

APRENDIZAJE AUTOMÁTICO (MACHINE
LEARNING)
Y
CIENCIA DE DATOS (DATA SCIENCE)

UPV (VALENCIA)

Benito J. Palacios Cerro

Diciembre, 24th 2023

Contents

Contents	1
1 Módulo 1: Introducción a la minería de datos y ciencia de datos	3
1 Motivación	3
2 Minería de Datos y Ciencia de Datos	5
2.1 Agente bancario	6
2.2 Gerente supermecado	7
2.3 Gerente de personal	8
2.4 Supervisor de fábrica	9
2.5 Papel de la inteligencia empresarial	10
2.6 Minería de datos	12
2.7 Almacenamiento de datos multidimensional	13
2.8 OLAP vs Minería de Datos	17
2.9 Ejemplos de Ciencia de Datos	19
3 Proceso de extracción de conocimiento	21
3.1 Estándar CRISP-DM	23
4 Tareas, técnicas y herramientas	25
4.1 Tareas	25
4.2 Técnicas	27
4.3 Herramientas	32

2	Módulo 2: Evaluación de modelos de aprendizaje automático	34
1	Métricas de Clasificación	34
2	Métricas para regresión	36
2.1	Métricas para Aprendizaje No Supervisado	38
2.2	Sobreajuste	39
2.3	Validación Cruzada	42
2.4	Clasificación binaria: Matriz de confusión	45
2.5	Datos no balanceados	48
2.6	Clasificadores duros y suaves	49
2.7	Clasificadores probabilísticos	51
2.8	Rankers	53
2.9	Evaluación sensible al coste	54
	Appendices	55
A	Introducción suave al método Bootstrap (Jason Brownlee)	56

Chapter 1

Módulo 1: Introducción a la minería de datos y ciencia de datos

1

Motivación

- Motivación
 - ¿Qué es esto? Sopa de letras.
- Minería de datos y ciencia de datos
 - Ejemplos de minería de datos
 - El papel en la inteligencia empresarial
 - Almacenaje de datos y OLAP
 - Ejemplos de ciencia de datos
- El proceso de extracción de conocimiento
 - El proceso D2K
 - CRISP-DM
- Tareas, técnicas y herramientas

En general de lo que se trata es de la **transformación de datos en conocimiento**, partimos de datos que pueden estar en distintos formatos, el objetivo principal es que por medio de ciertos procedimientos convertir esos datos en conocimiento. Este conocimiento es el que nos puede servir en las organizaciones, lo que referimos como *Inteligencia Empresarial*, que puede ser útil para determinar que un intruso que entra en una red o en

un entorno físico, se considere como peligroso o no, y actuar en función del conocimiento o de los modelos que se extraen de los datos.

Ahora veamos definiciones:

Minería de Datos , está asociada a herramientas e inteligencia empresarial.

Ciencia de Datos , incluye cualquier acto, procedimiento o utilidad que hagamos sobre los datos.

Análisis de Datos (Inteligente) , proveniente de la estadística, se restringe a la parte analítica de los datos.

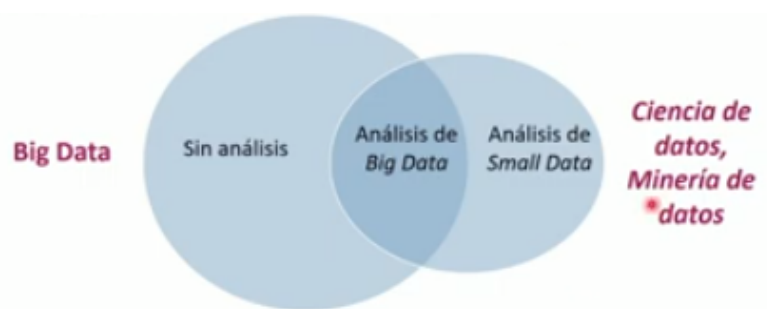
Análisis predictivo , es un tipo de *análisis de los datos*, existen más tipos de análisis como los *descriptivos* o la simple visualización de éstos.

Big Data , incluye una serie de aspectos que no son de análisis sino más bien de gestión de la información de grandes volúmenes de datos y a grandes velocidades.

Extracción de conocimiento, desde bases de datos , es la base de la transformación de los datos a conocimiento.

El **aprendizaje automático** aparece en todas las áreas que hemos visto como una herramienta muy flexible a la hora de extraer conocimiento a partir de los datos.

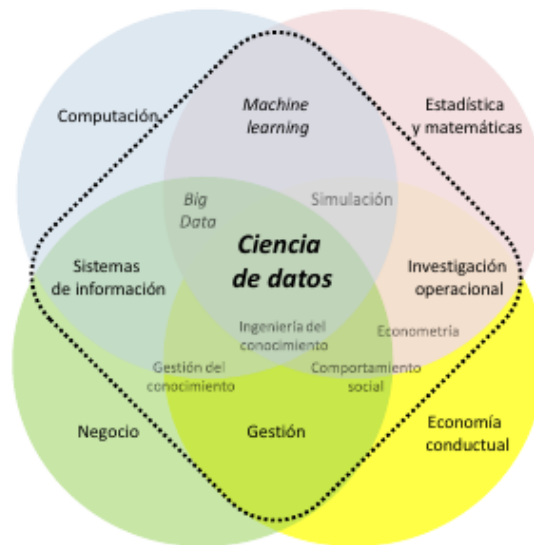
Para aclarar la diferencia entre *Big Data* y *Ciencia de Datos*, es que **no** todo lo que se hace en ciencia de datos tiene que ser con grandes volúmenes de datos, a gran velocidad o con gran variabilidad.



- Podemos tener un conjunto de datos pequeños, por ejemplo, pruebas diagnósticas de un número reducido de pacientes, y analizar esa información. Formaría parte del *Análisis de Small Data*.
- Podemos tener millones de registros de pacientes con medicamentos y tratamientos, de los que queremos analizar esos datos, hablaríamos de *Análisis de Big Data*.

- Por último, podemos tener los datos de todas las compras que se han realizado en un país durante unos cuantos meses o años y visualizar esos datos o resumirlos o hacer un informe y tener una gran complejidad a la hora de integrar esos datos o transformarlos, pero sin un análisis, hablaríamos de *Big Data*.

Es importante ver una serie de términos que están relacionados cuando hablamos de ciencia de datos y el papel del machine learning junto a la ciencia de datos.



2

Minería de Datos y Ciencia de Datos

Para entender lo que es la *ciencia de datos*, centrándonos particularmente en la *minería de datos*, la transformación de datos a conocimientos, vamos a ver cuatro ejemplos:

- Agente bancario. ¿Debo ofrecerle una hipoteca al cliente?. Veremos que con determinadas herramientas, que extraen conocimiento, modelos a partir de datos podremos resolver la pregunta.
- Gerente de supermercado. ¿Cuando los clientes compran huevos, también comprarán aceites?. Aquí veremos que podemos utilizar herramientas, de minería de datos, no tanto de análisis descriptivo o de aprendizaje automático.
- Gerente de personal. ¿Qué tipo de empleados tengo?.
- Supervisor de fábrica. ¿Cuántos fallos, para el módulo *X* esperamos cada mes?, o ¿Cuántas ventas vamos a hacer?.

2.1 Agente bancario

Cuando un cliente solicita una hipoteca o un préstamo de autoconsumo, la respuesta debe ser si se le concede o no. Para responder a esta situación, lo que le interesa al banco es saber si *los clientes anteriores han sido buenos clientes o malos para el banco*.

• AGENTE BANCARIO:

¿Debo ofrecerle una hipoteca a este cliente?

Histórico:

cld	Crédito-p (años)	Crédito-a (euros)	Salario (euros)	Tiene casa	Cuentas adeudadas	...	Devolución crédito
101	15	60.000	2.200	sí	2	...	no
102	2	30.000	3.500	sí	0	...	sí
103	9	9.000	1.700	sí	1	...	no
104	15	18.000	1.900	no	0	...	sí
105	10	24.000	2.100	no	0	...	no
...

Minería de datos

Patrón / Modelo:

```
If Cuentas adeudadas > 0 then Devolución crédito = no  
If Cuentas adeudadas = 0 and [(Salario > 2.500) or (Crédito-p > 10)] then Devolución crédito = sí
```

Un cliente que no devuelve un crédito es un *mal cliente para el banco*, pero puede haber otros conocimientos, como por ejemplo, un mes no se paga la mensualidad, no es un problema muy grande si después el cliente se recupera y puede ser incluso beneficioso para el banco. Lo que debe tener registrado el banco no es sólo si se devolvió o no el crédito, sino si realmente ha sido beneficioso o no para el banco.

Sit tuviéramos miles y miles de clientes anteriores a los que hemos concedido un préstamo, podríamos saber si al final, y al cabo de un cierto tiempo, esa persona ha tendido problemas o no a la hora de devolver el crédito.

Cuando un cliente solicita un crédito suelen hacérsele una serie de preguntas: cuanto tiempo durará el crédito, cuanta cantidad se solicita, que ingresos mensuales tiene, así como otro tipo de preguntas sobre su entorno familiar o si tiene otro tipo de propiedades que pudieran utilizarse como aval. La table mostrada está bastante simplificada, en relación al tipo de preguntas que se se pueden realizar.

Toda esta información se registra en una base de datos relacional. Ahora recibimos un nuevo cliente, que nos pide unos años, nos indica el dinero que quiere pedir y le hacemos otra serie de preguntas, lo que no sabemos es qué le responderemos. ¿Cómo podemos responder a esta petición?. Para ello lo que haremos será transformar los datos de la tabla en un modelo que a partir de las entradas del nuevo cliente nos de la predicción de si debemos conceder o no el crédito.

En este caso estamos realizando una **tarea predictiva**, y en este caso de **clasificación**,

donde los datos de entrada se corresponden con las conlumnas de la tabla, salvo la última columna, «devolución crédito», que se corresponde con nuestra variable de salida o predicción.

Como ejemplo, el modelo nos indica las siguientes reglas:

- que si las cuentas adeudar son mayores que cero, entonces, no devolverá el crédito
- si las cuentas adeudadas son igual a cero y el salario es mayor de 2.500€ o los años son mayores que diez, entonces, normalmente se devuelve el crédito.

Se trataría de un modelo muy simplista, pero nos da una solución al problema.

2.2 Gerente supermercado

Es un caso muy general en el ámbito del análisis de datos, puede ocurrir en cualquier tienda online o cualquier medio por el que se compran productos conjuntamente.

• GERENTE DE SUPERMERCADO:

Cuando mis clientes compran huevos, ¿cogen también aceite?

Histórico:

Nº cesta	Huevos	Aceite	Pañales	Vino	Leche	Mantequilla	Salmón	Endivia	...
1	sí	sí	no	sí	no	sí	sí	sí	...
2	no	sí	no	no	sí	no	no	sí	...
3	no	no	sí	no	sí	no	no	no	...
4	no	sí	sí	no	sí	no	no	no	...
5	sí	sí	no	no	no	sí	no	sí	...
6	Sí	no	no	sí	sí	sí	sí	no	...
7	no	no	no	no	no	no	no	no	...
8	sí	sí	sí	sí	sí	sí	sí	no	...
...

Patrón / Modelo:

Minería de datos

Huevos → Aceite: Confianza = 75%, Apoyo = 37%

Uno puede preguntarse, ¿Qué productos se compran conjuntamente? o cuando uno sale con la cesta, qué suele acompañar a éstos productos. En principio, si son pocos productos, se puede observar directamente, pero si son muchísimos productos, como un supermercado que puede tener miles de productos en su stock, si queremos ver que productos se compran conjuntamente, se pueden buscar todas las compras hechas durante y una semana, que previamente se ha registrado en una tabla como la de la imagen, y comprobar si los «síes» de una columna están asociados con los «síes» de la otra columna.

En la table vemos que aceites y pañales no estarían demasiado asociados.

Lo que queremos es responder a la pregunta general ¿qué productos se compran conjuntamente? Esto nos obliga a ver todas las combinaciones posibles de pares entre los miles de productos; para ello podemos recurrir a *herramientas de minería de datos* que lo que hacen es extraer todas las asociaciones, relaciones entre atributos que son muy frecuentes.

Cuando ejecutamos el análisis de datos, en relación con la pregunta "*cuando los clientes compran huevos, ¿compran también aceite?*", obtendremos un patrón que dice que cuando se compran huevos, entonces se compra aceite y la confianza es un 75%, quiere decir que de 100 compras de huevos, en 75 ocasiones también se compra aceite. Aún así, puede que ambos productos estén muy relacionados pero sean poco frecuentes en conjunto.

En el caso visto, los atributos de la tabla son todos de entrada, no hay variables de salida, en este caso hemos obtenido un modelo que establece una regla o patrón que nos indica la relación entre dos productos.

2.3 Gerente de personal

En este caso, un gerente de personal se pregunta *¿qué empleados tengo?*. De los empleados a su cargo conoce algunos, pero dependiendo del número de empleados de la empresa es probable que desconozca a la gran mayoría. Necesito información, datos, para tomar decisiones respecto a la reestructuración de la empresa, si tengo que formarlos de alguna manera o cualquier otra actividad que les afecte.

• GERENTE DE PERSONAL:

¿Qué tipo de empleados tengo?

Histórico:

Id	Salario	Casado	Coche	Niños	Alquilado/ Propietario	Sindicado	Bajas/año	Años de trabajo	Género
1	10000	si	no	0	alquilado	no	7	15	M
2	20000	no	si	1	alquilado	si	3	3	F
3	15000	si	si	2	propietario	si	5	10	M
4	30000	si	si	1	alquilado	no	15	7	F
5	10000	si	si	0	propietario	si	1	6	M
6	40000	no	si	0	alquilado	si	3	16	F
7	25000	no	no	0	alquilado	si	0	8	M
8	20000	no	si	0	propietario	si	2	6	F
15	8000	no	si	0	alquilado	no	3	2	M
...

Patrón / Modelo:

Minería de datos

- **Grupo 1:** Sin niños y alquilados. Baja participación sindical. Muchos días de baja.
- **Grupo 2:** Sin niños y con coche. Alta participación sindical. Pocos días de baja. Más mujeres en casas alquiladas.
- **Group 3:** Con niños, casados, con coche. Mayormente hombres y propietarios de casa. Baja participación sindical.

Toda empresa suele tener una base de datos con la información importante relacionada con sus empleados, como puede ser: el salario, si tiene coche, si está casado, ... A partir de esa información, quiero poder conocer **qué tipologías o grupos de empleados tiene la empresa**.

A simple vista podríamos tener una idea general, pero la gradación que puede haber de unos empleados a otros, teniendo atributos similares, es tanta que nos perderíamos y no concretaríamos dichas tipologías. Para ayudarnos hay herramientas de **aprendizaje automático, de agrupamiento, de clustering** que nos permitirán, a partir de los datos de la tabla, todos los atributos son de entrada, desarrollan un algoritmo que nos generará tres grupos, los cuales agrupan registros que se parecen y difieren de los registros de otros grupos.

En el ejemplo vemos que los tres grupos se componen:

- **Grupo 1:** sin hijos, con vivienda en alquiler, baja participación sindical y con muchos días de baja.
- **Grupo 2:** sin hijos, que tienen coche, alta participación sindical, pocos días de baja y más mujeres que hombres en casas alquiladas.
- **Grupo 3:** con hijos, casados, con coche, mayoritariamente hombre y con vivienda en propiedad, baja participación sindical.

La herramienta crea los grupos, en función de nuestras necesidades.

2.4 Supervisor de fábrica

Aquí puede interesarnos contestar a la pregunta *¿Cuántos fallos para el módulo «X» esperamos cada mes?*. Disponemos de una serie de datos relativos a ese módulo, entre ellos como ha ido fallando.

