Fundamentos de Machine Learning

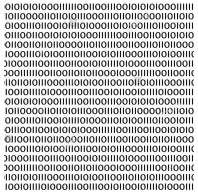
Antonio Jesús Gil

Contexto

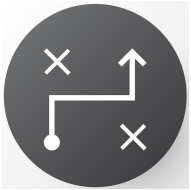
1. Introducción al Machine Learning
2. Aprendizaje Supervisado Vs Aprendizaje no Supervisado
3. Scikit-Learn
4. Pipelines
5. Evaluación y Validación de Modelos
6. Selección de Variables
7. Aprendizaje Inductivo

Machine Learning

Data



Algorithm



Model

¿Qué es?

- **Área** dentro de la Inteligencia Artificial.
- Se divide en Aprendizaje Supervisado y No Supervisado.
- **No** es solo un **algoritmo simple** que pueda ser colocado en cualquier lugar y comenzar a obtener resultados fantásticos.
- P**roceso** que comienza en la **definición** de datos y termina con un **modelo** preciso.

Machine Learning: Clasificación

Las tareas a abordar desde el aprendizaje automático se pueden dividir en dos grandes grupos:

- **Descriptivas**
- **Predictivas**

Entre las primeras están las tareas de detección de **agrupamientos** y **correlaciones**, y entre las segundas están las tareas de **clasificación** y **regresión**.

Machine Learning: Proceso

1. **Definir** problema: Comienza con definición de un problema empresarial, ¿**Cuál** es la necesidad del aprendizaje automático? ¿Esta tarea realmente **necesita** un algoritmo predictivo?
2. **Recolectar** datos
3. **Preparar** datos
    a. **Cleaning ->** datos erróneos, atributos sin valor.
    b. **Formatting ->** El algoritmo necesita datos **formateados.**
    c. **Sampling -> No** todos los datos son **útiles**: Borrar datos similares o si estos están etiquetados, eliminar instancias en proporción.
    d. **Descomposición** -> Hay características más útiles si se descomponen, ej. Fechas
    e. **Escalado** -> Atributos en diferentes unidades o valores: pulgadas, centímetros.

Proceso II:

4. Split data in **training** and **testing**: El objetivo es predecir datos nunca observados. El **modelo** se construye con datos de **entrenamiento** intentando reducir el **error**.
5. Algorithm **selection**: Dependerá de la definición del problema.

Problema

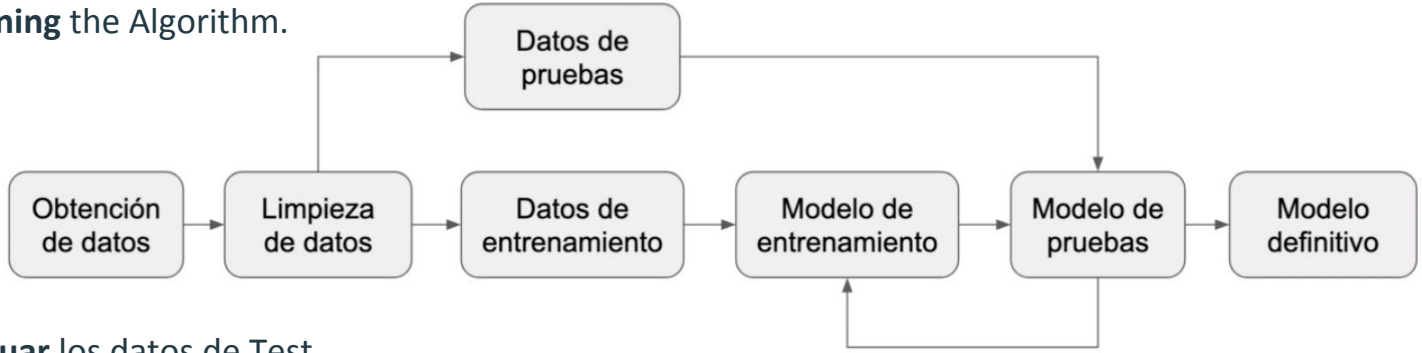
- Clasificar mails en spam / no spam
- Predecir una variable contínua
- Análisis de texto

Algoritmos

- Clasificadores
- Regresiones
- Clustering o Agrupamiento

Proceso III:

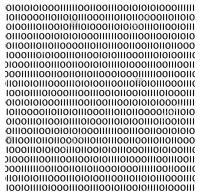
6. **Training** the Algorithm.



