplos con los que se le entrena, el mejor "pasillo" que separa ambos conjuntos de datos; de tal manera que *predice*, para nuevos datos, a que grupo pertenecerán. Si en vez de tener 2 dimensiones tenemos, por ejemplo, 1000; en vez de hablar de "pasillo" hablamos de hiperplano.

> *Un **hiperplano** es una superficie de dimensión n-1 en un espacio de n dimensiones que separa ese espacio en dos mitades. En caso de 2 dimensiones, una línea es un hiperplano. En 3 dimensiones sería un plano.*

Veamos el código que con SVM crea un modelo para predecir si un cáncer de mama es o no benigno, partiendo de unos datos de entrenamiento:

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

# Cargar el conjunto de datos de cáncer de mama
cancer_data = datasets.load_breast_cancer()
X = cancer_data.data
y = cancer_data.target

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)

# Crear el clasificador SVM
svm_classifier = SVC(kernel='linear', C=1)

# Entrenar el clasificador
svm_classifier.fit(X_train, y_train)
```

Posteriormente llevamos a cabo la predicción y y la evaluación del modelo:

```
# Realizar predicciones en el conjunto de prueba
y_pred = svm_classifier.predict(X_test)

# Evaluar el modelo
print("Matriz de confusión:")
print(confusion_matrix(y_test, y_pred))
print("\nInforme de clasificación:")
print(classification_report(y_test, y_pred))
print("\nExactitud:")
print(accuracy_score(y_test, y_pred))
```

223 *Técnica matemática que permite clasificar conjunto de datos – en principio – inclasificables, por no poder separarlos por un hiperplano.*

El resultado de este ejecución dos muestra una matriz de confusión e información sobre la precisión, la sensibilidad (*recall*) y un F1-score. Por ahora no importan estos datos, sólo saber que **hemos creado un modelo que con una exactitud del 96.5% es capaz de predecir si una biopsia muestra un cáncer de mama benigno o no** a partir de los datos de entrada[224].

En el código anterior tenemos una variable (objeto *svm_classifier*, en negrita) que apunta a un modelo que puede ser almacenado en disco y vuelto a cargar en memoria posteriormente, y que es capaz de predecir (función *svm_classifier.predict*) si una muestra obtenida puede ser o no cáncer de mama benigno con más de un 96% de exactitud.

14.3.2 Algoritmos no supervisados

A diferencia de los algoritmos supervisados, los algoritmos no supervisados se utilizan cuando no se dispone de etiquetas o respuestas previas. Estos algoritmos encuentran patrones y estructuras ocultas en los datos sin ninguna guía explícita. El objetivo es encontrar las relaciones intrínsecas que existen entre los datos.

A continuación, se presentan algunos ejemplos de tipos de algoritmos no supervisados:

Clustering (agrupamiento)

El *clustering* agrupa instancias de datos similares en grupos o clústeres. El objetivo es que las instancias dentro de un clúster sean similares entre sí y diferentes de las instancias en otros clústeres. Por ejemplo, si tenemos datos de clientes de una tienda en línea, el algoritmo de *clustering* puede agrupar a los clientes en diferentes segmentos basados en sus hábitos de compra o preferencias, permitiendo así a la empresa adaptar su estrategia de marketing para cada segmento.

Podemos tomar un dataset como *Iris* y quitarle la columna de especie (la columna objetivo); y ya tenemos un dataset susceptible de ser tratado con un algoritmo no supervisado (no tenemos la columna que nos dice de que especie es cada muestra). El objetivo es, a partir de las características, agruparlas de tal manera que se muestren las diferencias que permite reunir las muestras en grupos (ver Figura 65). La ventaja de esta aproximación es que posteriormente podemos recuperar la columna objetivo y comprobar que tal fue el agrupamiento. Sin embargo no siempre tendremos esa ventaja.