• SUPERVISOR DE FÁBRICA:

¿Cuántos fallos para X módulo esperamos cada mes?

Histórico:

Módulo (F*)	Mes-12	...	Mes-4	Mes-3	Mes-2	Mes-1	Mes
X	20	...	52	14	139	74	?
Y	11	...	43	32	26	59	?
Z	50	...	61	14	5	28	?
W	3	...	21	27	1	49	?
R	14	...	27	2	25	12	?
...

Minería de datos

Patrón / Modelo:

Modelo lineal: Fallos de X para el siguiente mes:

$$FX(\text{Mes}) = 0.62 \cdot FX(\text{Mes-1}) + 0.33 \cdot FX(\text{Mes-2})_X + 0.12 \cdot FZ(\text{Mes-1}) - 0.05$$

Se trata de un problema más complejo que los anteriores ya que involucra el *tiempo*, los

otros ejemplos vistos no son atemporales, pero en éste tendremos información relativa a los fallos que cada módulo ha podido tener cada mes.

Muchas veces las predicciones, sobre todo relativas al tiempo, se basan en los valores, no sólo del tiempo inmediatamente anterior, sino de valores agregados en tiempos o en ciclos anuales, mensuales, ... A partir de aquí, con el procesamiento de los datos, indicándole qué queremos predecir, los fallos para el mes que viene, partimos de la información sobre fallos de los meses anteriores, información que puede depender del propio módulo, pero también de fallos de otros módulos, con lo que la información que podemos incluir para predecir el valor de X , puede depender de sus valores, pero también de la información de otros registros del resto de módulos.

En este caso, utilizando la tabla referida y mediante una herramienta de minería de datos, con una preparación concreta, procesaremos los atributos necesarios como entradas, para obtener la última columna que se corresponde con la salida tras el procesamiento.

En este caso, podríamos utilizar una función lineal como la que se indica a continuación, en concreto *regresión lineal*, que predice un valor numérico, los fallos de el módulo X :

$$FX(Mes) = 0.62 * FX(Mes_1) + 0.33 * FX(Mes_2) + 0.12 * FZ(Mes_1) - 0.05$$

- Mes_1 , mes anterior al que queremos predecir.
- Mes_2 , dos meses antes.
- FX referido a los fallos de otros módulos.
- Los coeficientes: 0.62, 0.33, 0.12 y 0.05, son proporcionados por la herramienta de minería utilizada.

El modelo es **predictivo**, pero en este caso es un **modelo de regresión**.

2.5 Papel de la inteligencia empresarial

Inteligencia Empresarial es un concepto que engloba una serie de herramientas que intenta extraer conocimiento y tomar decisiones a partir de la información. Son herramientas que se estructuran en varios niveles, se conoce como estructura piramidal, desde los datos más básicos hasta el conocimiento.



Un ejemplo sencillo sería que *quiero reservar un vuelo desde Valencia a Montevideo, con varias escalas*, lo que haré será mirar a través de una aplicación, que me solicitará el origen, el destino, los días, etc; esta aplicación lo que hará es localizar los vuelos que tiene en su base de datos que coincidan con los datos aportados y si hay espacio en ellos proporniendolos como rutas posibles. Una vez que quiero realizar la transacción, la reserva, la aplicación deberá comprobar en todo momento que sigue habiendo plazas disponibles, y finalizada la transacción, si había plaza, confirmar la transacción.

La decisión que toma el sistema **no es estratégica** es una decisión a nivel **operacional**, que los sistemas pueden tomar en todo momento basándose en información, que normalmente se recoge en bases de datos, que internamente se consultan y modifican a través de lenguajes de gestión de bases de datos, como SQL.

Todo el proceso visto constituiria la base de la piramide, la interacciones más sencillas entre un dispositivo o aplicación y una persona.

Por encima del **nivel operacional** hablaríamos del **nivel táctico**, en él podemos hacer consultas mucho más complejas, como el informe que llega a la mesa de algunas personas, que toman decisiones en la organización, como pueden ser las ventas de la última semana, mes o año, desglosados por zonas geográficas y por tipos de productos. Son informes que van más allá de lo que es el dato preciso o concreto, no es información hipotética, sino de hechos ocurridos y, a partir de los cuales, tomar decisiones. Esta información se puede obtener a partir de consultas complejas, mediante instrucciones de lenguaje de programación como *group by*, agrupamiento, información que suele acompañarse de gráficos.

La información obtenida a nivel táctico podemos transformarla en **conocimiento** y observar si las ventas han ido como queríamos o si hay algo sorprendente.

El **nivel de planificación**, está por encima del nivel táctico, e incluye decisiones más complejas, por ejemplo decidir si *tengo que hacer una oferta a un clinte o conceder un crédito a un usuario*, no son preguntas que se puedan obtener de forma directa de los registros de una base de datos, ya que lo que necesitamos es anticipar o entender el con-

junto de datos, de que disponemos, entender el histórico y, a partir de ahí, extrapolar (puede ser desde el punto de vista predictivo o de grupos o tipologías). Este tipo de decisiones a este nivel, si queremos automatizarlas, requieren de herramientas de *aprendizaje automático*, la minería de datos nos permite tener modelos descriptivos o predictivos que nos permitan tomar decisiones.

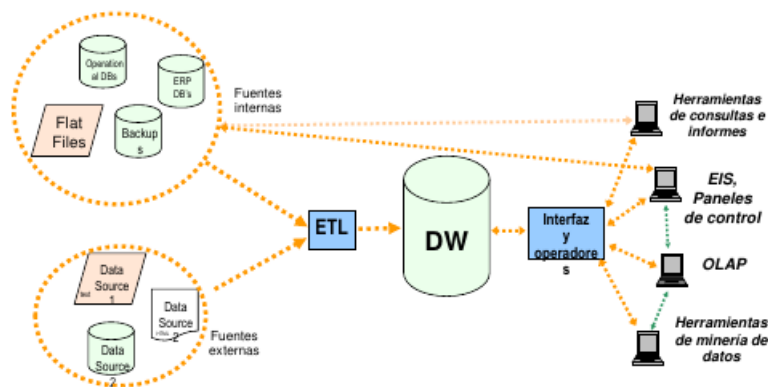
Vemos que a medida que ascendemos en la pirámide se requieren herramientas cada vez más sofisticadas.

La *inteligencia empresarial* es un término que engloba las herramientas de tecnología de la información y normalmente suelen estar compuesta de tres áreas:

- herramientas de almacenamiento de datos y OLAP, permiten manejar volúmenes de información y combinarlos, agregarlos y rotarlos.
- paneles de control y herramientas de supervisión, permiten a través de gráficas ver como va la organización en todo momento o unirlos a los informes.
- herramientas de minería de datos, permiten extraer de estos datos modelos con los que tomar decisiones; son modelos que pueden ser predictivos o descriptivos.

2.6 Minería de datos

Una estructura básica, sin entrar en detalles, de como se organiza la *inteligencia empresarial* sería la representada en el gráfico.



En primer lugar, al analizar datos e integrarlos para tomar decisiones estratégicas utilizaremos tanto las fuentes internas de la organización como las externas. Fuentes internas suelen ser el sistema de información de la organización, información que se vuelca en un *almacén de datos*, que generalmente está desconectado del trabajo diario de la organización. Normalmente, la fuentes internas, no permiten una visión completa del entorno de

la organización, en la organización funciona normalmente un entorno económico y social y se requieren datos e información del exterior, como pueden ser: tendencias del sector, qué eventos están ocurriendo. Temas como meteorología y calendarios, son fundamentales a la hora de hacer modelos predictivos; es información que podríamos necesitar integrarla, no es información necesaria desde el punto de vista operacional o trasaccional, pero es fundamental a la hora de extraer modelos que sean realmente predictivos y descriptivos sobre lo que ocurre alrededor de la organización.

Es una información, que como externa, puede estar en muchas fuentes diferentes, en bases de datos que uno compra o que están disponibles gratuitamente, de informes textuales, muchas veces en páginas web, ...

Como hemos dicho todo se integra en una base de datos o almacén de datos, que podemos ver que podemos organizarlos de diferentes maneras y, sobre esta información integrada, podemos aplicar una serie de herramientas con las que extraer informes.

Por último, tenemos *paneles de control* y *sistemas de información ejecutivos* que pueden hacer lo mismo. Las herramientas *OLAP* son herramientas que permiten, una vez realizada una consulta compleja que incluye la selección de varias dimensiones de los datos y agregadas a distintos niveles, entre otras:

- cambiar rápidamente entre niveles
- avanzar la consulta sin tener que esperar a que ésta acabe

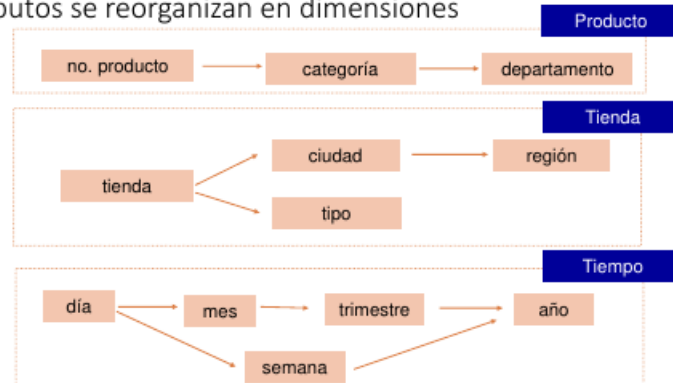
Las herramientas de minería de datos permiten a través de los datos históricos generar vistas *minables* y extraer modelos descriptivos y predictivos a partir de estas vistas.

2.7 Almacenamiento de datos multidimensional

Los almacenes de datos normalmente se organizan como una base de datos, integrando información de diferentes fuentes que posean información valiosa desde el punto de vista de responder a cuestiones de tipo estratégico, más que de tipo transaccional.

Almacenaje de datos multidimensional

- Los atributos se reorganizan en dimensiones



Para el trabajo transaccional utilizamos la base de datos de toda la vida, generalmente relacional.

Cuando el almacén de datos es sólo para realizar consultas, no habrá actualizaciones ya que estas normalmente van a la base de datos transaccional, ¿qué tipo de organización es más habitual?. Desde el punto de vista de la organización del almacén de datos tenemos, lo que se conoce como **modelo multidimensional**, es un modelo muy intuitivo ya que la información se suele dividir en dimensiones, que nos permiten agregar o desagregar la información a través de esas dimensiones.

Tomemos el ejemplo de un supermercado. En un supermercado puedo realizar las típicas preguntas o adverbios interrogativos:

- ¿Qué es lo que se vende en el supermercado?, hablamos del producto.
- ¿Dónde se vende?, nos aparece la dimensión tienda.
- ¿Cuándo se vendió?, nos aparece la dimensión temporal, cada producto se ha vendido en una tienda en un momento determinado.

Podemos hacer más preguntas como el tipo de pago, la hora, etc.

Nos vamos a centrar en qué, dónde y cuándo.

- El **qué**, hablamos del producto, cuando intentamos analizarlos vemos que se organizan jerárquicamente o en diferentes niveles. En general, del producto tenemos la siguiente información:
 - número del producto o código.
 - categorías.
 - departamentos.

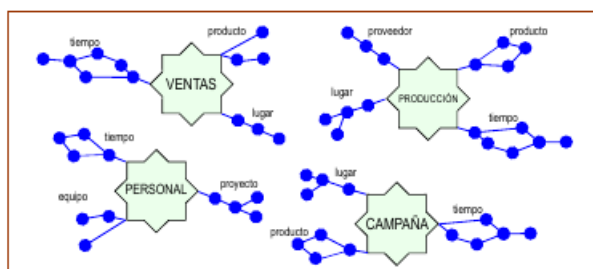
Puede haber más subdivisiones, aunque por simplicidad sólo mostramos tres.

- El **dónde**, referencia al punto de venta, una tienda concreta o venta online. En una tienda podríamos subdividir, incluso, por cajas. Podrían agregarse el tipo de tienda: minis, grandes o grandes superficies, ... Podemos agregar una dimensión *geográfica*: ciudad, región, país, ...
- El **cuándo**, el tiempo o momento en que se realizó la venta, bien el día, semana, mes, año, etc, incluso añadir más nivel de granularidad como la hora y minuto.

Este tipo de dimensiones y jerarquías de los niveles es parte del *diseño multidimensional* de este tipo de almacén de datos.

¿Cómo funcionan?. Hablabamos de ventas, para este caso sería la información que tendríamos sobre productos, lugar y tiempo.

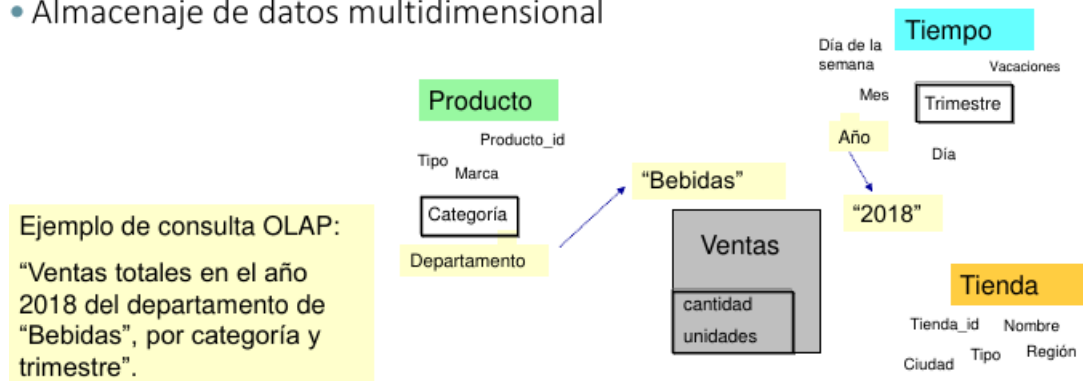
Cada diagrama se llama *datamart*:



Podríamos tener otros subalmacenes que normalmente se conocen con el nombre de **datamarts** en el que la información es de otro tipo. Para producción, podríamos tener información sobre campañas, que podrían ser independientes de la información de ventas. Cada una de ellas sería un *datamart* y cada uno de éstos tiene su estructura multidimensional.

Analicemos el *datamart de ventas*. Tenemos un ejemplo de una consulta, de momento ignoramos la parte de OLAP.

- Almacenaje de datos multidimensional



Quiero saber las ventas totales en el año 2018, del departamento de bebidas, por categorías y por trimestre.

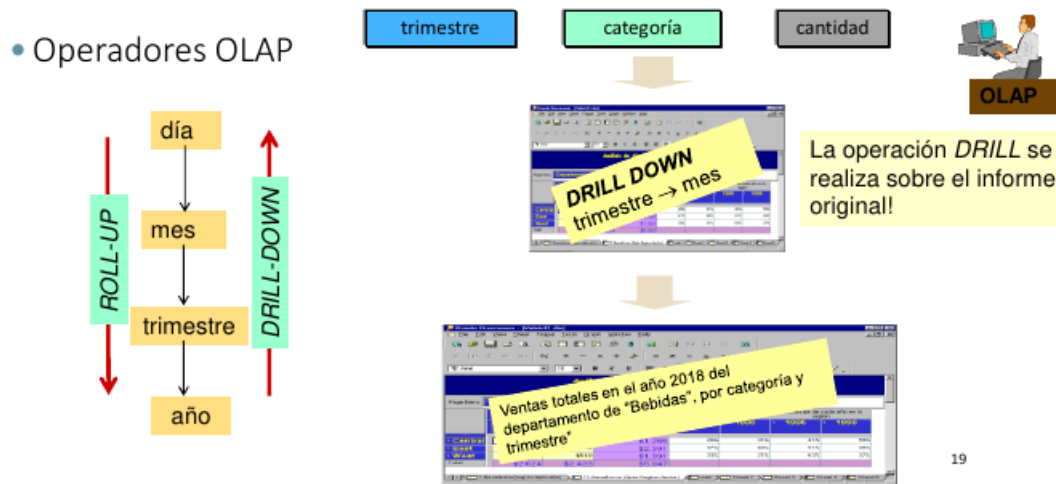
1. Nos centramos en el *datamart de ventas*, encontramos que hay una tabla central que tiene la información de los indicadores, en este caso cantidad(precio) y unidades.
2. Como cualquier pregunta suelen estar asociados con el qué, cuándo y cómo. En nuestro caso, el indicador son las ventas en el año 2018, tendremos que irnos al tiempo, en esta dimensión agregará *año*, en la dimensión de productos agregará la dimensión *departamento, bebidas*.
3. Por último, categoría y trimestre. Estos atributos no son selecciones sino agrupamientos.