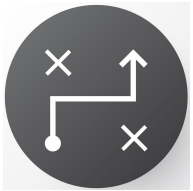
7. **Evaluar** los datos de Test.
8. Parameter **Tuning.**
9. Selection: Dependerá de la definición del problema.
10. Start **using** your **model.**

Resumen: Se identifica un problema

Data



Algorithm

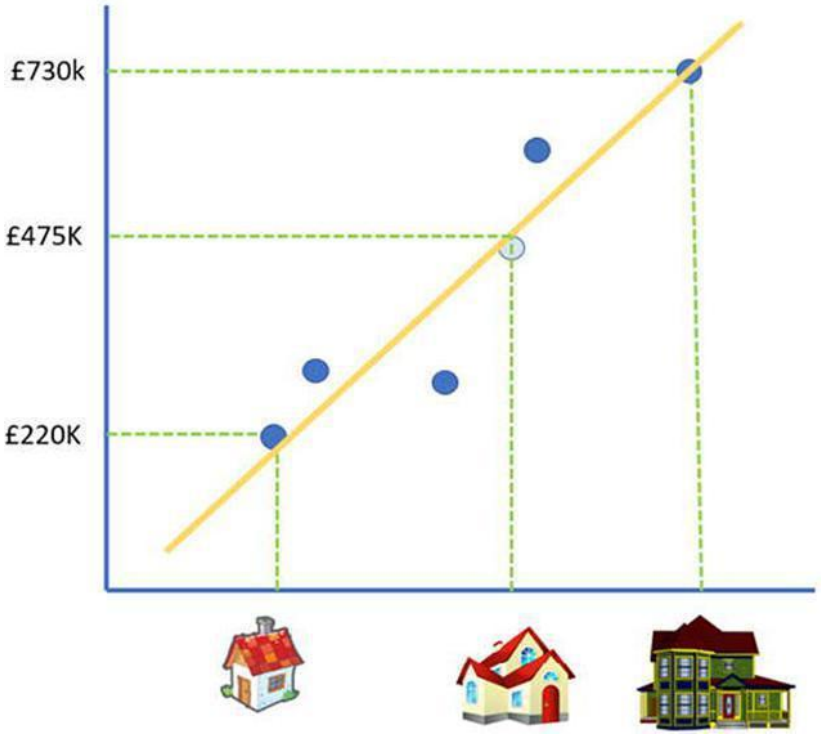


Model

$$f(x)$$

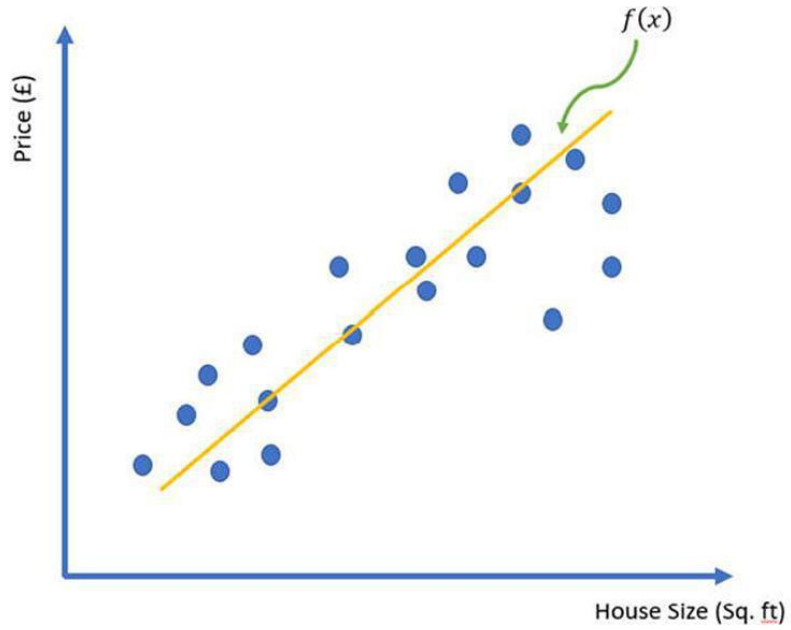
Finaliza con un algoritmo para resolver el problema.