Reducción de dimensionalidad

La reducción de dimensionalidad se utiliza para reducir la cantidad de variables en un conjunto de datos, conservando la mayor cantidad posible de información relevante. Esto es especialmente útil cuando se trabaja con conjuntos de datos de alta dimensionalidad. Un ejemplo común de reducción de dimensionalidad es el Análisis de Componentes Principales (PCA), que transforma los datos originales en un nuevo conjunto de variables no correlacionadas llamadas componentes principales.

224 *El conjunto de datos de cáncer de mama en scikit-learn (Breast Cancer dataset) tiene 30 características numéricas (por tanto 30 dimensiones) que se calculan a partir de una imagen digitalizada de una biopsia de una masa mamaria. Estas características describen las cualidades del núcleo de las células presentes en la imagen y se dividen en tres grupos: las estadísticas del área, las estadísticas de la textura y las estadísticas del perímetro de la célula.*

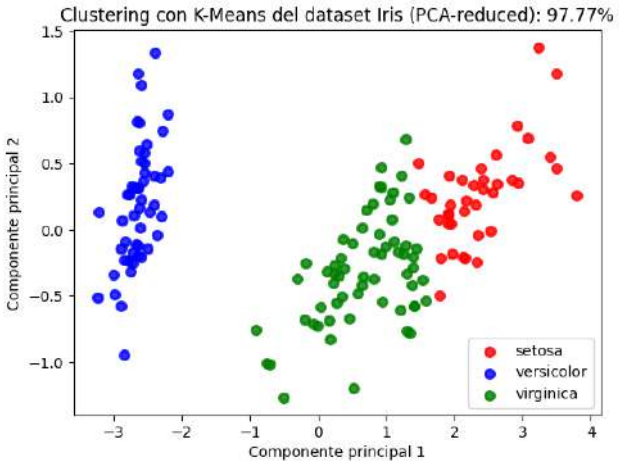
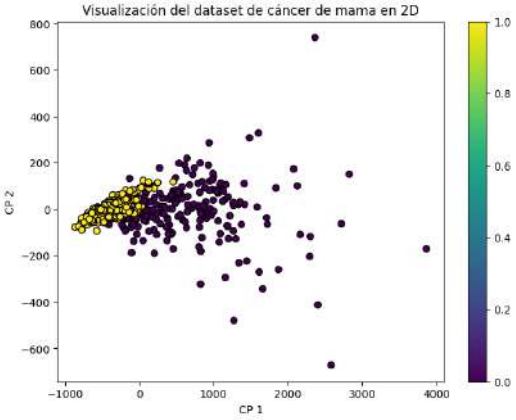
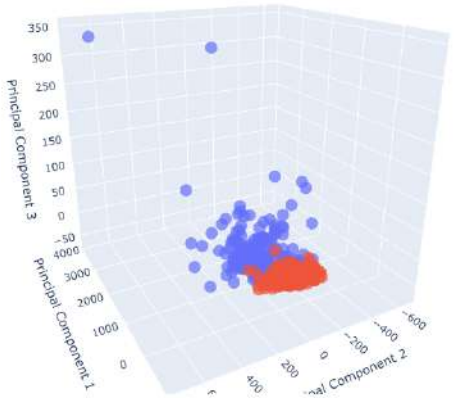


Figura 65: Representación en 2D de las variables independientes del dataset Iris.

Por ejemplo, el *dataset* que predecía si un cáncer de mama era benigno o no, posee 30 dimensiones, y visualizarlo por completo es imposible para nosotros. Pero podemos recurrir a este algoritmo para reducir la dimensionalidad, conociendo la cantidad de información que conservamos:



Visualización en 2D del dataset conservando el 99.82% de la información original



Visualización en 3D interactiva del dataset conservando el 99.98% de la información original

14.3.3 Algoritmos de Aprendizaje por Refuerzo

Los algoritmos por refuerzo son una forma de aprendizaje automático que permite a las máquinas (un agente) aprender a tomar decisiones de manera autónoma. Estos algoritmos simulan un entorno en el que el agente puede tomar acciones y recibir recompensas o castigos en función de sus resultados. A medida que este toma más decisiones/acciones, aprende cuales le dan más recompensas y se vuelve más hábil.

Imagina que estás jugando a un videojuego. Al principio, quizás no sepas muy bien qué hacer, cuáles botones apretar o cómo moverte para ganar puntos. Pero a medida que avanzas, empiezas a entender qué acciones te dan más puntos y cuáles te hacen perder. Eventualmente, quizá, te vuelves muy bueno en el juego porque has aprendido las mejores estrategias para ganar.

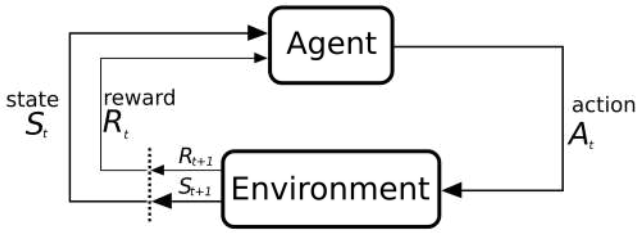


Figura 66: Esquema de un sistema de aprendizaje por refuerzo.
Fuente: Wikimedia commons

Los algoritmos de refuerzo funcionan de manera similar. Son una forma de aprendizaje automático en la que el agente aprende a tomar decisiones al interactuar con un entorno (que podría ser un juego, un laberinto, un mercado de acciones, etc.). Este recibe "recompensas" o "penalizaciones" en función de las acciones que toma, y con el tiempo, aprende a tomar las decisiones que maximizan su recompensa total. Es como entrenar a un perro con golosinas cuando hace algo bien, y diciéndole "no" cuando hace algo mal, hasta que aprende a comportarse de la manera que deseas.

14.4 Evaluación de modelos

Ponte en esta situación: has creado un modelo a partir de unos datos de entrenamiento y un algoritmo. ¿Cómo es de bueno tu modelo?

La evaluación del rendimiento de los modelos de aprendizaje automático es una etapa muy importante en donde se pone a prueba el modelo, con el objetivo de entender qué tan bien ha aprendido y cómo generaliza cuando se le presentan nuevos datos. Esta evaluación se lleva a cabo por medio de métricas. Estas métricas te permiten seleccionar el mejor modelo entre varios candidatos[225]. Aquí hay algunas técnicas y métricas comunes para tareas de clasificación binaria[226]:

- **Precisión**: indica el porcentaje de verdaderos positivos entre todas las predicciones positivas (verdaderas y falsas) realizadas por el modelo.

$$P = \frac{Verdaderos\,positivos}{Verdaderos\,positivos + Falsos\,positivos}$$

225 *Pues sí. Lo normal es crear varios modelos (quizá partiendo de varios algoritmos y/o modificando los hiperparámetros), seleccionando el que mejor ha aprendido. Es como escoger a la persona adecuada para un punto de trabajo por medio de un examen de conocimientos y aptitudes.*

226 *En el Capítulo 16 veremos la evaluación de modelos con más profundidad.*

- **Sensibilidad** [*recall*]: es la proporción de ejemplos positivos que fueron identificados correctamente por el clasificador respecto al total de positivos reales [verdaderos positivos y falsos negativos].

$$R = \frac{Verdaderos\,positivos}{Verdaderos\,positivos + Falsos\,negativos}$$

- **F1-score**: Combina la precisión y la sensibilidad en una única métrica, proporcionando un equilibrio entre ambas. Se calcula como la media armónica de ambas.

> ***Nota**: En términos simples, la precisión nos indica si hemos clasificado erróneamente instancias positivas como negativas, mientras que la sensibilidad nos indica si hemos perdido instancias positivas.*

Y para las tareas de regresión:

- **Error absoluto medio** [MAE]: Es la media del valor absoluto de los errores.
- **Error cuadrático medio** [MSE]: Es la media del cuadrado de los errores.
- **Raíz del error cuadrático medio** [RMSE]: Es la raíz cuadrada del MSE.

Otras técnicas relacionadas con la evaluación de manera robusta y sistemática del rendimiento de los modelos, son la validación cruzada y la división del conjuntos de datos.

- **La validación cruzada**: consiste en dividir el conjunto de datos de entrenamiento en varios subconjuntos, y luego entrenar el modelo en algunos de los subconjuntos y evaluarlo en los otros. Esto permite estimar el rendimiento del modelo con datos nuevos, que es lo que realmente importa en el mundo real, y durante la etapa de entrenamiento.

  Hay varios tipos diferentes de validación cruzada. El más común es la validación cruzada *k-fold*, que divide el conjunto de datos en *k* subconjuntos. Luego, el modelo se entrena en *k-1* subconjuntos y se evalúa en el subconjunto restante. Esto se repite *k* veces, y los resultados se promedian para obtener una estimación del rendimiento del modelo.

- **La división de conjuntos de datos**: como indica su nombre implica dividir el conjunto de datos en un conjunto de entrenamiento [*trainset*] y un conjunto de prueba [*testset*], donde el primero se utiliza para entrenar el modelo y el segundo para evaluar su rendimiento una vez que se ha acabado de entrenar.

  > ***Atención**: Esta división en trainset y testset es de vital importancia. Los modelos deben ser entrenados con el primero y evaluados con el segundo. Entre ambos conjuntos no puede haber elementos comunes, esto es, deben ser conjuntos disjuntos.*

  ¿Por qué? Veamos algunas razones:

- **Generalización**: El objetivo principal de un modelo de aprendizaje automático es generalizar/aprender bien con datos nuevos que no han visto antes. Si se utilizan los mismos datos para el entrenamiento [*trainset*] y la prueba [*testset*], no es posible evaluar la capacidad del modelo para generalizar ante nuevas situaciones.