Obtendríamos una tabla con la información seleccionada. Esto nos da un resultado que puede representarse de esta forma:



Podríamos ponerlo como una tabla en el que las categorías son las filas y los trimestres las columnas, y cada celda, las ventas totales.

Ahora podríamos querer cambiar algo en la consulta, para ello utilizaremos las *herramientas OLAP* (*OnLine Analytic Processing* o *Proceso analítico online*) para realizar un refinamiento de las consultas sin tener que lanzar una nueva consulta. En nuestro caso podríamos hacer, lo que se conoce con *drill down*, ir de trimestre a mes sin tener que realizar un informe nuevo, sino desde el informe original.



2.8 OLAP vs Minería de Datos

¿Es **OLAP** lo mismo que aprendizaje automático, que la minería de datos?. En principio, son cosas diferentes, aunque existe una delgada línea entre las preguntas que pueden responderse con herramientas de minería de datos o técnicas de aprendizaje automático y las que se pueden responder con herramientas *OLAP*, un almacén de datos y sin extrapolar.

La línea que separaría ambos ámbitos, es que *OLAP*, aunque sea de manera agregada, de manera compleja, todo el tipo de consultas estratégicas que hacemos sobre el almacén de datos no dejan de ser consultas sobre datos que existen en dicho almacén, la consulta no se inventa nada, no hace ninguna extrapolación ni hipótesis sobre los datos. Sin embargo, cuando hablamos de *minería de datos*, aquí existe un razonamiento hipotético, normalmente inductivo.

En la tabla tenemos preguntas que podemos clasificar en un de los dos ámbitos vistos.

OLAP

Minería

¿Cuál es el promedio de accidentes entre los fumadores y los no fumadores?	¿Cuáles son los mejores vaticinadores para los accidentes?
¿Cuál es el promedio de la factura de teléfono de mis actuales clientes vs. mis exclientes?	¿Dejará X la compañía? ¿Qué factores afectan a las dimisiones?
¿Cuál es el promedio de compras diarias entre los usuarios de tarjetas de crédito robadas y usuarios legítimos?	¿Qué patrones están asociados al uso de tarjetas de crédito fraudulentas?

Las preguntas que aparecen en la columna de *OLAP* vemos que son datos que se encuentran en el almacén de datos, y que lo que hacemos es lanzar consultas sobre datos reales recogidos en ellas, sin tener que elaborarlos. Por el contrario, en el lado de la *minería*, la información tienen que se interpretada, requiere la elaboración de un *modelo hipotético* que relacione las entradas, en el caso de la primera pregunta, las características de los clientes como: edad, sexo, años de antigüedad del carnet, tipo de comportamiento, dónde vive y demás, con respecto al número de accidentes que pueda tener; esto se hace a través de información histórica y no de información futura, que nos permite entender cuál es esta relación y describe lo ocurrido de forma abstracta.

En relación con la pregunta ¿qué patrones están asociados al uso de tarjetas de crédito fraudulentas?, la pregunta es bastante más amplia, aquí lo que podemos responder son cuestiones que son anómalas, por ejemplo, realizar dos o más compras de un tipo concreto, puede estar asociado a un uso fraudulento de las tarjetas, ya que normalmente son compras, por ejemplo, en joyería, alguien que nunca ha comprado este tipo de artículos y por cantidades elevadas, son pagos bastante inusuales para el perfil del individuo; aquí se recurriría a herramientas de aprendizaje automático principalmente, para extraer ese tipo de patrones.

De las siguientes afirmación, cuales pueden resolverse mediante herramientas **OLAP**:

- A) ¿Cuál es el producto más vendido entre el rango de edad 25-45?
- B) ¿Cuántos productos de la categoría «hogar» se espera vender el año que viene?
- C) ¿Qué diferencia de tamaño hay entre las cestas de la compra de los sábados y entre semana?
- D) ¿Qué días del mes suele haber más rotura de stock?
- E) ¿Qué tipo de productos tengo atendiendo a sus características: peso, precio, etc.?

Las respuestas correctas son A), C) y D).

2.9 Ejemplos de Ciencia de Datos

La *ciencia de datos* en un contexto de *inteligencia de negocio* es una actitud más que una disciplina desde el punto de vista de una organización. Cuando una organización contrata un científico de datos, lo que quiere es una persona con un rol muy particular.

Podemos pensar que un científico de datos es lo mismo que otros roles existentes en las organizaciones: director de sistemas de información o *data manager* o el director de datos o *data officer*. En empresas relacionadas con la tecnología, son roles muy importantes en la organización, son roles de primer nivel, como el caso de Google o Facebook, donde lo más importante son los datos. El científico de datos va en esa línea, debería ser una persona con un conjunto integrado de habilidades que abarcan:

- matemáticas
- aprendizaje automático
- inteligencia artificial
- estadística
- optimización de bases de datos

pero con un profundo conocimiento en la elaboración de problemas para el diseño de soluciones efectivas. Su función debe ser extraer valor de los datos y sacar valor a partir de ellos, haciendo que éstos sean el motor de la organización, no simplemente una herramienta que existe en la organización que está ahí para el funcionamiento de las aplicaciones, sino conseguir que esos datos sean los que proporcione valor y hagan que la organización funcione.

Nos podemos preguntar ¿dónde se originan esos datos? y ¿para quién queremos obtener ese valor?. Dependiendo de estas preguntas podemos clasificar distintos tipos de aplicaciones o problemas que podemos resolver mediante la ciencia de datos.

- Los datos que tengo son valiosos para mí ($in \rightarrow in$), prácticamente el 100% de las organizaciones que tengan datos pueden sacar rendimiento a partir de ellos, a parte de lo que pueda ser el sistema transaccional, pueden sacar valor, sacar patrones que ayuden a la hora de la toma de decisiones. Aunque hablamos de organizaciones es extensible a los individuos. Esto es lo que se ha conocido como *inteligencia empresarial clásica*.

Cuando hablamos de **datos internos de la organización** para dar servicio a la organización, un ejemplo habitual es el de una compañía de seguros de automóviles donde queremos predecir qué póliza va a comprar un cliente. Teniendo en cuenta su historial de transacciones, miro el almacén de datos y extraigo los datos que me

permitan establecer un *modelo predictivo* sobre qué póliza es la más adecuada. No necesita buscar datos fuera de la organización.

- Los datos que he visto exteriores a la organización, los podría utilizar para sacar patrones que me ayuden en mis decisiones ($out \rightarrow in$), dichas fuentes pueden ser redes sociales.

Como ejemplo, tenemos la *Comisión del Crimen de Detroit*, en su momento intentaron utilizar información sobre redes sociales para ver realmente dónde y en qué momentos se estaba produciendo tráfico de drogas u otro tipo de delincuencia. Es sorprendente, pero algunos delincuentes son capaces de decir lo que están haciendo en las redes sociales o de lo que van a hacer o dónde van a quedar, a partir de ahí la policía puede saber qué puntos calientes hay, e incluso poder abortar algunos de los posibles delitos. Son datos completamente externos que le permiten tomar decisiones estratégicas.

- Nosotros disponemos de datos, que son valiosos para nosotros, pero también podrían serlo para otras organizaciones externas ($in \rightarrow out$), como sacar esos datos es un servicio que los pone disponibles. Podemos vender los datos directamente o el «conocimiento», como se extrae dicho conocimiento.

Cuando hablamos de externos, por ejemplo una compañía de telefonía, sus clientes no son externos son internos, al hablar de externos nos referimos que somos conscientes de que tengo información sobre miles de usuarios, que son internos, y que esa información puede ser útil para terceros. Por ejemplo, tendría información de dónde se encuentran los usuarios en una ciudad, si tengo el 20% de cuota de teléfonos móviles en una localidad, puedo saber dónde están, no a nivel individual, esta información no se puede proporcionar a terceros por razones legales, pero sí a nivel *agregado*. Puedo saber que proporción de gente se mueve en una zona de la ciudad en un rango de horas. Si mi porcentaje es lo suficientemente representativo, podré extrapolar y saber los flujos de población en la ciudad. Esta información puede ser importante para ayuntamientos, comercios, transportes, etc. Evidentemente, el poseedor de la información sacará un beneficio de proporcionar este conocimiento.

- En ocasiones, yo no tengo los datos, ni siquiera soy el fin, soy un intermediario ($out \rightarrow out$), un experto en ciencia de datos que dispone de los datos y establece que si se extrajeran determinados patrones, los datos podrían ser útiles para una organización o usuarios. Muchos servicios hoy en día intentan hacer este tipo de cosas.

Cómo en casos anteriores, recurrimos a las redes sociales. La gente, con frecuencia, cuando se encuentran mal, enfermos, pueden indicar sus síntomas e incluso que acudieron al médico e indicar que pronóstico les dió en redes sociales. Se intenta hacer asociaciones entre síntomas, como se toman los medicamentos, etc. Es una información que es externa, y que estaba dirigida para otro tipo de uso.

- Finalmente, la visión más ambiciosa, no hay datos ($\emptyset \rightarrow out$), pero si los hubiera

habría valor, es decir puedo intentar crear aplicaciones cuyo objetivo final sea crear datos que den valor. Gran parte de la economía digital reciente se basa en esta idea. Si volvemos a las redes sociales, éstas a partir de la nada han creado datos, a partir de éstos conocen el perfil de la gente, sus aficiones, gustos, horas de contacto, amigos, familiares, es decir, no es que esa información existiera, pero mucha se crea a través de esas aplicaciones que se crean inicialmente de la nada.

Una de las aplicaciones más típicas, relacionadas con este punto de vista, es que instalamos una aplicación en nuestro dispositivo móvil que nos informa, a través de datos que recoge de otros usuarios de la misma aplicación. El ejemplo típico son aplicaciones de tráfico, nos permiten saber las rutas más cortas y menos congestionadas.

Otra forma de ver la ciencia de datos es en las áreas donde se aplica, aquí podemos ver algunas de ellas:

- | | |
|---|---|
| <ul style="list-style-type: none"> • Datos de telecomunicaciones <ul style="list-style-type: none"> ▪ Valioso para comerciantes, tráfico, ayuntamiento, policía... • Otros datos de geolocalización(Flickr, Instagram, Wikiloc, ...) <ul style="list-style-type: none"> ▪ Valioso para agencias de viaje... • Datos en consumo de energía <ul style="list-style-type: none"> ▪ Valioso para anuncios de televisión... • Datos del transporte público (bus, metro, tren, taxi, tráfico, ...) <ul style="list-style-type: none"> ▪ Valioso para turismo, consumo, contaminación, comercio... • Datos de redes sociales. <ul style="list-style-type: none"> ▪ Valioso para casi todo... | <ul style="list-style-type: none"> • Datos de uso de tarjetas de crédito <ul style="list-style-type: none"> ▪ Valioso para comercios, ayuntamientos, ... • Datos de policía <ul style="list-style-type: none"> ▪ Valioso para aseguradoras, agentes inmobiliarios, ... • Datos comerciales (Amazon, Ebay, segundamano.es, ...) <ul style="list-style-type: none"> ▪ Valioso para salud, demografía, sociología... • Datos climatológicos <ul style="list-style-type: none"> ▪ Valioso para comercios. • Datos de búsquedas web. <ul style="list-style-type: none"> ▪ Valioso para casi todo. |
|---|---|

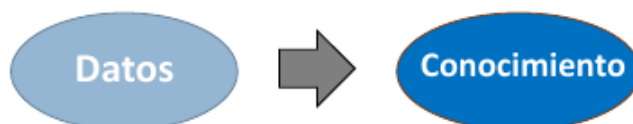
3

Proceso de extracción de conocimiento

Cuando nos centramos en la parte de la *minería de datos*, más predictiva o descriptiva, nos centramos en convertir datos en conocimiento, proceso que se conoce por sus siglas en inglés de **D2K (Data to Knowledge)**, de datos a conocimiento.

“Proceso no trivial de identificar datos válidos, novedosos, potencialmente útiles y comprensibles”.

(Fayyad et al. 1996)



Este proceso podemos traducirlo como extracción de conocimiento a partir de datos o bases de datos de cualquier tipo. Es un proceso que se conoce hace tiempo, no trivial de identificar patrones válidos, potencialmente útiles y comprensibles.

- Es un **proceso no trivial**, si fuera sencillo no serían necesarias herramientas, de conocimiento o reglas o técnicas para resolver el problema.
- **Identificar patrones**, son patrones que van más allá de los datos, son tendencias, reglas, grupos que no son sólo datos agregados; normalmente inferen otro tipo de información a partir de los datos ya existentes. Como son datos hipotéticos, que extrapolan a partir de otros datos, históricos, nos interesa que sean válidos, que los patrones tengan un porcentaje de error mínimo. La parte de evaluación es fundamental.
- **Novedoso**, por ejemplo que en una unidad de maternidad de un hospital, el porcentaje de ingresos de mujeres siempre será del 100%, este procedimiento no es novedoso. También se da el caso de ser novedoso pero inútil, puedo suponer que dos productos, champú y una marca de tomate se compran conjuntamente, ¿qué puedo hacer con esa información?, poner el champú cerca del tomate, probablemente será una curiosidad, pero difícilmente será útil.
- La información debe de ser **comprensible**, hablamos de técnicas de aprendizaje automático, redes neuronales o redes profundas, máquinas de vectores, que proporcionan información poco comprensible para un humano, frente a técnicas que producen reglas que sí son comprensibles por un humano, como pueden ser los árboles de decisión.

Este proceso normalmente se desglosa en fases más concretas, no sólo en cómo convertir datos en conocimiento



en la parte izquierda tenemos el conocimiento, el final aparece como decisiones, a partir de ese conocimiento, es lo que se denomina extracción del conocimiento a partir de datos.

El número de tareas o de fases puede variar, pero generalmente se empieza por una parte de integración de datos, con fuentes internas y externas con distintos formatos, que integraremos generalmente en un repositorio, que no tiene por qué ser multidimensional, pero sí un repositorio donde tenemos todos los datos juntos, integrados y consistentes. Aunque tengamos todo los datos almacenados, será necesario prepararlos para el modelo que queremos extraer. Este proceso de preparar para extraer conocimiento es lo que se conoce como *crear la vista minable*.

En la vista minable pongo una filas y columnas con la información que quiero, una vez que tengo todo esto puedo aplicar minería de datos, o aprendizaje automático, para obtener patrones, que normalmente tendré que evaluar, tendrán un error, fallarán con alguno de los ejemplos del histórico. Habrá que seleccionar aquel patrón que mejor se ajuste a los datos y que tenga un menor error.

Una vez realizado el proceso, dispondremos de un **modelo evaluado y validado**, y hablamos realmente de conocimiento, ya podremos utilizarlos y tomar decisiones.

Estos modelos en ocasiones quedarán obsoletos, las situaciones cambian o aparecen problemas inesperados; podemos ir hacia atrás, al inicio o a un reentrenamiento de los modelos dependiendo del problema que detectemos.