Ejemplo: Análisis precio vivienda

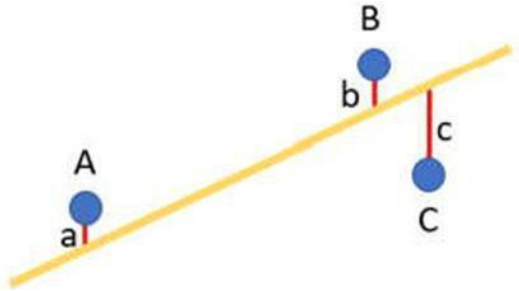


1. Queremos **predecir** el **coste** de la casa que hay en el medio.
2. Conocemos metros cuadrados que tiene y ubicación en la ciudad (**características**)
3. Usamos **Regresión Lineal**

Ejemplo: Análisis precio vivienda

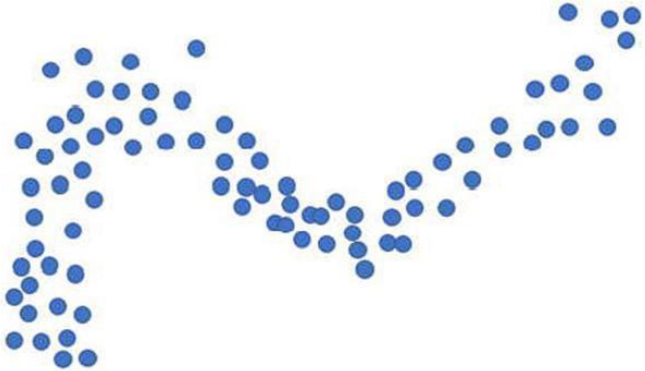


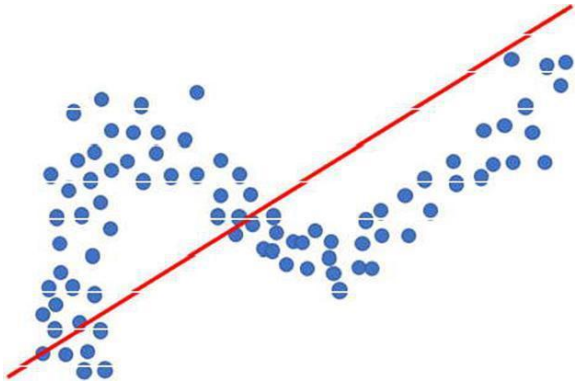
1. Las características formarán el eje X
2. La salida o target el eje Y
3. Para calcular las distancias usamos una técnica denominada **gradiente descendiente**