- **Sobreajuste** [o*verfitting*]: Cuando un modelo se entrena en un conjunto de datos y también se evalúa en el mismo conjunto, hay un riesgo elevado de sobreajuste. El modelo puede "memorizar" los datos en lugar de "aprender" las características subyacentes, lo que significa que funcionará mal en nuevos datos [y no nos daremos cuenta de ello].

- **Subajuste** [*underfitting*]: Al igual que el sobreajuste, no se podrá detectar el subajuste si se usa el mismo conjunto para entrenamiento y prueba. El subajuste se produce cuando el modelo no aprende lo suficiente de los datos de entrenamiento y muestra un rendimiento deficiente incluso en esos datos.

- **Validación del modelo**: Seleccionar un modelo implica comparar su rendimiento con otros modelos o configuraciones. Si se evalúan múltiples modelos en el mismo conjunto de datos en el que fueron entrenados, la comparación será sesgada y posiblemente incorrecta.

- **Estimación de errores realistas**: Para entender cómo funcionará el modelo en un entorno de producción, se necesita una estimación precisa del error. Usar un conjunto de prueba separado que el modelo no haya visto durante el entrenamiento proporciona una medida más realista del rendimiento del modelo.

- **Confianza en el modelo**: Saber que un modelo ha sido probado en un conjunto de datos separado y ha demostrado un buen rendimiento aumenta la confianza en las predicciones del modelo para futuros datos no vistos.

14.5 Selección y optimización de hiperparámetros

Antes del entrenamiento de un modelo por medio de un conjunto de datos y un algoritmo, se han de tomar una serie de decisiones que afectarán al propio modelo y a su rendimiento. La primera decisión es la respuesta a ¿qué algoritmo usar? Lo normal es probar varios. Pero para cada uno de ellos deberemos determinar una serie de parámetros, muy dependientes del algoritmo, que influirán en el entrenamiento; no solo el precisión/sensibilidad o errores, sino también en tiempo de entrenamiento [minutos, horas o días].

> *Los **hiperparámetros** son parámetros configurables que no se aprenden directamente del conjunto de datos y afectan el rendimiento y la capacidad de generalización y aprendizaje del modelo.*

Ejemplos de hiperparámetros incluyen la tasa de aprendizaje, el número de capas ocultas en una

red neuronal, el número de agrupaciones, etc. entre otros muchos. La elección adecuada de los hiperparámetros puede mejorar significativamente la calidad del modelo y el tiempo de ejecución del entrenamiento.

¿Qué valores deben tomar los hiperparámetros? Los algoritmos asignan unos valores para los hiperparámetros por defecto, los más habituales. Quien cree un modelo a partir de unos datos de entrenamiento y un algoritmo, podría prescindir de otorgar valores concretos a los hiperparámetros, dejando que el algoritmo use los que *vienen de fábrica*. Pero si queremos afinar el entrenamiento o mejorar los resultados, debemos tenerlos en cuenta.

Pero no estamos solos para llevar a cabo esta tarea; existen diversas técnicas para seleccionar y optimizar los hiperparámetros de los algoritmos. Algunas son:

- **Búsqueda en cuadrícula**: Consiste en definir un conjunto de valores posibles para cada hiperparámetro (un mínimo/máximo, una lista de posibles valores) y evaluar el rendimiento del modelo para cada combinación posible. Esta estrategias, de hecho, crea multitud de modelos, que entrena y evalúa. Al final entrega la combinación de hiperparámetros que optimiza una métrica indicada[227].

- **Búsqueda aleatoria**: Implica seleccionar valores de hiperparámetros de forma aleatoria y evaluar el rendimiento del modelo para cada combinación. La búsqueda en cuadrícula puede generar muchos modelos y agotar el tiempo del que disponemos. En este caso esta estrategia confía en la suerte, llevando a cabo muestreos aleatorios de combinaciones de hiperparámetros. De esta manera podemos poner un límite al tiempo y los recursos empleados.