3.1 Estándar CRISP-DM

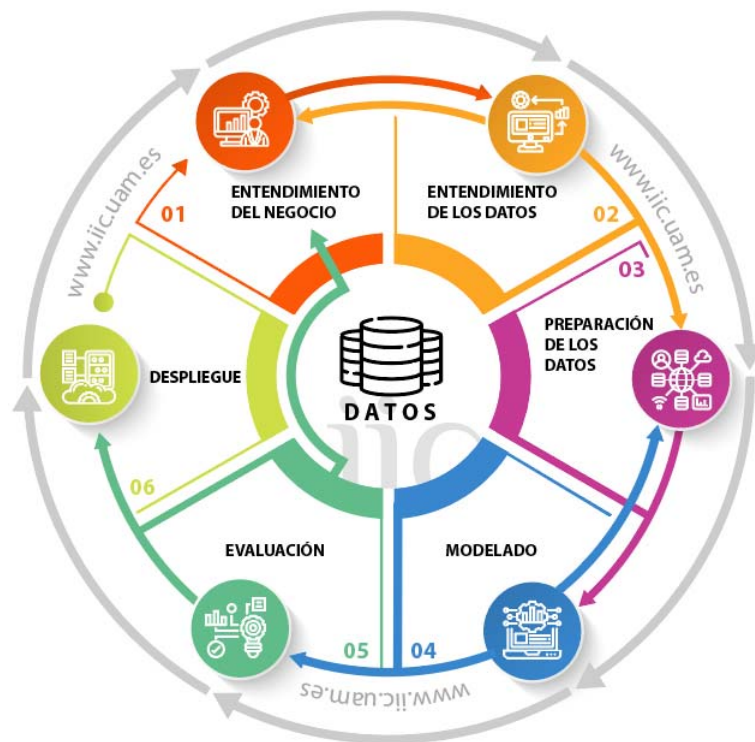
Hemos visto que el proceso de extracción de conocimiento se estructura en una serie de fases, que pueden variar dependiendo del contexto o de la organización, incluso podemos ver el proceso a partir de conocimiento muy simple.

Cuando vemos un proceso que tiene que seguir una serie de pasos aparecen estándares o metodologías que guían el proceso. Seguir estos estándares permite que diferentes usuarios, e incluso organizaciones, se entiendan a la hora de desarrollar un proyecto. Un proyecto ya finalizado, es más entendible si sigue una metodología.

Algo importante de las metodologías es que van acompañadas de documentación, que permite que los usuarios externos al proyecto puedan entender e integrarse en el proyecto más fácilmente. No es más que **gestión de proyectos** aplicado a ciencia de datos.

La metodología más común se conoce como **CRISP-DM**, apareció hace más de veinte años, como un proyecto de la comisión europea y de empresas, siendo una de las metodologías más utilizadas.

La metodología CRISP-DM se conceptualiza en 6 fases:



1. **Entendimiento del negocio**, el equipo de trabajo debe comprender los objetivos y requisitos del proyecto definidos por el cliente, para convertir el conocimiento en una definición técnica del problema. Requiere una comunicación intensa entre cliente y equipo técnico.
2. **Entendimiento o compresión de los datos**, el equipo técnico realiza un *análisis exploratorio* para obtener una visión general de lo que se puede conseguir con los datos. Tras el análisis debería tenerse una idea clara de la viabilidad del proyecto y de los resultados esperados, de ser así se avanza a las siguientes dos fase.
3. **Preparación de los datos**, cubre las actividades para construir el conjunto de datos definitivo que se empleará en la siguiente fase.
4. **Modelado de datos**, el equipo técnico realiza los análisis y modelos pertinentes de los que se deriven los resultados y conclusiones del proyecto, y su validación frente a errores.
5. **Evaluación**, el cliente determinará la calidad de los resultados obtenidos y decidirá cómo pueden explotarse antes del despliegue.
6. **Despliegue**, fase en la que el modelo se pone en producción, con el fin de tomar decisiones.

Se trata de un proceso de **carácter iterativo**. Aunque algunos consideran que esta metodología es un poco rígida, siempre puede ser utilizada como base y adaptarla a nuestras necesidades.

4

Tareas, técnicas y herramientas

4.1 Tareas

Algo muy importante es distinguir entre tareas, técnicas y herramientas. Nos vamos a centrar ahora en ¿qué son las tareas?.

Vamos a ver cinco grandes familias de tareas:

- Regresión
- Clustering
- Reglas de Asociación
- Análisis Factorial
- Dispersión o Multivariante

La primera pregunta que nos haremos, si vemos una tabla como la siguiente:

x1	x2	x3	x4	x5	x6	...	xn
20	315	High	1.9	Married	0.2	...	sí
135	310	Low	2.1	Single	0.3	...	no
...

si hay alguna/s variable/s de salida y/o de entrada. Las **variables de salida** representan lo que quiero **predecir** a partir de las **variables de entrada**. En el caso que tenemos, X_n es una *variable de salida*, y el resto de variables, X_1, X_2, \dots , serían *variables de entrada*. En el momento en que determinamos la existencia de una/s variable/s de salida, hemos establecido una **tarea predictiva**.

Por ejemplo, quiero saber si a un cliente hay que darle un préstamo, o una operación con tarjeta de crédito es fraudulenta o las ventas del próximo trimestre, etc. Todas son tareas que tienen un valor de salida a partir de valores de entrada.

Podemos distinguir dos tipos de *tareas predictivas*:

- **Clasificación/Categorización**. Tenemos una *variable de salida*, el resto son *variables de entrada*, el valor de la variable predictiva es **nominal** o **cualitativa**. Por

ejemplo, predecir si se concederá un crédito da como resultado una variable cualitativa o nominal, **sí o no**, hablamos de una *tarea de clasificación*.

- **Regresión.** Si la *variable de salida* es numérica. Por ejemplo, cuando queremos saber cuantos productos puedo vender en el trimestre.

Pero y cuando no hay ninguna **variable de salida**, hablaríamos de una **tarea descriptiva**, muchas de las tareas del *aprendizaje automático* son descriptivas, no predicen nada. Hay tareas *descriptivas* que tienen un valor a la hora de entender y extraer conocimiento a partir de los datos. Por ejemplo, si lo que quiero hacer es entender las filas o las columnas, tengo diferentes tareas descriptivas.

Los tipos de *tareas descriptivas* son.

- **Clustering**, si quiero entender la relación entre las filas de la tabla, relación entre individuos en este caso, si éstos fueran clientes puedo hacer un *agrupamiento de clientes*, *clustering de clientes*. Es una tarea cuyo objetivo es describir grupos de datos, que grupos de clientes tengo.
- **Análisis Exploratorio**, es el análisis de las relaciones entre columnas:
 - **Reglas de asociación, dependencias funcionales**, cuando las variables son **nominales, cualitativas**, no son números. Por ejemplo, las columnas que hacen referencia a *alto*, *bajo*, *casado*, *soltero*,... son nominales. Si el valor de *High* en la variable X_3 está asociado con un valor de *Married* en la variable X_5 , sólo podré verlo si hay muchísimos ejemplos, ver si esta relación es más frecuente que otras. Otro ejemplo sería el *análisis de la compra*, ver que productos se compran conjuntamente, ejemplo típico de las **reglas de asociación**.
 - **Análisis Factorial o Multivariante**, si las variables que queremos relacionar son numéricas. El concepto de *relación binaria*, cuando hablamos de este tipo de variables, se conoce tradicionalmente como **correlación**. Entre las variables X_4 y X_6 podría ver si estos valores están correlacionados, por ejemplo, si la edad y el sueldo están correlacionados. Es un ámbito inmenso, que restringimos normalmente a preguntas más básicas como calcular correlaciones y una matriz de correlaciones. Esta última clasificación, también la conocemos como **análisis factorial de correlaciones, análisis de dispersión o análisis multivariable**.

La pregunta es ¿qué ocurre si tengo varias variables numéricas y varias nominales?.

- Si tenemos *muchas numéricas y pocas nominales*, podemos *numerizar todas las variables* y realizar un **análisis de correlaciones**.

- Si tenemos *muchas nominales y pocas numéricas*, las numéricas podemos **discretizarlas** y realizar un **análisis de regla de asociación**.

Son tipos de variables que no podemos analizar al mismo tiempo, pero podemos convertir unas en otras y aplicar el análisis que más convenga.

Es una manera simple, pero elegante de entender qué tipos de **tareas** podemos tener en *aprendizaje automático*, las diferencias fundamentales son:

- **predictivas**, qué tipo de variable tenemos que predecir, numérica o nominal.
- **descriptivas**, queremos analizar las relaciones entre filas, *agrupamientos*, o analizar relaciones entre columnas, *análisis exploratorio*, si los valores son numéricos, *correlaciones*, si son nominales, *asociaciones*.

¿Qué tipo de tarea es «*diagnostiscar la presencia de una enfermedad dados unos síntomas*»?:

- A) Clasificación
- B) Regresión
- C) Asociación
- D) Clustering

Solución: A), clasificación.

4.2 Técnicas

En un ámbito general, las tareas de minería de datos o en ciencia de datos (aprendizaje automático), se dividían en cinco clases principales, dependiendo de si eran:

- **supervisadas**,
- **no supervisadas**,
- **predictivas** o
- **no predictivas**

Es habitual confundir *tareas* con *técnicas* ya que muchas o algunas de las técnicas sólo resuelven un subconjunto o incluso una sola de las tareas que hemos visto.

La tabla simplifica la relación entre *tareas*

TÉCNICA	PREDICTIVA / SUPERVISADA		DESCRIPTIVA / NO SUPERVISADA		
	Clasificación	Regresión	Clustering	Reglas de asociación	Otros (factorial, correlación...)
Redes neuronales	✓	✓	✓ *		
Árboles de decisión	✓ (GLS)	✓ (CART)	✓		
Kohonen			✓		
Regresión lineal (local, global), exp.		✓			
Regresión logística	✓				
K-means	✓ *		✓		
A Priori (asociaciones)				✓	
Análisis factorial, análisis multivariable					✓
CN2	✓				
K-NN (vecinos más próximos)	✓		✓		
FBR	✓				
Clasificadores básicos	✓	✓			

La diferencia entre *trear* y *técnica* es importante ya que si no confundiremos el problema que queremos resolver.

Veamos de forma descriptiva alguna de estas técnicas:

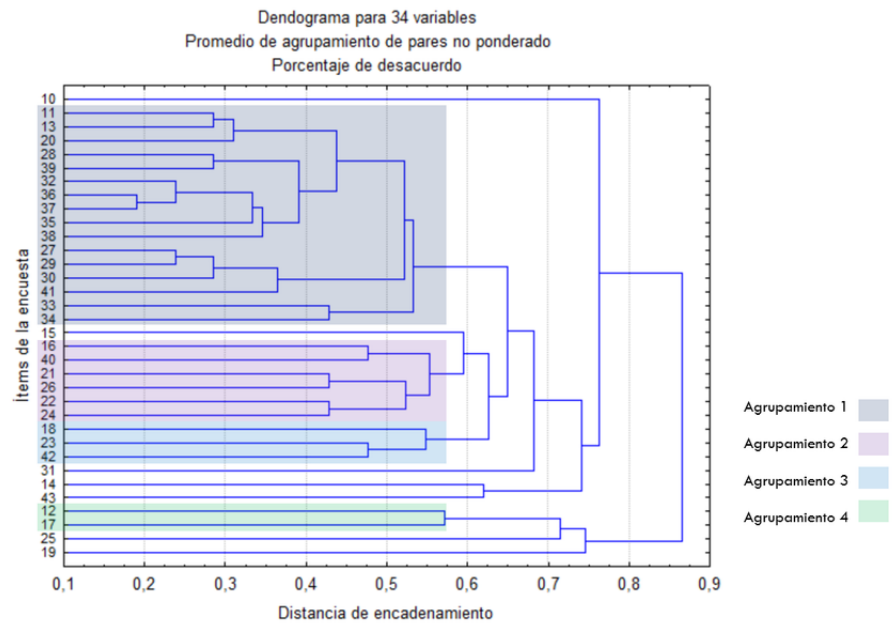
- **Técnicas descriptivas:**

- **Correlaciones y Asociaciones**, se suele denominar como *análisis exploratorio*, y aquí tenemos:

- * **Coefficiente de correlación**, cuando hablamos de variables numéricas y matrices de correlación, y queremos analizar más de dos variables. En principio el coeficiente de correlación es **bivariante**, ya que analizaría dos variables.
- * **Asociaciones**, hablamos de relaciones entre variables cualitativas, no numéricas, o categóricas. El ejemplo típico es la cesta de la compra, podemos preguntarnos si tabaco y alcohol están o no relacionados cuando se compran conjuntamente.
- * **Dependencias funcionales**, queremos conocer si una variable implica normalmente los valores de otra, se trata de una versión generalizada de las *reglas de asociación*.

- **Clustering**, normalmente se dividen en:

- * **Jerárquicos**, tenemos lo que se denomina un **dendograma**, en el que se agrupan ejemplos.



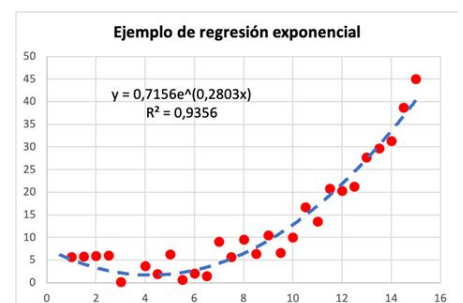
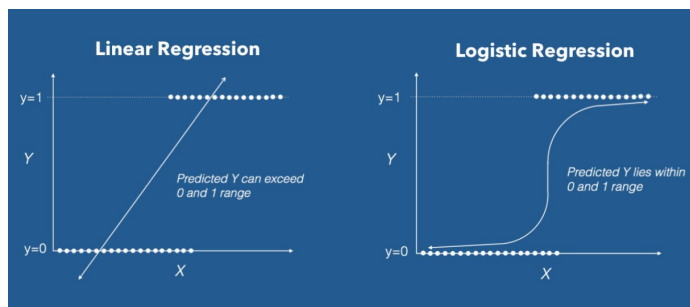
Son ejemplos que se agrupan por distancia de enlazado, al final podemos partir en los grupos que deseemos.

* **No Jerárquicos**, agrupan directamente en tres o cuatro grupos sin crear una jerarquía.

– kkkk

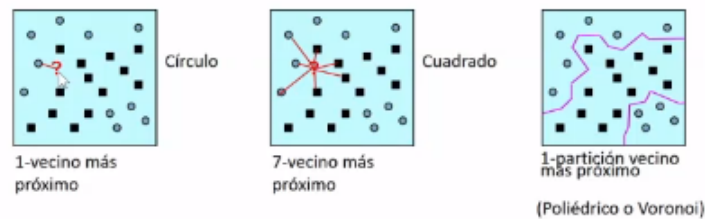
• **Técnicas predictivas.** Se refiere a los tipos de *regresión*:

- **Regresión lineal**, problemas de regresión, con variables de salida cuantitativa y numérica.
- **Regresión logística**, adaptación de la regresión lineal para problemas de clasificación, la variable de salida es *cualitativa*.
- **Regresión no lineal**, permite adaptarse a patrones, que de inicio, no son lineales.



Suelen verse juntas porque están muy relacionadas.