Ejemplo II: Análisis por barrios

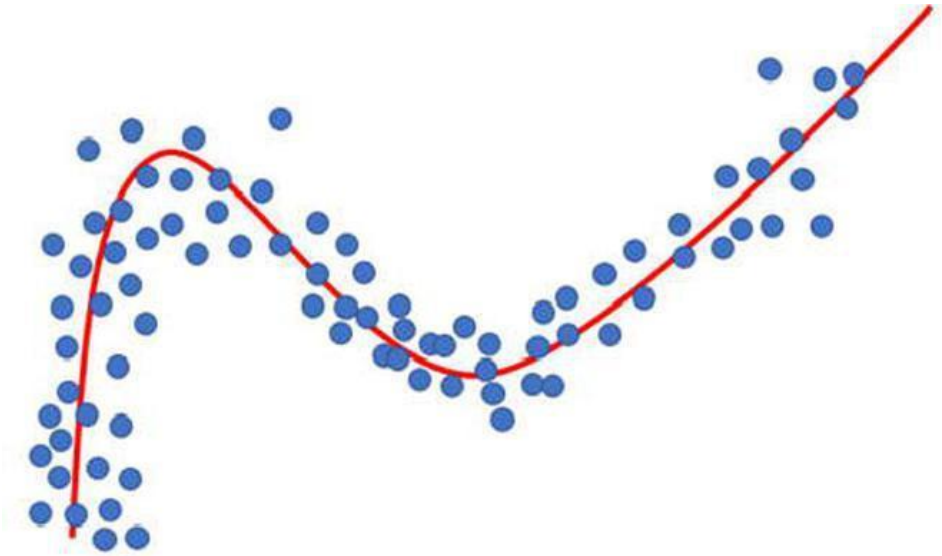
1. Ahora imaginemos que tenemos esta gráfica.
2. Aplicando regresión Lineal nos quedaría:





Ejemplo II: Análisis por barrios

1. Aplicando **Regresión polinómica**



Ejemplo III: Tienda online

1. Nuestro website vende zapatos multimarca. Tras una compra, la tienda nos envía un mail para disponer de un “feedback”. El cliente escribe su review en la sección de comentarios, algunas son positivas, otras negativas.
2. Problemas a resolver:
   a. Reviews negativas son enviadas al fabricante para mejorar el calzado
   b. Zapatos con una alta tasa de positivos, son posicionados al principio

Fase: Clasificar comentarios

Positivos	Negativos
A1: Nice quality!! I love it	B1: Bad material. Not fit
A2: Great product.	B2: Hate this product. Packing was also pathetic.
A3: Got one for my father. He loved it.	B3: Never buy this product.

Ejemplo III: Tienda online

Analizamos el dataset y descubrimos que los comentarios que contienen la palabra “love” es mucho más probable que sea comentario positivo. Establecemos este parámetro como regla y encontramos que el 60% contienen “love”son positivos y que tan solo el 10% de los comentarios negativos contienen “love”.

Para futuros comentarios podemos establecer si son positivos o negativos en función de las palabras que contengan. Este método es conocido como clasificador **Naïve Bayes**

Clasificación	Positiva	Negativa
Love	60%	10%
Great	45%	7%
Nice	36%	8%
Bad	4%	62%
Pathetic	2%	23%

Ejemplo IV: Lectura revistas

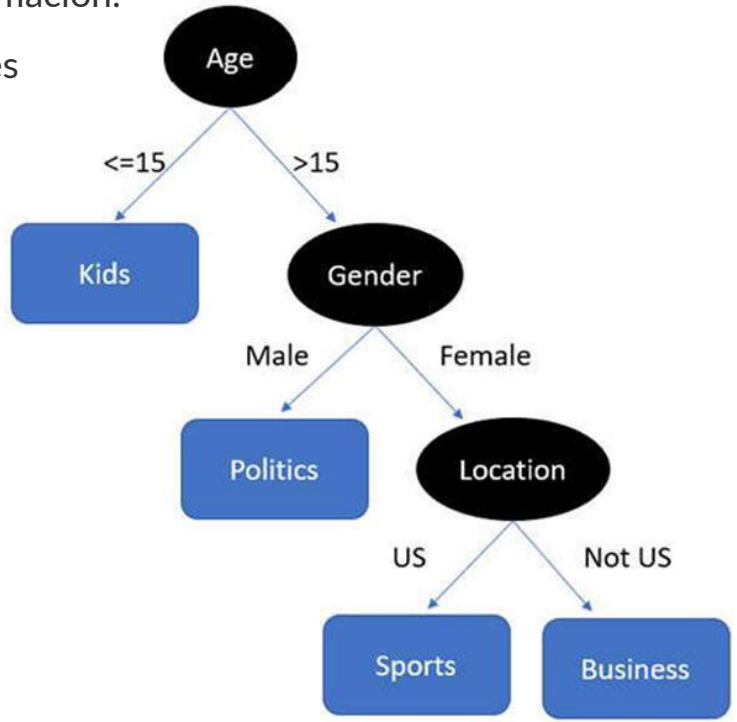
Disponemos este dataset :