- **Optimización estocásticas**: Utiliza métodos basados en la teoría de probabilidad para encontrar la combinación óptima de hiperparámetros.

Ejemplo: Supongamos que estamos trabajando en un proyecto de detección de *spam* en correos electrónicos. Para seleccionar el algoritmo y modelo adecuados, consideraríamos la naturaleza del problema, que en este caso sería un problema de clasificación supervisada. Luego, evaluaremos el tamaño y la calidad de los datos disponibles, asegurándonos de que tengamos suficientes ejemplos de correos electrónicos etiquetados como *spam* y no *spam*. Además, consideraríamos la interpretabilidad y el rendimiento, ya que en este caso, es importante que el modelo pueda ser interpretado para comprender cómo se identifican los correos electrónicos de *spam*. Finalmente, utilizamos métricas de evaluación como precisión, sensibilidad y F1 para medir el rendimiento del modelo y optimizaremos los hiperparámetros, como el umbral de decisión para clasificar un correo electrónico como *spam* o no *spam*. Esto podría lograrse mediante técnicas como la búsqueda en cuadrícula.

227 *Por ejemplo, podemos indicar que priorice la sensibilidad sobre la precisión, si los casos de falsos negativos son importantes. En el caso del dataset del cáncer de mama, un falso negativo es dar un diagnóstico consistente en que se tiene un cáncer benigno cuando realmente es maligno. Ya veremos esto.*

RETOS DEL CAPITULO 14

1. Investiga y describe qué es el aprendizaje automático y cómo se utiliza en la vida cotidiana.
2. ¿Cuáles son las diferencias entre el aprendizaje supervisado y el no supervisado? Busca un tipo de aprendizaje denominado auto-supervisado.
3. Investiga y comparte ejemplos de aplicaciones del aprendizaje automático en la medicina. Busca en la *web* y da preferencia a artículos científicos [fíjate sólo en el título y en el *abstract* del mismo].
4. ¿Qué son los algoritmos de clasificación en el aprendizaje automático? Explícalo en términos sencillos.
5. Investiga y comparte ejemplos de cómo se utiliza el aprendizaje automático en la publicidad en línea. Busca en la *web* y da preferencia a artículos científicos [fíjate sólo en el título y en el *abstract* del mismo].
6. En grupo: reflexionar y describir el concepto de "*entrenamiento*" en el contexto del aprendizaje automático. ¿Qué es entrenar un modelo? ¿Por qué no se dice entrenar un algoritmo?
7. Pregúntale a tu IA favorita, qué es un algoritmos de recomendación y que se puede hacer con modelos creados para tal fin.
8. Investiga más en profundidad el por qué se llama regresión a la predicción de valores numéricos. Usa la web, no una inteligencia artificial.
9. Reflexiona: ¿Qué son los datos etiquetados y no etiquetados en el aprendizaje automático? ¿Por qué hacemos esa diferenciación? Explícalo en términos simples.
10. Reflexiona: ¿Por qué no nos interesa caer en el *overfitting* al entrenar un modelo? ¿Por qué aceptamos cierto grado de error, aunque no lo busquemos? ¿podríamos conseguir algún día modelos 100% precisos?
11. ¿Por qué dividimos el conjunto de datos en dos, *trainset* y *testset*? ¿Qué relación tiene con el *overfitting*?
12. Dedúcelo tu mismo: ¿Qué es el *underfitting*?
13. Conversa con tu IA favorita: Quieres crear un modelo de predicción del tiempo, dialoga con ella para enumerar los pasos que necesitarías para conseguirlo.
14. En el capítulos hemos visto el código que entrena y evalúa un modelo SVM sobre un dataset que contiene datos sobre cáncer de mama. Crea un *notebook* que lleve a cabo este objetivo usando la IA incorporada en Colab o con otra IA externa.