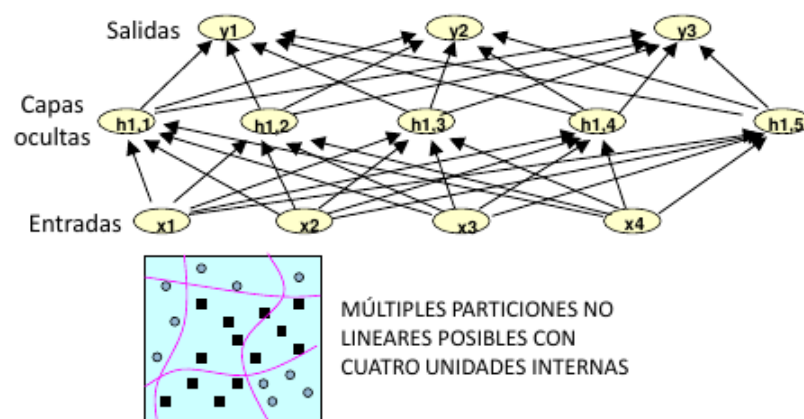
Veamos brevemente una serie de técnicas, una bastante antigua y usual es la denominada *vecinos más próximos*, muy intuitiva, se categoriza o predice dependiendo de sus vecinos; para determinar los vecinos hay una distancia, una medida de similitud y, a través de ella, se miden cuáles son los vecinos más próximos a un nuevo ejemplo. De ese nuevo ejemplo no tenemos la categoría, la clase. Lo que se hace es mirar cuáles son los vecinos más próximos y a partir de ahí determinar de qué clase es.



Si hablamos del vecino más próximo, sólo se mira uno, es menos robusto porque puede haber «ruido», puede ser que un grupo pueda estar rodeado de cuadraditos y el vecino más cercano pueda ser un círculo, cerca del punto que quiera predecir. La partición que se hace es muy parecida a la que haríamos con un lápiz si intentáramos determinar fronteras entre las clases, en la imagen entre cuadrados negros y círculos verdes.

Otro ejemplo bastante popular son las *redes neuronales*, y lo son por las aplicación de las *redes neuronales profundas* (las *redes neuronales no profundas* son sólo una capa oculta), se utilizan para muchas tareas, especialmente en las que las variables de entrada son bastante predictivas y no hay que crear nuevas características a partir de los datos.

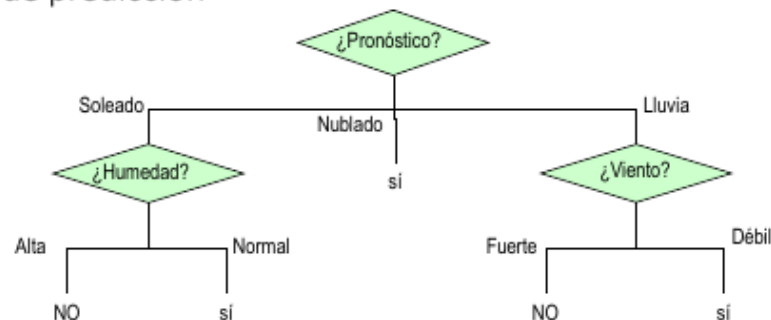
Redes neuronales



La ventaja que tienen las *redes neuronales*, de una capa o múltiples capas ocultas, es que permiten comportamientos **no lineales** como el mostrado en la imagen. Las fronteras se combinan a través de líneas que no son lineales, pudiendo ajustarnos a las formas que producen los datos como nosotros queramos.

Los *árboles de decisión* son muy comunes y es otra forma sencilla de entender los modelos extraídos utilizan técnicas que pueden representarse como un árbol, y se pueden representar en forma de reglas. Es una técnica muy utilizada en los entornos donde prima la *comprensibilidad*.

Árboles de predicción



A partir de los datos generamos un modelo, que dependiendo de tres variables: pronóstico, humedad y viento, podría determinar cuál será la clase de salida mediante los valores.

- Si pronóstico soleado,
- Si pronóstico nublado,
- Si pronóstico lluvia,
- Si viento fuerte,
- Si viento débil,
- Si humedad alta,
- Si humedad baja

Para tomar una decisión particular no tengo que evaluar todas las variables.

De la siguiente lista señala las **tareas**:

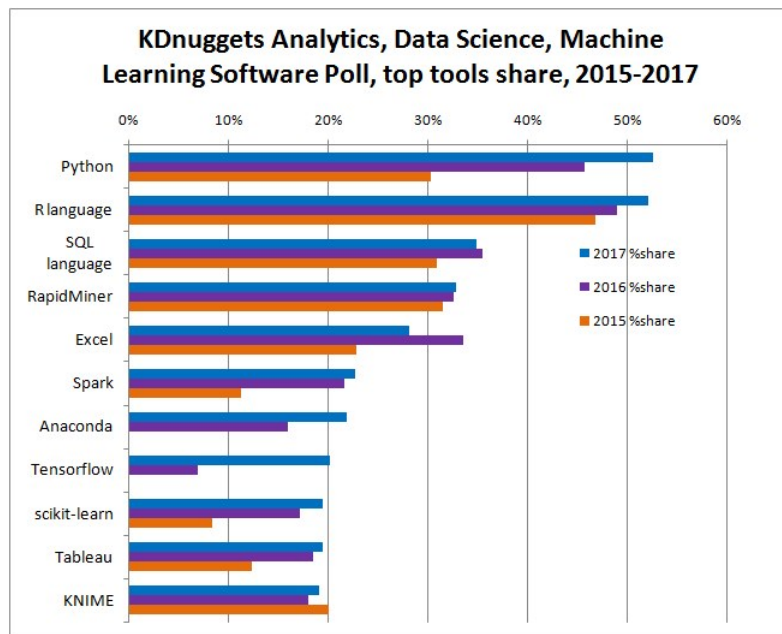
- A) Regresión lineal

- B) Clasificación
- C) Regresión
- D) Custering (agrupamiento)
- E) Redes neuronales
- F) Correlaciones
- G) Regresión Logística
- H) Reglas de asociación
- I) Vecinos más próximos

Solución: *B), C), D), F) y H)*

4.3 Herramientas

Para llevar a cabo las técnicas que hemos visto, y muchas más, se recurre a herramientas, en las que normalmente ya están implementadas dichas técnicas, ya sea en forma visual o mediante lenguajes de programación mediante el uso de librerías, o con programación orientada a objetos y librerías específicas como ocurre con *Python* o *R*.



Muchas de estas herramientas son gratuitas.

Lo interesante es ver que algunas de las herramientas se utilizan bastante en la comunidad:

- *R* y *Python* son muy utilizadas y además son gratuitas.
 - **R**, es un lenguaje interpretado, que dispone de muchas librerías, pero es necesario saber programar, crear código, lo que tiene ventajas y desventajas. Una de las grandes ventajas que tiene es que dispone de librerías de visualización muy potentes, por ejemplo **ggplot**.
 - **Python**, ha ido tomando espacio a *R* como lenguaje de programación predominante en el análisis de datos, especialmente a partir de aparecer la librería **Scikit-learn**, ésta contiene gran cantidad de técnicas de aprendizaje automático (*R* también dispone de algunas librerías de este tipo).
- Existen otro tipo de herramientas, estilo *RapidMiner*, que son más visuales, más sencillas de manejar, pero que suelen tener una licencia que ya no es completamente abierta.
- Están las herramientas comerciales, cuyo uso requiere de la compra de licencias, están relacionadas con el *BigData* y no tanto con el *aprendizaje automático*.
- Existe una serie de herramientas adicionales que podemos utilizar en muchos casos, que aunque no sean propiamente de análisis de datos o de aprendizaje automático son muy utilizadas; es el caso de *SQL*, muy utilizado cuando se trabaja con bases de datos, pero no es una herramienta de análisis, por el momento.

.

En la «nube» hay plataformas que permiten hacer el *aprendizaje automático* como son: AzureML y BigML, pero hay muchas otras.

Existen otras muchas herramientas que se *pagan por cómputo*, como **TensorFlow**, que podemos instalar en nuestra máquina pero si requiere hacer redes profundas y tienes unos clusters, en principio, a veces, es más cómodo lanzar todo esto mediante algún *servicio en la nube*, lo mismo ocurre con *Keras*. Todas estas herramientas, más sofisticadas, suelen utilizar en *aprendizaje profundo*, *reconocimiento de imágenes* o *de lenguaje natural*.

Chapter 2

Módulo 2: Evaluación de modelos de aprendizaje automático

La evaluación depende de la tarea que se realice. Distinguiremos entre tres tipos de tarea:

- **Aprendizaje supervisado**, donde tenemos variables de entrada y una variable de salida, que representa la solución deseada. La meta es aprender la *regla general* que convierte los datos de entrada en la solución correcta. Distinguimos entre:
 - **Clasificación**, donde la salida es una categoría.
 - **Regresión**, la variable resultado es numérica.
- **Aprendizaje no supervisado**, no se asigna ninguna etiqueta al algoritmo de aprendizaje, sólo existen variables de entrada. Distinguiremos:
 - Clustering
 - Reglas de asociación,
 - Correlaciones
- **Aprendizaje de Refuerzo**, donde un programa informático interactúa con un ambiente controlado en el que debe alcanzar una meta concreta: conducir un vehículo o los videojuegos son ejemplos de este tipo de tarea. Muy utilizado en Inteligencia Artificial y/o Robótica.

1

Métricas de Clasificación

La clasificación se predice a partir de una serie de entradas, para obtener una variable que es **categorica**:

- puede tener dos o más valores,
- no tiene un orden, puede ser si/no, a/b/c o d


el modelo lo que intentará predecir a cuál de las clases pertenecen los datos. Lo que tenemos que establecer es que si comparamos la clase predicha y la real coinciden.

La clase predicha se denota como $h(x)$ y la clase actual como $f(x)$, donde x es el ejemplo.

$$error_S(h) = \frac{1}{n} \sum_{x \in S} \delta(f(x), h(x))$$

Donde $\delta(a,b)=0$ si $a=b$ y de otra manera 1.

Clase predicha ($h(x)$)	Clase actual ($f(x)$)	Error
Comprar	Comprar	No
No Comprar	Comprar	Yes
Comprar	No Comprar	Yes
Comprar	Comprar	No
No Comprar	No Comprar	No
No Comprar	Comprar	Yes
No Comprar	No Comprar	No
Comprar	Comprar	No
Comprar	Comprar	No
No Comprar	No comprar	No

Errores / Total
 Error = 3/10 = 0.3

En el ejemplo, el modelo desplegado, a través de los datos etiquetados como *se ha comprado* o *no se ha comprado* un producto, se comparan con los que predice el modelo. Vemos que en algunos casos no concuerdan, esos casos son **errores**. En la clasificación lo que contamos es cuantas veces se falla, nos estamos refiriendo al *error*; en el ejemplo las veces que falla la predicción son 3 sobre 10, el error es de 30 % o 0.3. Podemos referirnos de forma inversa y decir que se acierta un 70 % de las predicciones, lo que normalmente se llama **porcentaje de acierto**, en inglés se designa con el término **accuracy**.

Este tipo de medida, porcentaje de acierto o porcentaje de error, es muy simple y en ocasiones se queda corta a la hora de entender como funciona realmente el modelo.

Una de las métricas habituales en clasificación son las que se conocen como:

- **Precision**, representa el porcentaje de documentos que son relevantes para la consulta, con **TP** no referimos a **True Positives**, *valores positivos que han sido verdaderos*, en el ejemplo TP /positivos predichos. ..
- **Recall**, representa el porcentaje de documentos que se devuelven por el estudio o modelo (TP), TP /positivos reales.

Si queremos integrar estados medidas, que nos dé más información, podemos recurrir otro tipo de medida, también habitual que se denomina **Medida F**, o también denominada **media armónica**:

$$\text{Medida } F = 2 \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} = \frac{2TP}{2TP + FP + FN}$$

Un modelo con una *Medida F* alta suele tener medidas altas en precision y recall.

TP o Positivos Predichos correctamente.

FP o Negativos Predichos como Positivos.

TN o Negativos Predichos correctamente.

FN o Positivos Predichos como Negativos.

TNRate o porcentaje de negativos verdaderos

estos parámetros nos permiten conocer el rendimiento de un modelo de clasificación binaria en circunstancias específicas.

☐ Positivos Predichos=TP+FP; Positivos Reales=TP+FN

☐ Positivos Predichos=TP+FN; Positivos Reales=TP+TN

☐ Positivos Predichos=TP; Positivos Reales=TP+FN

☒ Positivos Predichos=TP+FN; Positivos Reales=TP

2

Métricas para regresión

Recordemos que los modelos de regresión predicen un valor numérico a partir de una serie de entradas, este valor numérico no es algo exacto, sino que tiene decimales, en ocasiones muchos. Puesto que utiliza gran cantidad de datos, éstos pueden contener «*ruido*» que hace que sea difícil obtener un valor exacto para la variable de salida.

Si quisieramos predecir la edad de una persona es una tarea difícil, lo que nos interesa es que los valores que obtengamos no se alejen mucho del valor real. Para evaluar cuánto nos alejamos o acercamos a ese valor real existen diferentes aproximaciones, que vemos a continuación, y se fundamentan en el cálculo del **valor absoluto**, entre *valor real* y el *valor predicho*:

Error Medio Absoluto, $MAE_S(h)$, lo que se hace es evaluar si el valor real menos el predicho, valor absoluto, para cada uno de los ejemplos de la muestra.

$$MAE_S(h) = \frac{1}{n} \sum_{x \in S} |f(x) - h(x)|$$

Error Cuadrático Medio, $MSE_S(h)$, lo que queremos es penalizar más aquellos casos en los que el error es más grande, lo que desequilibra el valor real del error, donde hay ejemplos que tienen un error pequeño y otros que puede ser enorme; utilizamos el **error cuadrático**, consiste en elevar al cuadrado la diferencia entre el valor real y el predicho para cada uno de los ejemplos y calculamos su media.

$$MSE_S(h) = \frac{1}{n} \sum_{n \in S} (f(x) - h(x))^2$$


Con este cálculo un error de 5 unidades se transforma en 25, pero uno de 10 se convierte en 100.

Raíz Cuadrada del Error Cuadrático Medio, $RMSE_S(h)$, es un método que se utiliza para ajustar el valor que obtuvimos con $MSE_S(h)$ a algo más real, consiste en obtener la **raíz cuadrada**. En ocasiones se confunden estos dos últimos cálculos, y no se distingue cuál se utiliza hasta que no se ven los cálculos realizados.

$$RMSE_S(h) = \sqrt{MSE_S(h)} = \sqrt{\frac{1}{n} \sum_{n \in S} (f(x) - h(x))^2}$$

Veamos un ejemplo, tenemos valores predichos y actuales (reales) para diez ejemplos:

Valor predicho (h(x))	Valor Actual(f(x))	Error	Error ²
100 mill. €	102 mill. €	2	4
102 mill. €	110 mill. €	8	64
105 mill. €	95 mill. €	10	100
95 mill. €	75 mill. €	20	400
101 mill. €	103 mill. €	2	4
105 mill. €	110 mill. €	5	25
105 mill. €	98 mill. €	7	49
40 mill. €	32 mill. €	8	64
220 mill. €	215 mill. €	5	25
100 mill. €	103 mill. €	3	9


 $MAE = 70/10 = 7$
 $MSE = 744/10 = 74,4$
 $RMSE = \sqrt{744/10} = 8.63$

Ignoraremos los millones y calculamos el **error**, valor absoluto, como la diferencia entre valor actual y valor predicho, se suman todos estos valores obtenidos de error y se dividen por el número de ejemplos y obtendremos el valor de $MAE(h) = 7$. Por otra parte, el error cuadrático medio lo que hace es elevar al cuadrado cada uno de los valores obtenidos para el error, valor absoluto, y calcula la media, obteniendo un valor de $MSE_S(h)$ de 74,4, como vemos al elevar al cuadrado, algunos de los errores son muy elevados, lo que dificulta la comparación ambos valores. Por último, para corregir esto recurrimos a calcular la raíz cuadrada de $MSE_S(h)$, $RMSE_S(h) = \sqrt{74,4} = 8,63$.

¿Qué media se utiliza más?. En principio el $MSE_S(h)$, penaliza más los errores, más grandes, fijándonos en el caso de valor 400, es más de la mitad de la suma de todos

los cuadráticos, no contribuye a la media de todo el error cuadrático, lo que penaliza la métrica muchísimo más.