Edad	Género	Localización	Revista
21	Female	USA	Sports
15	Male	USA	Kids
37	Male	España	Politics
42	Female	UK	Business
32	Female	USA	Sports
14	Female	España	Kids
53	Male	España	Politics

Ejemplo IV: lectura revistas

Árbol de decisiones, los datos nos aportan mucha información:

- Los menores de 15 leen revistas para adolescentes
- Los hombres leen sobre política
- El deporte solo interesa en USA



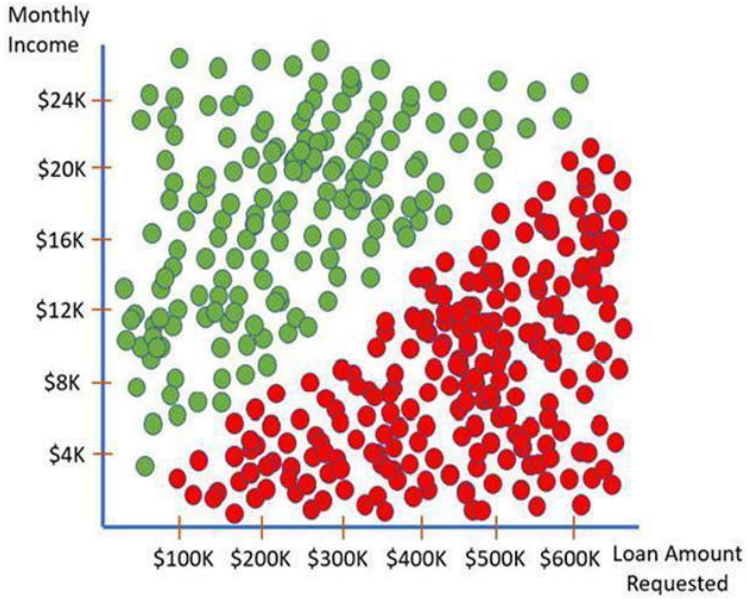
Ejercicio IV: Construye tu árbol

Existen múltiples caminos para crear árboles de decisión.