Con el valor del *error cuadrático medio* podemos ver los valores que detecta bien esos valores. Mediante la *raíz cuadrada* obtenemos un valor expresado en términos de la magnitud.

Podemos tener un modelo en que los valores alrededor de mil, y casi todos los ejemplos son mil uno, novecientos noventa y nueve, ..., es decir, que si yo tengo un error de tres, proporcionalmente es pequeño, tres entre mil es muy pequeño, pero en realidad si los valores reales están entre mil uno, mil dos, novecientos noventa y nueve, el error que cometo es muy alto, la variabilidad que tiene esos datos es tan pequeña que el error es mucho más grande que la variabilidad o por eso normalmente las métricas de error relativo lo que hacen es dividir por algo que mida más o menos la variabilidad. La forma de medir esta variabilidad es utilizando las **varianzas**:

Error Cuadrático Medio Relativo

$$RSE_S(h) = \frac{\sum_{n \in S} (f(x) - h(x))^2}{\sum_{x \in S} (\bar{f} - f(x))^2}$$

Error Absoluto Medio Relativo

$$RAE_S(h) = \frac{\sum_{n \in S} |f(x) - h(x)|}{\sum_{x \in S} |\bar{f} - f(x)|}$$

Una medida fácil de calcular y que permite ver si el modelo es alto cuando el valor real es alto y bajo cuando el valor real es bajo, es calcular la correlación:

- **Correlación de Pearson**, que es la correlación lineal.
- **Correlación de Spearman**, correlación de rangos o de orden.

2.1 Métricas para Aprendizaje No Supervisado

Una de las tareas más habituales en *aprendizaje no supervisado* es el **agrupamiento** o **clustering**; lo que queremos obtener con el agrupamiento, donde tenemos un conjunto de datos y en el que no hay variables de salida, sólo de entrada, es separar los ejemplos que tienen relación entre sí y forman un grupo, de los que se diferencian y pertenecen

a otros grupos. La idea es la diferencia o similitud en términos de distancia, se trata de una *métrica de distancia*. Para ver que elementos son diferentes, establecemos si la distancia entre elementos es alta.

¿Cómo creamos estos grupos? . Podemos crear grupos donde los elementos que los integran estén muy cerca, a poca distancia, y que estén muy lejos unos grupos de otros. Esto dependerá de la técnica que utilicemos para el agrupamiento, no todas las técnicas hacen el agrupamiento de la misma forma.

Por ejemplo, si utilizo una técnica de agrupamiento que me saca bordes en los grupos, lo que me puede interesar es establecer la distancia entre los «*bordes de los clusters*» o entre los *elementos más extremos*, pero puede interesarme establecer la distancia entre los centros de dos grupos para ver si están lejos o no. Por lo tanto, podemos mirar un *centroid*, *el radio, lo grande que son los clusters...*, nos compleja la evaluación de elementos de los modelos de agrupamiento, pero hay muchas métricas, dependiendo de si queremos más separación, que estén mas compactos.

Un buen clustering basado en distancias debe cumplir que:

- ☐ Los elementos del mismo cluster están cerca, y los centros también.
- ☒ Los elementos del mismo cluster están cerca, y las distancias entre centros deben ser altas.
- ☐ Los elementos de diferentes clusters deben estar lejos y los centros cerca.
- ☐ Los elementos de diferentes clusters deben estar cerca y los centros lejos.

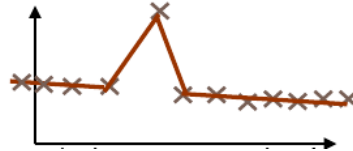
2.2 Sobreajuste

Pensemos en un modelo de regresión en el que queramos evaluar la edad o las ventas dependiendo del mes, si disponemos del *error cuadrático medio* o el *error absoluto*, lo que necesitamos es extraer el menor error posible. Es relativamente fácil, si disponemos de unos datos, ajustar esos datos de forma que minimizen el error, pero si luego lo utilizamos con otros datos, el mismo modelo, puede que no se ajuste igual de bien.

El típico ejemplo de **sobreajuste**, que todos vivimos como estudiantes al estudiar un exámen anterior real, nos hemos ajustado a las preguntas que salían en él, pero cuando hacemos el exámen real, las preguntas han cambiado, y el resultado no ha sido lo bueno que esperábamos. Es un caso típico de *sobreajuste* que aparece en el «*aprendizaje humano*», en el *aprendizaje automático* aparece exactamente lo mismo; si entrenamos un modelo a partir de unos datos y lo evaluamos sobre esos datos es fácil engañarse, es muy fácil ajustarse a esos datos y cuando tengamos otros datos desconocidos fallemos.

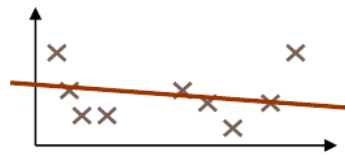
Podemos verlo reflejado gráficamente como:

Over-fitting o sobreajuste , utilizamos todos los datos para entrenar y evaluar los modelos. Gráficamente uniríamos todos los puntos, como cuando hacemos un dibujo., pero no generalizamos en las zonas en las que estén cerca esos puntos



En la gráfica vemos un pico, que se correspondería con un error o porque los datos en esa zona tienen una cierta variabilidad e intentamos un ajuste a algo que realmente no es un patrón de datos. Lo que deberíamos haber hecho es seguir la línea y evitar el pico.

Under-fitting o subajuste , compensamos el modelo generalizándolo, el ejemplo sería *podar un árbol de datos*



Viendo la gráfica podríamos pensar que los datos tienen un cierto patrón o continuo entre los puntos representados, que podría intentar ajustar, pero no lo hacemos. Hemos generalizado demasiado el modelo con lo que prácticamente no obtenemos ningún patrón.

Esta es la duda principal, cómo puedo resolver o distinguir entre ambos casos automáticamente o visualmente. Lo que haremos es lo que se denomina **regla de oro de la evaluación**:

nunca usar el mismo ejemplo para entrenar el modelo y evaluarlo

La mejor forma de hacer esto es escoger un conjunto de datos y separarlos o disponer de dos conjuntos diferentes de datos y utilizar un conjunto para el entrenamiento y el otro para la evaluación. En la práctica lo que haremos será coger los datos y los dividiremos entre datos de entrenamiento y datos para test.



¿Cuántos cogemos para entrenamiento y cuántos para el test? Variará dependiendo del experimento, pero suele ser 2/3 para entrenamiento y 1/3 para test, o 70 : 30.

¿Cómo se hace esta separación?. No se hace en orden, sino de forma aleatoria; no se hace en el orden en que se reciben los datos ya que pueden tener algún tipo de sesgo, se hace un muestreo aleatorio, donde extraemos el 70 % de los datos para entrenamiento y el resto para el la evaluación.

En el *entrenamiento* lo que hacemos es aplicar los **algoritmos** y **obtener modelos**, que constituye la parte central del aprendizaje automático. Una vez obtenidos los modelos los **evaluamos** con los datos de test, con ello podremos empezar a calcular las métricas o lo que sea necesario. En ese caso sabemos que si el modelo se hubiera ajustado a particularidades, que por casualidad han caído en los datos de entrenamiento o intenta ajustarse mucho al «*ruido*» de esos datos, variabilidades que puedan presentar los datos, cuando se entra en la fase de test, eso se notará, el modelo tendrá peores resultados en la evaluación. Con eso podremos resolver el *sobreajuste* y el *subajuste*, al final la medida que tenemos que mirar es la *medida que nos den los datos de la evaluación*.

¿Y la **cohesión**?. Hemos dividido los datos en la proporción 70:30, imaginemos que tengo cien datos, puedo pensar ¿por qué no entreno con los 100 datos?, si 100 datos son pocos y ahora los divido en dos grupos (70:30), pierdo un 30 % de datos en el entrenamiento y tan sólo utilizo 30 ejemplos para evaluar el modelo. La solución no es aumentar el porcentaje de datos para entrenamiento, esto hace que siga disminuyendo el número de datos para evaluar el modelo, el balance entre datos de entrenamiento y de test es difícil de encontrar, sobre todo cuando el volumen de datos es pequeño.

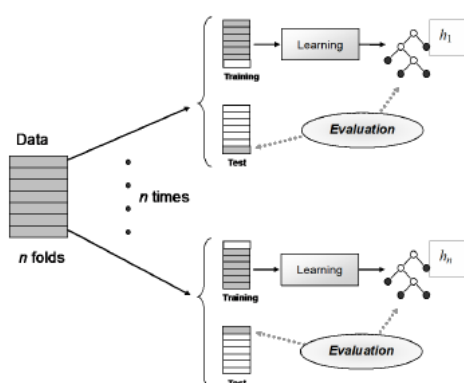
Cuantos más ejemplos tengamos para entrenamiento y test mejor, pero siempre sin romper la *regla de oro*. Si no queremos un entrenamiento y evaluación pobres, y tenemos pocos datos, la solución se basa en la **repetición del experimento**, varias veces sin romper en cada uno de los experimentos la *regla de oro*, y al final hacer **promedios de los resultados** obtenidos en cada experimento. Hay bastantes maneras de hacerlo, pero nos centramos en dos:

- **Bootstrap**, se cogen n-muestras con repeticiones y se realiza la evaluación con el resto, se obtienen ejemplos con repetición, un ejemplo puede salir varias veces, y luego se testea con el resto. No se sigue ningún orden.
- **Validación cruzada**, particionamos los datos en lo que se llaman **pliegues**, evaluamos con un pliegue y aprendemos con el resto de ejemplos.

2.3 Validación Cruzada

En un contexto donde tengamos muchos datos, el problema de los datos de entrenamiento y de evaluación no sería un problema, tendríamos de datos suficientes para aplicar la *regla de oro*, pero en los casos en que tenemos pocos datos (100, 200, ...) tenemos un dilema, al dividirlos en datos de entrenamiento y de validación, tendríamos pocos casos para ambos grupos, por lo tanto, hemos de intentar resolver este problema.

Existe una solución, que a base de aplicar repetición podemos conseguir no romper la *regla de oro*. ¿Cómo conseguir este proceso mágico?. Es un proceso que se conoce como **validación cruzada**, en principio, tenemos los datos que partiremos en *n*-*pliegues*,



en la imagen vemos que se han creado 7 pliegues, quiere decir que por cada iteración tendremos siete iteraciones, en la imagen sólo vemos la primera y la última; en cada iteración cogeremos uno de los pliegues para test, evaluación del modelo, y el resto para entrenamiento, esto se hará siete veces. A continuación, lo que se hace es sencillo, cogemos los seis pliegues de entrenamiento, cada pliegue puede tener 10, 15, ... ejemplos, los que hayan salido y entrenar el modelo y, por último, evaluarlo con el pliegue destinado a test.

Supongamos que tras el proceso obtenemos un 93% en el primero de los pliegues. En el pliegue dos podríamos obtener un 92.5%, así con el resto de pliegues, y el último podría ser de 94.2%. Hacemos un promedio de esos valores, que podemos suponer que sea de 92.7%, con ese promedio, que es el de *evaluación*, indicaría cómo se comporta el modelo. Lo siguiente sería determinar con cuál de los grupos (pliegues) nos quedamos, y a cuál de ellos asigno el promedio estimado. La respuesta es **a ninguno**. Lo que haríamos, que no refleja la imagen, es escoger todos los datos y entrenamos el modelo con todos esos datos, el resultado, en este ejemplo que hemos supuesto que era de clasificación, es un **árbol de decisión** que se comporta como la media de todos los otros modelos que entrenamos en su momento, que era 93.7%, lo que hacemos es quedarnos con el modelo que acabamos de entrenar, con todos los datos y le asignamos una evaluación de 93.7%. ¿Por qué establecemos esta cantidad? Si nos fijamos, todos los modelos se han evaluado sólo con seis pliegues, pero el último modelo lo hemos generado con todos los pliegues,

el conjunto de datos inicial, con lo que tenemos más datos, lo que hace que el modelo obtenido es mejor, suele ser una estimación más robusta y, además, conseguimos entrenar el modelo con todos los datos. Si nos fijamos, en ningún momento hemos evaluado con datos que se utilizaron para el entrenamiento.

Normalmente, en este método de *variación cruzada* es habitual utilizar diez pliegues, pero depende del coste computacional, cada pliegue son repeticiones, si hay diez pliegues, son diez repeticiones más el modelo final, es decir, *once procesos de aprendizaje*. Si se trata de una tarea costosa como puede ser una red neuronal, podemos limitar esos pliegues a cinco o cuatro, pero si no son muchos los datos, que es cuando realmente utilizamos la *variación cruzada*, podemos irnos a 10, 20 o incluso hacer tantos pliegues como ejemplos, en el caso de que los recursos nos lo permitan y tengamos pocos ejemplos.

¿Qué modelo se utiliza cuando se realiza validación cruzada?

☐ Se elige el modelo que obtiene mejor resultado de todos los pliegues

☐ Se elige el modelo que obtiene peor resultado de todos los pliegues

☐ Se combinan las predicciones de todos los modelos de los pliegues

☒ Se usa un modelo entrenado con todos los datos

Hemos visto que los modelos de *clasificación* se suelen evaluar con métricas relacionadas con el *porcentaje de acierto*, mientras que las tareas de regresión se evalúan con métricas relacionadas con el *error absoluto medio* o el *error cuadrático medio*. Lo que no sabemos es dónde falla exactamente nuestro modelo. Para ello, vamos a ver para la *clasificación* y la *regresión* dos formas de representar cómo se manifiestan esos errores, como se distribuyen:

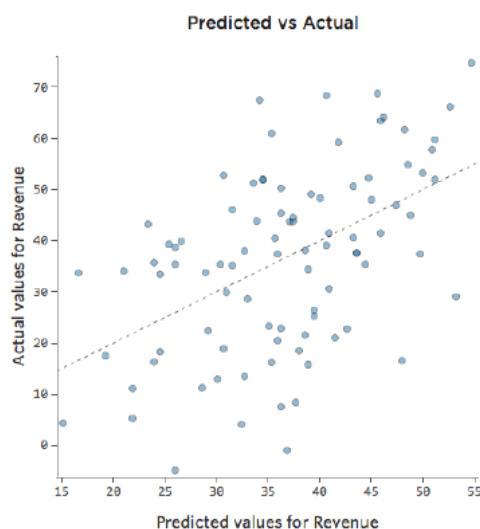
- *Clasificación*, veremos la **contingencia**, cómo se distribuyen los errores, también se denomina **confusión** o **matriz de confusión**.

		Actual	
Pred.	c	Comprar	No Comprar
	Comprar	4	1
	No Comprar	2	3

Esta matriz que representa un problema de dos clases: comprar o no comprar, quiere predecir si un cliente compra o no compra un producto. Suponemos que tenemos diez ejemplos, de ellos, si vemos la parte superior (Actual) representa los

valores reales, tendremos $4 + 2 = 6$ en la primera columna, seis casos reales en los que el cliente compró, y $1 + 3 = 4$ en los que no compró. Lo que queremos ver es si nuestro modelo *predice bien o no*.

- Podemos ver que realmente acierta siete veces, 4 se predicen como de compra y 3 como de no compra. Pero queremos saber su distribución, ¿cuántos de los que realmente se compró, van a parar a parar a no comprar y cuántos de los que no se compró, van a compró?, es decir, la otra diagonal de la tabla, la *diagonal de los errores*. Podemos ver que de los 6 que se compraron, en 4 acierta el modelo y en dos falla, y de los que no compraron, 1 falla y 3 acierta; vemos claramente como se distribuye. Vemos que de los 6 casos de comprar, acierta 4 y falla 2 que representa una tercera parte, y de los no compra, 4 ejemplos, falla 1 y acierta 3, una cuarta parte, por lo tanto, los errores parecen estar bien balanceados.
- Regresión*, en lugar de una matriz de 2,3, n-clases lo que tenemos es un número, en principio, infinito de valores y no podemos representarlos en una clase. Si predice 3.2 y el valor real es 3.1, estamos bastante cerca, aunque si se hiciera como clasificación, serían casillas diferentes. Aquí lo que generaremos es un **gráfico de dispersión** que representa en un eje el valor predicho y en el otro el actual (no importa en qué eje pongamos estos valores)



Si tenemos un valor predicho de 40 podemos ver el valor real correspondiente, siendo $\approx 20.xxx$, en este ejemplo indica que el valor real es mucho menor que el predicho. Los valores por encima de la diagonal del gráfico nos indican valores predichos que se han quedado más cortos que el valor real, mientras que los que están por debajo se han alejado más. Dicha diagonal representa un **clasificador perfecto**, y nos servirá para ver los casos de *sobreajuste* y *subajuste*, tendremos una visión bastante clara de cómo se distribuyen nuestros errores, tanto en clasificación como regresión.

Con estos procesos podemos comparar incluso distintas matrices de confusión o de contingencia y distintos gráficos de dispersión.

Una matriz de confusión donde todos los valores de la diagonal descendiente son ceros significa:

☐ Un modelo perfecto.

☐ Un modelo que acierta la mitad de las veces.

☒ Un modelo que falla siempre.

☐ Un modelo que acierta sólo cuando las clases son diferentes.

2.4 Clasificación binaria: Matriz de confusión

De una matriz de contingencia o de confusión de un clasificador podemos extraer cualquier tipo de medida.

Si tenemos esta matriz de confusión, en la que tenemos los valores reales, son las columnas, y los valores predichos son las filas, estos serían los positivos verdaderos, los positivos falsos, que son errores, por eso son falsos, los negativos verdaderos que son aciertos y los negativos falsos.

		Actual	
		Comprar	No comprar
Pred.	+	4	1
	-	2	3

		actual	
		+	-
predicted	+	TP true positive	FP false positive
	-	FN false negative	TN true negative
		TP+FN	FP+TN

donde:

- **TP** serían los positivos verdaderos.
- **FP** serían los positivos falsos, los errores.
- **FN** serían los negativos falsos.

- **TN** serían los negativos verdaderos, aciertos.

En el ejemplo de comprar-no comprar, la diagonal de aciertos se corresponde con los valores 4-3 o *true positive*, TP , mientras que la diagonal 1-2 son los fallos o *true negative*, TN . En cuanto a los porcentajes de aciertos o error se calcula mediante las dos fórmulas siguientes:

Accuracy : $\frac{TP+TN}{N} \implies \frac{4+3}{10} = 0.7$ o 70 % de aciertos.

Error : $\frac{FP+FN}{N} = 1 - \text{Accuracy} \implies \frac{2+1}{10} = 0.3$ o 30 % de error.

Por lo tanto, las dos medidas más clásicas en evaluación de clasificación se derivan fácilmente de la **diagonal de la matriz de confusión**.

Veamos otras medidas más complejas:

Sensibilidad o TPRate se representa como $\frac{TP}{TP+FN}$,

Especificidad o TNRate , se representa como $\frac{TN}{FP+TN}$,

Precision o PPV (valor positivo predecible) , se representa como $\frac{TP}{TP+FP}$

Estas dos medidas son simétricas, son dos puntos de la diagonal de aciertos divididos por la suma de cada una de las respectivas columnas.

Lo que suele confundir es que **sensibilidad**, que en inglés se conoce como **recall**, es el TP dividido por la primera columna , mientras que la *precisión*, es TP pero dividido la primera fila. Tanto *recall* como *precisión* tienen TP en el numerador, pero una denominador difiere en uno de los sumandos. En algunos libros o ejemplos, la posición de *actual* y *predicción* están cambiadas a como aparecen en la imagen, deberemos tener cuidado para determinar correctamente los valores que deberemos utilizar en las fórmulas para realizar los cálculos correctos. Muchas de las herramientas que podemos utilizar calculan tanto *recall* como *precision*, pero es importante saber como cada celda de la matriz de confusión afectan a estas medidas.

Por último, hay una medida, la que denominamos *F-measure* que es la **media armónica de esas dos medidas, TPRate y PPV**:

$$TP = a$$

$$FN = b$$

$$FP = c$$

$$\text{TPRate} = \frac{a}{a+b}$$

$$\text{PPV} = \frac{a}{a+c}$$

$$\begin{aligned}
TPRate * PPV &= \frac{a}{a+b} * \frac{a}{a+c} \\
&= \frac{a^2}{a^2 + ab + ac + bc} \\
&= \frac{a^2}{a(a+b+c) + bc}
\end{aligned}$$

$$\begin{aligned}
TPRate + PPV &= \frac{a}{a+b} + \frac{a}{a+c} \\
&= \frac{a(a+c) + a(a+b)}{(a+b)(a+c)} = \frac{a(a+c+a+b)}{a(a+b+c) + bc} \\
&= \frac{a(2a+c+b)}{a(a+b+c) + bc}
\end{aligned}$$

$$\begin{aligned}
F\text{-mesaure} &= 2 * \frac{PPV * TPRate}{PPV + TPRate} \\
&= 2 * \frac{a^2 / (a(a+b+c) + bc)}{a(2a+bc) / (a(a+b+c) + bc)} \\
&= 2 \frac{a}{2a+b+c} = 2 \frac{TP}{2TP + FP + FN}
\end{aligned}$$

Estas medidas se aplican a *clasificadores de dos clases*, es por lo que en la tabla vemos los signos $+$ y $-$ (positivos y negativos); ¿qué hacemos cuando tenemos más de dos clases?. Por ejemplo un problema como el siguiente:

ERROR		<i>actual</i>		
		low	medium	high
<i>predicted</i>	low	20	0	13
	medium	5	15	4
	high	4	7	60

La **matriz de confusión o contingencia** funciona del mismo modo que para dos clases, en este caso, para la clase real tenemos tres valores: bajo, medio y alto, y los mismos para la clase predicha. La **diagonal descendente (porcentaje de aciertos, 20-15-60)**, el porcentaje de aciertos sería:

$$\text{Porcentaje de aciertos} = \frac{20 + 15 + 60}{20 + 13 + 5 + 15 + 4 + 4 + 7 + 60} = \frac{95}{133} = 0.714286 \approx 71.4286 \%$$

El error serían todos los valores que no están en esa diagonal, con lo que es fácil generalizar estas medidas, ya sean tres, cuatro, cinco o más. Simplemente la *matriz de confusión será*

más grande, pero la **diagonal descendente** representará siempre los **aciertos**. Lo que sí será más complejo es derivar medidas como la *precision* o *recall*, ahora no podríamos hablar de positivos o negativos, en estos casos, lo que se suele hacer, *se elige una de las clases como la positiva*. En nuestro ejemplo podría ser *low*, agrupando como **clase negativa** a *medium* y *high*, con lo que estos valores se sumarían, transformándose en una matriz de 2×2 , *medium* y *high* pasarían a ser *medium-high*. Lo hemos planteado para *low*, pero podríamos haber tomado *medium* o *high* como clase *positiva*. Si hacemos las tres combinaciones posibles y promediamos esos valores, obtendríamos una media en la que podríamos calcular cualquiera de estas métricas; se podrían hacer:

- 1 contra todos, promedio de N medidas parciales.
- 1 contra 1 (promedio $N * (N - 1)/2$ medidas parciales): *low* contra *medium*, *low* contra *high* y *medium* contra *high*. Tenemos estas tres combinaciones, pero si tuviéramos más clases, aquí el número de combinaciones sería mayor, se dispararía.

Son sólo **medidas promedio**, podemos ver que en nuestro ejemplo, la clase *high* tiene muchos más ejemplos que la clase *medium*, por lo que podemos recurrir a opciones de promediar o ponerle pesos a distintas de esas medidas para calcular el valor medio.

Viendo la *matriz de confusión* podemos analizar como mejorar nuestro modelo.

Una matriz de confusión o contingencia en un problema de cuatro clases:

☐ Tiene 4 celdas

☐ Tiene 8 celdas

☒ Tiene 16 celdas

☐ No se puede definir para cuatro clases

2.5 Datos no balanceados

Uno de los problemas que se tienen en los problemas de clasificación es que no tenemos el mismo número de ejemplos en las dos clases. En la mayoría de los casos son situaciones **no balanceadas**, tenemos muchos más de un ejemplo que de otro.

Podemos encontrar diferencias significativas en la proporción de datos entre clases. Por ejemplo, si tenemos un 99 % de ejemplos de una clase y un 1 % para la otra clase, en

este caso la predicción será siempre de la clase mayoritaria, tendría un 99% de aciertos en la *matriz de confusión*.

Pero nos encontramos que con frecuencia **la clase minoritaria** suelen ser las más importantes, que más nos interesa determinar cuando fallan, pueden ser la causa de problemas bastante serios y necesitaríamos disponer de un predictor para estos casos.

Estas situaciones desbalanceadas nos obligan a establecer un error muy bajo, ya que ignoran a esta clase minoritaria, priorizando a la mayoritaria. En estos casos podemos querer fijarnos en una única medida, la **macroprecisión**¹, la medida clase por clase, nos mostraría que hay algo problemático en nuestra clasificación. Si tenemos que siempre acertamos con la clase mayoritaria, pero siempre fallamos con la minoritaria, al hacer el promedio por clases tendríamos un 50% de acierto de una clase perfecta y una clase desastrosa, lo que nos indicaría que nuestro clasificador no sirve para nuestro objetivo.

Es un ejemplo en el que vemos que simplemente mirar medidas agregadas, muchas veces nos puede confundir y es mucho más útil, en general, mirar la matriz de confusión o la matriz de contingencia.

$$macroacc(h) = \frac{\frac{Hits_{class 1}}{total_{class 1}} + \frac{hits_{class 2}}{total_{class 2}} + \dots + \frac{hits_{class m}}{total_{class m}}}{m}$$

Un clasificador que siempre prediga positivo en un problema binario que tenga un 90% de instancias positivas y un 10% de instancias negativas tendrá, para ese conjunto de datos

☐ Un 90% de precisión y un 90% de macroprecisión

☒ Un 90% de precisión y un 50% de macroprecisión

☐ Un 50% de precisión y un 90% de macroprecisión

☐ Un 10% de precisión y un 50% de macroprecisión

2.6 Clasificadores duros y suaves

Una forma avanzada de analizar los clasificadores es ver lo que predicen, podemos tener clasificadores que nos indiquen:

¹macroprecisión $\rightarrow macroacc(h)$

- si tenemos varias clases, nos predigan a qué clase pertenece nuestro ejemplo.
- la probabilidad de que un ejemplo pertenezca a una de esas clases, o poner en orden la probabilidad de a qué clases es más probable que el ejemplo se ajuste, es lo que se conoce como **clasificador suave**.

Por lo tanto, un **clasificador suave** o «*scoring*» predice una clase, pero acompaña cada predicción con una *estimación de la fiabilidad* de cada predicción. Por ejemplo, tenemos tres clases, un clasificador suave nos puede dar como resultados:

- 7 para la primera clase.
- 2 para la segunda,
- 0.3 para la tercera

esto quiere decir que el valor más alto corresponde a la primera de las class, el algoritmo de clasificación estará más seguro de que pueda ser de la primera clase.

Los clasificadores suaves no proporcionan más posibilidades de funcionar y mejorar nuestras predicciones que los **clasificadores duros** o «*crisp*».

Probabilidades y decisiones

Una **probabilidad** se puede convertir en **decisiones** mediante el uso de un **umbral**. Por ejemplo, un médico puede dar un medicamento a un paciente sólo si la probabilidad de que sea efectivo sea mayor de el 80 % , mientras que otro médico puede establecer el umbral en el 60 % .

Si un modelo asignara el 70 % de probabilidad, con el primer umbral no se le daría el medicamento, pero con el segundo sí. Vemos que con el mismo modelo suave, variando el umbral, podemos tener diferentes decisiones.

No es necesario que el **valor suave** sea realmente una probabilidad en el rango de 0 y 1 (0 % y 100 %), ocurre lo mismo con cualquier otro valor numérico, como puede ser la **confianza**.

Un clasificador que nos ofrece una medida de la fiabilidad de la clasificación realizada es un clasificador:

☐ Duro

☒ Suave

2.7 Clasificadores probabilísticos

Como vimos, los *clasificadores suaves* no nos dan directamente la clase que queremos estimar, sino un valor de qué clase tiene una fiabilidad mayor de ser correcta.

Un tipo de *clasificador suave* es el denominado **estimador probabilístico de clase**, un sistema al que se asignan unas entradas sobre el ejemplo, pero en lugar de predecir la clase nos da la **probabilidad para cada una de las clases u opciones**. Si tenemos tres clases (a, b y c), obtendremos la probabilidad para cada una de ellas: p_a , p_b y p_c .

Digamos que tenemos dos clasificadores, $clasificador_1$ y $clasificador_2$ a los que asignamos los mismo valores de entrada, obtenemos los siguientes resultados:

- $clasificador_1$: $p_a = 0.2$, $p_b = 0.5$ y $p_c = 0.3$.
- $clasificador_2$: $p_a = 0.3$, $p_b = 0.4$ y $p_c = 0.3$.

ambos predicen mejores resultados para el *clase* b ; si nuestro umbral a la hora de determinar que predecimos, simplemente es elegir la clase con la mayor probabilidad, ésta corresponde al $clasificador_1$ obtiene mejor resultado; si tuvieramos que determinar que clasificador está más seguro de la predicción, en principio, también sería el $clasificador_1$, es más confiable pues se corresponde con el 50%, mientras que el segundo nos da el 40%. El segundo clasificador habría sido más **conservador**, y si al final la clase correcta no fuera la b , el segundo clasificador habría cometido un *error menor* que el primero.

¿Cómo podemos evaluar lo buenas que son las estimaciones de nuestro ejemplo?. Necesitamos medidas que nos evalúen lo buenas o acertadas que son esas probabilidades.

Los *clasificadores probabilísticos* predicen una probabilidad, tenemos un ejemplo y denominamos la *probabilidad para el ejemplo i sobre la clase j* , obtendremos una definición que sirve para problemas de dos, tres o cuatro clases, que determinará o estimará la probabilidad para el ejemplo i y la clase j , y vamos a ver el valor real para el ejemplo i y la clase j . Si la clase del ejemplo i es la *clase j* su valor será 1, y si no será 0, en cambio, las probabilidades pueden ir entre 0 y 1.

$$MSE = \frac{1}{n} \sum_{i \in S} \sum_{j \in C} [f(i, j) - p(i, j)]^2$$

Aplicando la fórmula del *error cuadrático medio* (MSE), tendremos que en la parte izquierda, valores reales ($f(i, j)$) sólo habrá valores 0 o 1, mientras que en la parte de predicción ($p(i, j)$) tendrá valores que pueden ir en el rango de 0 a 1. Si $f(i, j) = 1$, es decir, la *clase es la j* y la probabilidad estimada $p(i, j) = 1$ para esa clase, es perfecto, estaríamos diciendo que para las otras clases la probabilidad estimada sería 0. Igual ocurre si los dos valores anteriores son 0, establece que la probabilidad de que sea la clase j es nula.

Por lo tanto, si no es esa clase, sea 0, y obtengamos un valor que no sea 0, será un error y todo lo que nos alejemos de 1 y nos desviemos desde el 1, que sería predecir que sí va a ser esa clase, también estamos desviandonos de la posibilidad perfecta. Es lo que hace la diferencia entre f y p , con el error cuadrático medio nos aseguramos de quitar un posible signo negativo, al estar elevada al cuadrado, por lo tanto el resultado siempre será positivo.