Objetivo: Construye un **árbol** distinto al anterior

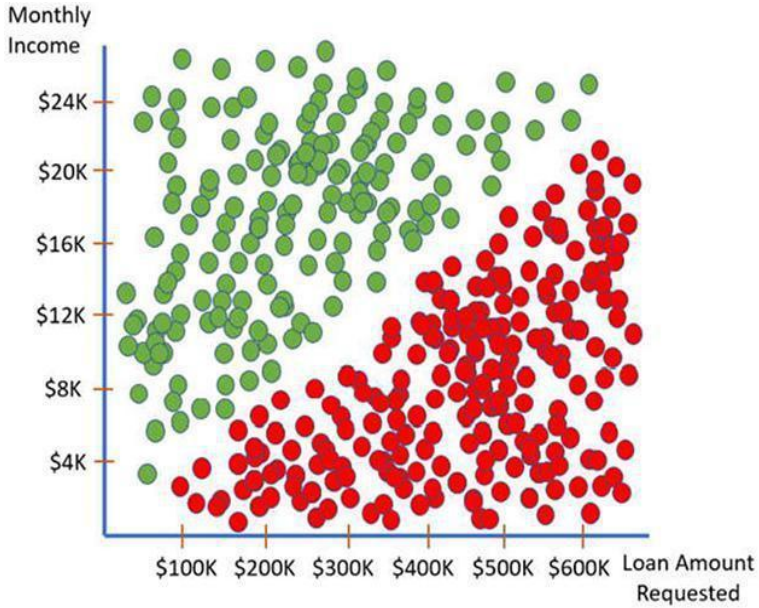
Ejemplo V: Préstamos bancarios

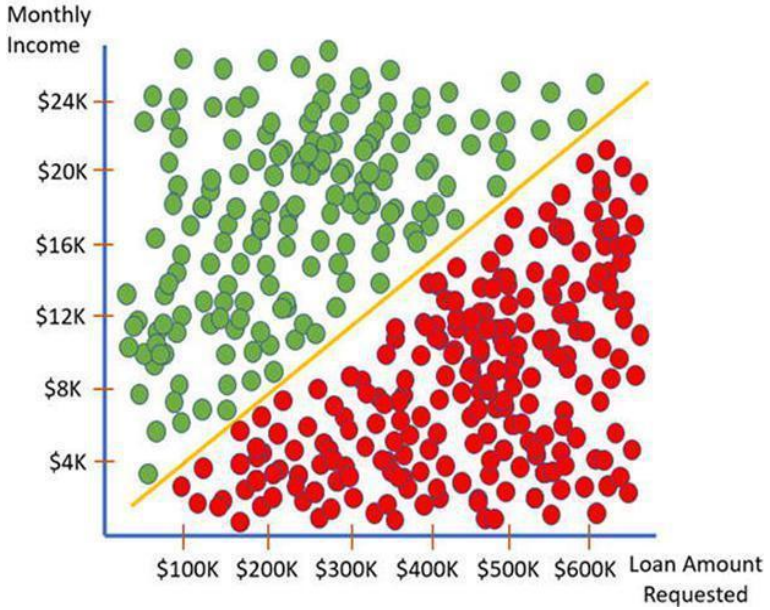
1. Una persona con ingresos mensuales de $ 20K, solicita $ 100K, el banco puede otorgar el préstamo, conoce fuente de ingresos.
2. Una persona ingresa $ 3K y solicita $ 600K, el banco rechaza la solicitud



Ejemplo V: Préstamos bancarios

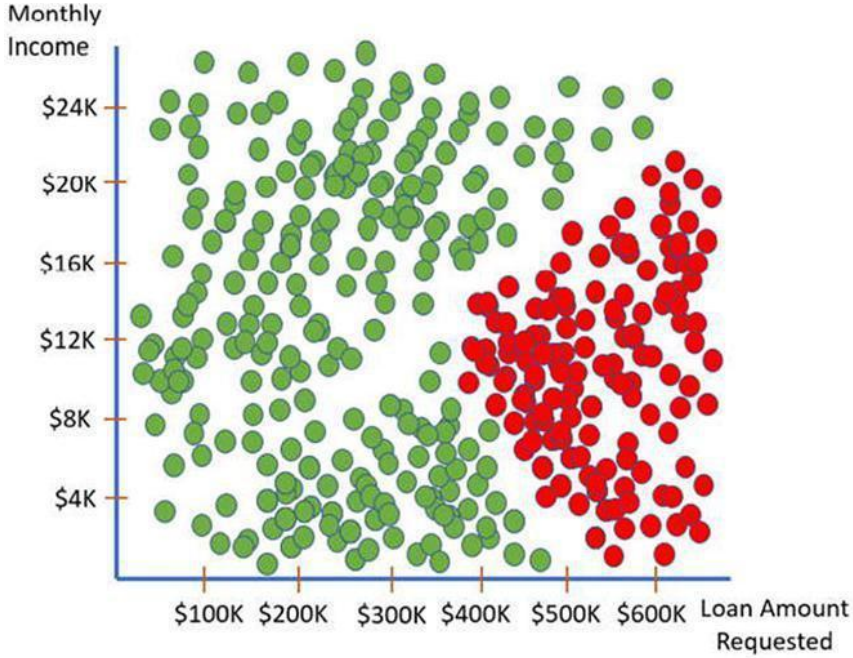
Hemos ajustado una línea para separar los puntos, utilizando el algoritmo gradiente descendiente usado en regresión. Aquí, la variable objetivo es categórica en lugar de continua. Esta técnica se conoce como **regresión logística.**