En el caso de los clasificadores probabilísticos el **error cuadrático medio** también se denomina **Brier Score**, aunque la definición es la misma, se denomina así porque este valor sólo puede ser **1** o **0**. Ésta es la medida más habitual para evaluar lo bueno que es un *clasificador probabilístico*.

Otra medida bastante habitual, relacionada con la *entropía* es lo que denominamos **Log Loss**, simplemente es operar con los mismos valores, donde el valor real que se multiplica por el **logaritmo** de la probabilidad, si los valores son similares darán valores bajos y si son diferentes, valores altos. Cuando más altos sean los valores, peor será el clasificador.

$$Logloss = -\frac{1}{n} \sum_{i \in S} \sum_{j \in C} (f(i, j) * \log_2 p(i, j))$$

Tanto *Brier Score* y el *Log loss* incluyen una medida que es un estimador de probabilidades. En ocasiones no sabemos si un estimador es bueno porque la clasificación que hacemos con él es buena, o porque realmente las probabilidades están bien estimadas. Podemos tener un clasificador que siempre nos de 0.5 de probabilidad para todas las clases, suponiendo que es un problema binario, en principio, si tenemos sólo dos clases, si tenemos un 50 % de ejemplos para dos clases, podríamos ver si la estimación se decanta por una clase o mucho por la otra; podría estar bien calibrado o balanceado, pero el sistema no servía para nada ya que siempre estima un 50 % para cada clase, no discrimina entre ejemplos.

Para determinar si en un modelo las probabilidades están bien calibradas y si el modelo realmente refina bien entre las dos clases utiliza la descomposición, para el *Brier Score*, entre *medidas de calibración* y *medidas de refinamiento*. Cuando analizamos la información entre *calibración* y *refinamiento* podemos observar que a veces, sin modificar el modelo, podremos mejorar la *estimación de probabilidad*; podemos tener medias que estén muy cerca de 0.5 o muy extremas, lo que hacemos es un *postproceso* al clasificador, calibrándolo mejor y mejorando la estimación de probabilidades.

Un clasificador probabilístico se convierte en un clasificador duro (crisp) mediante:

☒ El uso de un umbral.

☐ Calibración.

☐ Descomposición entre CAL y REF.

☐ Escalado de Platt.

2.8 Rankers

Es otro tipo de *clasificador suave*, nos proporciona un valor y se suele aplicar a problemas de clasificación de dos clases, en éstos tenemos que el clasificador estimará un valor, un *scort* para cada uno de los ejemplos. Cuanto más alto sea el valor de *scort*, mayor será la probabilidad de en ese ejemplo sea positiva, y cuanto más bajo que sea de la clase negativa.

Si cogemos todos los ejemplos de un *dataset* lo que tenemos es un **ranking**, donde los que están más arriba son los más probables de ser positivos, y los que están abajo, más probable que sean negativos.

El valor predicho por el *ranker* no tiene por qué ser una probabilidad, simplemente es un valor más alto cuanto más probable es y viceversa. Lo que nos interesa es evaluar lo bueno que es ese ranking. Si tengo veinte unidades de un producto que quiero vender a mil clientes, como sólo tengo veinte unidades lo que me interesa saber, de los mil clientes, quienes tienen mayor probabilidad de compra de esps productos, con lo que puede que realmente necesite hacer una campaña sólo, por ejemplo, para un 10 % de los clientes que tengan mayor probabilidad de compra, muchos no comprarán aunque tenga una mayor probabilidad, ya que la compra dependerá de otros muchos factores. Por lo tanto, lo que nos interesa saber es el *ranking* y no tanto la *probabilidad*, esto está relacionado con las medidas de discriminación entre dos clases y lo que es el refinamiento, cuando se habla de estimación de probabilidad; esta parte del refinamiento es lo que realmente evaluamos al hablar de **ranker**.

En resumen, un **ranker** es un *clasificador suave* que da un *valor monóticamente relacionado con la probabilidad de clase uno*. Cuanto más alto sea mayor probabilidad de que sea la clase uno, clase positiva, y cuanto más bajo mayor probabilidad de que sea la clase cero, o negativa.

El *ranker* es muy utilizado en *marketing* a la hora de recomendar productos, lo que se


hace es escoger un segmento de aquellos items que están más altos en el ranking, un percentil (5 % o el 10 %) y sobre ellos se hace la campaña, de ellos puede que contesten un subconjunto, pero hemos limitado la campaña a aquellos con mayor probabilidad. Lo que nos interesa es ordenar todos los clientes y escoger los que nos interesan. Cuanto mayor sea el ranking, jmejor será la segmentación.

¿Cómo evaluamos lo bueno que es un ranker? Lo que más se utiliza es el **área bajo la curva ROC (Receiver Operating Characteristic)**, normalmente se utilizan las siglas inglesas, *AUC* (*Area Under the ROC Curve*). Normalmente esta medida es equivalente a una estadística conocida como *estadística de Willcoxon-Mann Whitney*, que se define como: *dado un ejemplo psitivo y un ejemplo negativo, la probabilidad de que el modelo coloque en el ranking de estimación al ejemplo positivo por encima del ejemplo negativo*. Por ejemplo, tenemos un conjunto de ejemplos de entrenamiento, con sus valores de clase positivos y negativos, cogemos al azar un ejemplo positivo, igual hacemos con los ejemplos negativos, comprobamos si nuestro **ranker** pone el positivo arriba del negativo; la probabilidad de que eso pase es el AUC. La forma de definirlo es simplemente realizar un conteo de cuántas veces, eligiendo al azar entre positivos y negativos, se confirma dicha ordenación. Esta métrica nos dice lo bueno que es nuestro ranking, un valor de 1 será perfecto, mientras que 0 será un desastre, lo normal es que el ranking esté en torno al 0.5.

Existe otras distancias para evaluar el ranking, por ejemplo, la distancia entre el *ranking perfecto* y el *ranking estimado*; esa discrepancia entre rankings, por ejemplo la denominada **discrepancia de Kendall**, suele ser equivalente al área bajo la curva ROC, por lo que muchas veces sólo se utiliza ésta para evaluar rankers.

Selecciona qué afirmación es FALSA:

☐ Un estimador de probabilidad siempre puede usarse como ranker

☒ Un estimador de probabilidad se define generalmente para dos clases únicamente 

☐ Un ranker se define generalmente para dos clases únicamente

☐ Un ranker puede calibrarse y actuar como un estimador probabilidad

2.9 Evaluación sensible al coste

Appendices

Appendix A

Introducción suave al método Bootstrap (Jason Brownlee)



¿Quieres ayuda con las estadísticas? [Toma el Mini-Curso GRATIS](#)



Una introducción suave al método Bootstrap

por **Jason Brownlee** el 25 de mayo de 2018 en <https://machinelearningmastery.com/a-gentle-introduction-to-the-bootstrap-method/>

El método bootstrap es una técnica de remuestreo que se utiliza para estimar estadísticas sobre una población mediante el muestreo de un conjunto de datos con reemplazo.

Se puede usar para estimar estadísticas de resumen, como la media o la desviación estándar. Se utiliza en el aprendizaje automático aplicado para estimar la habilidad de los modelos de aprendizaje automático al hacer predicciones sobre datos no incluidos en los datos de entrenamiento.

Una propiedad deseable de los resultados de la estimación de la habilidad del modelo de aprendizaje automático es que la habilidad estimada puede presentarse con intervalos de confianza, una característica que no está disponible fácilmente con otros métodos como la validación cruzada.

En este tutorial, descubrirá el método de remuestreo bootstrap para estimar la habilidad de los modelos de aprendizaje automático en datos desconocidos.

Después de completar este tutorial, sabrás que:

- El método bootstrap consiste en remuestrear iterativamente un conjunto de datos con reemplazo.
- Al usar el bootstrap debes elegir el tamaño de la muestra y el número de repeticiones.

Empecemos.

Tutorial general

Este tutorial está dividido en 4 partes; son:

1. Método Bootstrap
2. Configuración de la Bootstrap
3. Ejemplo

¿Necesitas ayuda con Statistics for Machine Learning?

Tome mi curso gratuito de correo electrónico de 7 días ahora (con código de ejemplo).

Haga clic para inscribirse y también obtenga una versión gratuita en PDF de Ebook del curso.

Descarga tu mini-curso GRATIS

Método Bootstrap

El método bootstrap es una técnica estadística para estimar cantidades sobre una población promediando estimaciones múltiples muestras más pequeñas.

Es importante destacar que las muestras se construyen sacando de una en una observaciones de una muestra de datos de gran tamaño y devolviéndolas a la muestra de datos después de haber sido elegidas. Esto permite que una observación dada se incluya en una muestra pequeña dada más de una vez. Este enfoque de muestreo se llama muestreo con reemplazo.

El proceso para construir una muestra se puede resumir como sigue:

1. Elija el tamaño de la muestra.
2. Mientras que el tamaño de la muestra es menor que el tamaño elegido
 1. Selecciona aleatoriamente una observación del conjunto de datos
 2. Añádela a la muestra.

El método bootstrap se puede usar para estimar un parámetro de una población. Esto se hace tomando muestras pequeñas repetidamente, calculando la estadística y tomando el promedio de las estadísticas calculadas. Podemos resumir este procedimiento de la siguiente manera:

1. Elija una serie de muestras bootstrap para realizar
2. Elija un tamaño de muestra
3. Para cada muestra bootstrap.
 1. Sacar una muestra con reemplazo con el tamaño elegido.
 2. Calcula la estadística sobre la muestra.
4. Calcular la media de las estadísticas de la muestras.

El procedimiento también se puede utilizar para estimar la habilidad de un modelo de aprendizaje automático.



El bootstrap es una herramienta estadística ampliamente aplicable y extremadamente poderosa que se puede usar para cuantificar la incertidumbre asociada con un estimador dado o un método de aprendizaje estadístico.

- Página 187, [Introducción al aprendizaje estadístico](#), 2013.

Esto se hace entrenando el modelo con la muestra y evaluando la habilidad del modelo en aquellas muestras no incluidas en la muestra. Estas muestras no incluidas en una muestra dada se denominan muestras fuera de bolsa, o OOB, para abreviar.

Este procedimiento de uso del método bootstrap para estimar la habilidad del modelo se puede resumir de la siguiente manera:

1. Elija una serie de muestras bootstrap para realizar
2. Elija un tamaño de muestra
3. Para cada muestra bootstrap.
 1. Sacar una muestra con reemplazo con el tamaño elegido.
 2. Ajustar un modelo en la muestra de datos.
 3. Calcule la habilidad del modelo con los datos de fuera de bolsa.
4. Calcule la media de las estimaciones de habilidades de los modelos creados con cada muestra

“ Las muestras no seleccionadas generalmente se conocen como muestras "fuera de la bolsa". Para una iteración dada de remuestreo bootstrap, se construye un modelo sobre las muestras seleccionadas y se usan para evaluarlo las muestras fuera de la bolsa.

- Página 72, [Modelización Predictiva Aplicada](#) , 2013.

Es importante destacar que cualquier preparación de datos antes de ajustar el modelo o ajustar el hiperparámetro del modelo debe ocurrir dentro del bucle for en la muestra de datos. Esto es para evitar la fuga de datos cuando se utiliza el conocimiento del conjunto de datos de prueba para mejorar el modelo. Esto, a su vez, puede resultar en una estimación optimista de la habilidad del modelo.

Una característica útil del método bootstrap es que la muestra resultante de estimaciones a menudo forma una distribución gaussiana. Además de resumir esta distribución con una tendencia central, se pueden dar medidas de varianza, como la desviación estándar y el error estándar. Además, un intervalo de confianza se puede calcular y utilizar para acotar la estimación presentada. Esto es útil cuando se presenta la habilidad estimada de un modelo de aprendizaje automático.

Configuración del Bootstrap

Hay dos parámetros que se deben elegir al realizar la rutina de arranque: el tamaño de la muestra y el número de repeticiones del procedimiento a realizar.

Tamaño de la muestra

En el aprendizaje automático, es común usar un tamaño de muestra que sea el mismo que el conjunto de datos original.



La muestra de bootstrap es del mismo tamaño que el conjunto de datos original. Como resultado, algunas muestras se representarán varias veces en la muestra de arranque, mientras que otras no se seleccionarán en absoluto.

- Página 72, [Modelización Predictiva Aplicada](#), 2013.

Si el conjunto de datos es enorme y la eficiencia computacional es un problema, se pueden usar muestras más pequeñas, como el 50% u 80% del tamaño del conjunto de datos.

Repeticiones

El número de repeticiones debe ser lo suficientemente grande para garantizar que se puedan calcular estadísticas significativas, como la media, la desviación estándar y el error estándar en la muestra.

Un mínimo puede ser de 20 o 30 repeticiones. Los valores más pequeños que se pueden usar agregarán más varianza a las estadísticas calculadas en la muestra de valores estimados.

Idealmente, la muestra de estimaciones sería lo más grande posible, dados los recursos de tiempo, con cientos o miles de repeticiones.

Ejemplo

Podemos demostrar el procedimiento de bootstrap con un pequeño ejemplo. Trabajaremos a través de una iteración del procedimiento.

Imagina que tenemos un conjunto de datos con 6 observaciones:

```
1 [0.1, 0.2, 0.3, 0.4, 0.5, 0.6]
```

El primer paso es elegir el tamaño de la muestra. Aquí, vamos a utilizar 4.

A continuación, debemos elegir al azar la primera observación del conjunto de datos. Vamos a elegir 0.2.

```
1 sample = [0.2]
```

Esta observación se devuelve al conjunto de datos y repetimos este paso 3 veces más.

```
1 sample = [0.2, 0.1, 0.2, 0.6]
```

Ahora tenemos nuestra muestra de datos. El ejemplo demuestra a propósito que el mismo valor puede aparecer cero, una o más veces en la muestra. Aquí la observación 0.2 aparece dos veces.

Luego se puede calcular una estimación sobre la muestra dibujada.

```
1 statistic = calculation([0.2, 0.1, 0.2, 0.6])
```

Las observaciones no elegidas para la muestra pueden usarse como observaciones fuera de la muestra.

```
1 oob = [0.3, 0.4, 0.5]
```

En el caso de evaluar un modelo de aprendizaje automático, el modelo se ajusta a la muestra dibujada y se evalúa a la muestra fuera de la bolsa.

```
1 train = [0.2, 0.1, 0.2, 0.6]
2 test = [0.3, 0.4, 0.5]
3 model = fit(train)
4 statistic = evaluate(model, test)
```

Con esto concluye una repetición del procedimiento. Se puede repetir 30 o más veces para obtener una muestra de estadísticas calculadas.

```
1 statistics = [...]
```

Esta muestra de estadísticas se puede resumir calculando una media, una desviación estándar u otros valores de resumen para obtener una estimación final utilizable de la estadística.

```
1 estimate = mean([...])
```

Extensiones

Esta sección enumera algunas ideas para ampliar el tutorial que tal vez desee explorar.

- Enumere 3 estadísticas de resumen que podría estimar utilizando el método bootstrap.
- Encuentre 3 trabajos de investigación que utilicen el método bootstrap para evaluar el rendimiento de los modelos de aprendizaje automático.
- Implemente su propia función para crear una muestra y una muestra fuera de bolsa con el método bootstrap.

Si exploras alguna de estas extensiones, me encantaría saberlo.

Otras lecturas

Esta sección proporciona más recursos sobre el tema si desea profundizar.

Mensajes

- [Cómo calcular los intervalos de confianza de Bootstrap para los resultados del aprendizaje automático en Python](#)

Libros

- [Modelado predictivo aplicado](#) , 2013.
- [Una introducción al aprendizaje estadístico](#) , 2013.
- [Una introducción a Bootstrap](#) , 1994.

API

- [sklearn.utils.resample \(\) API](#)
- [sklearn.model_selection: API de selección de modelo](#)