Ejemplo VI: Préstamos bancarios II

Un nuevo manager cambia los parámetros y toda petición comprendida entre $100K y $300K no tienen riesgo para cualquier usuario, cambia las reglas y ahora los datos tienen este aspecto:

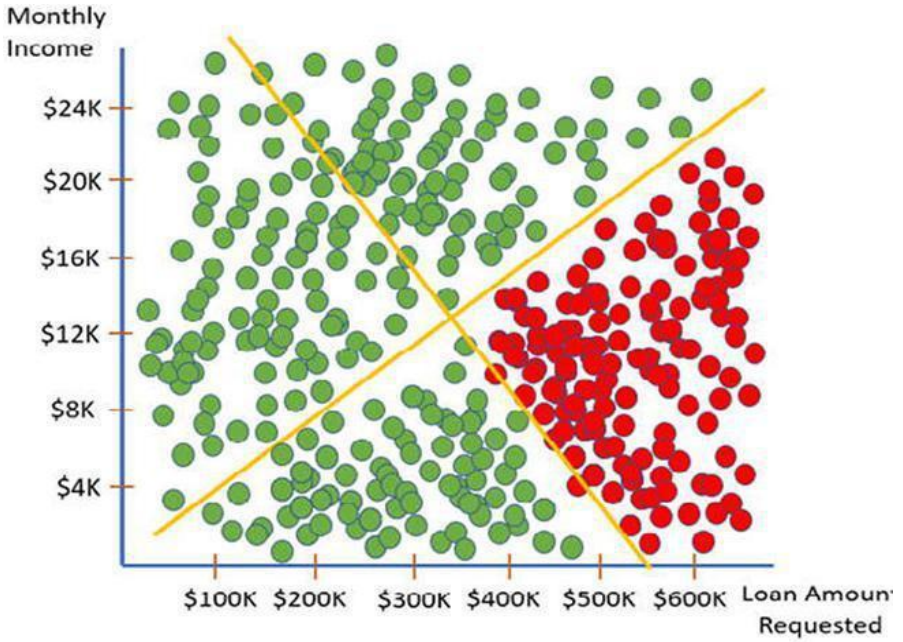


No podemos aplicar una división como la anterior, y ahora, ¿**cómo resolvemos el problema**?

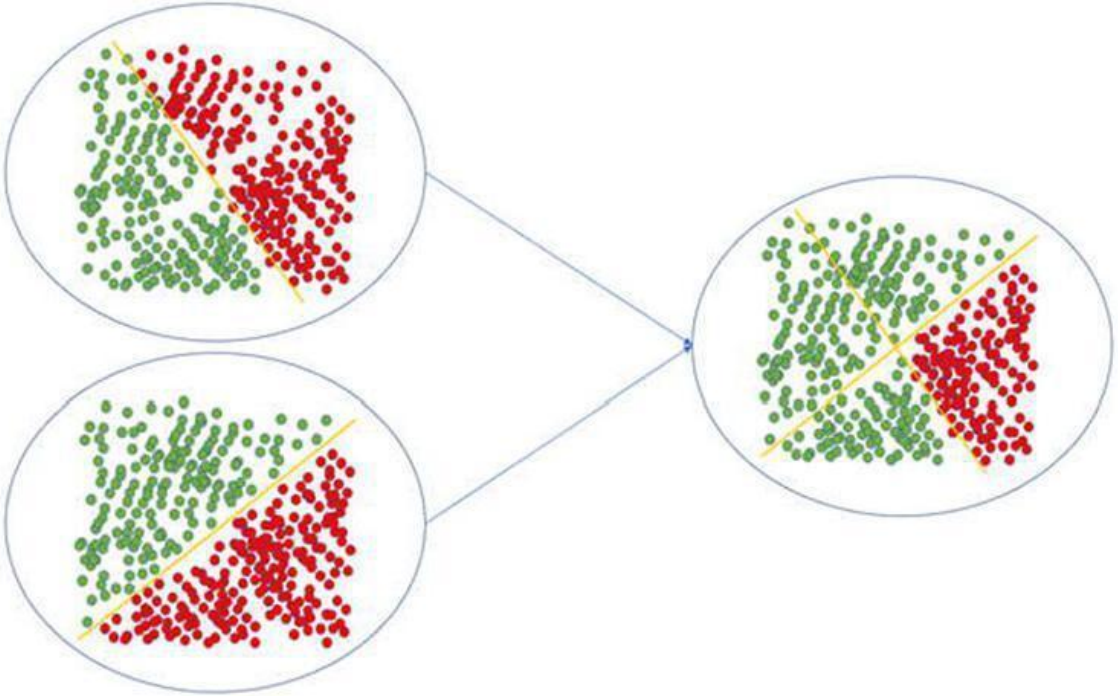
Ejemplo VI: Préstamos bancarios II

Solución: **Redes Neuronales**

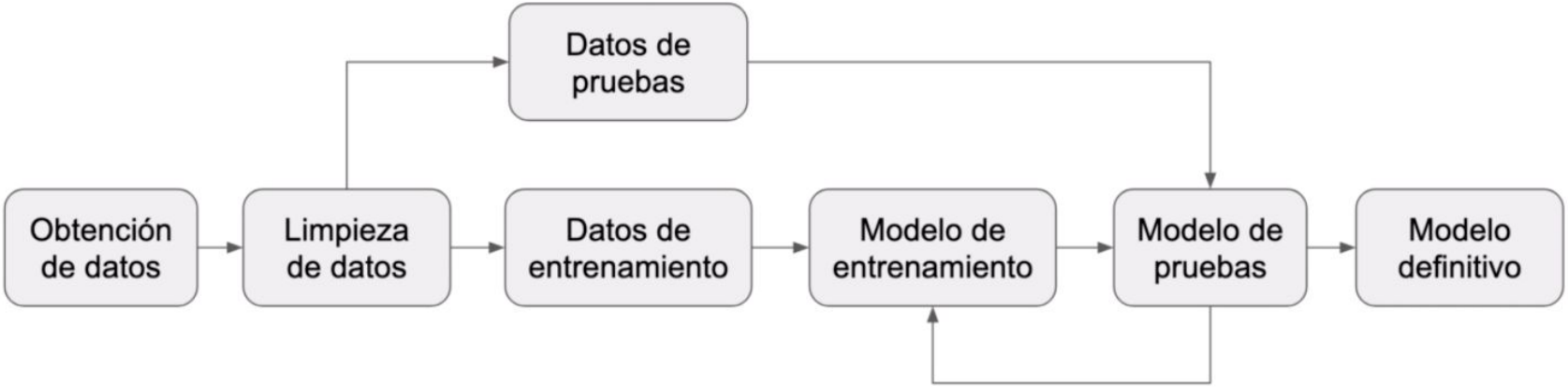
La red neuronal se basa en el concepto de neuronas de nuestro cerebro. En el cerebro, las neuronas recopilan la información y la pasan a otras neuronas. Según la entrada de neuronas anteriores, la siguiente neurona decide una nueva salida. También reenvía su información a otras neuronas. Por último, al procesar diferentes neuronas, nuestro cerebro toma la decisión.



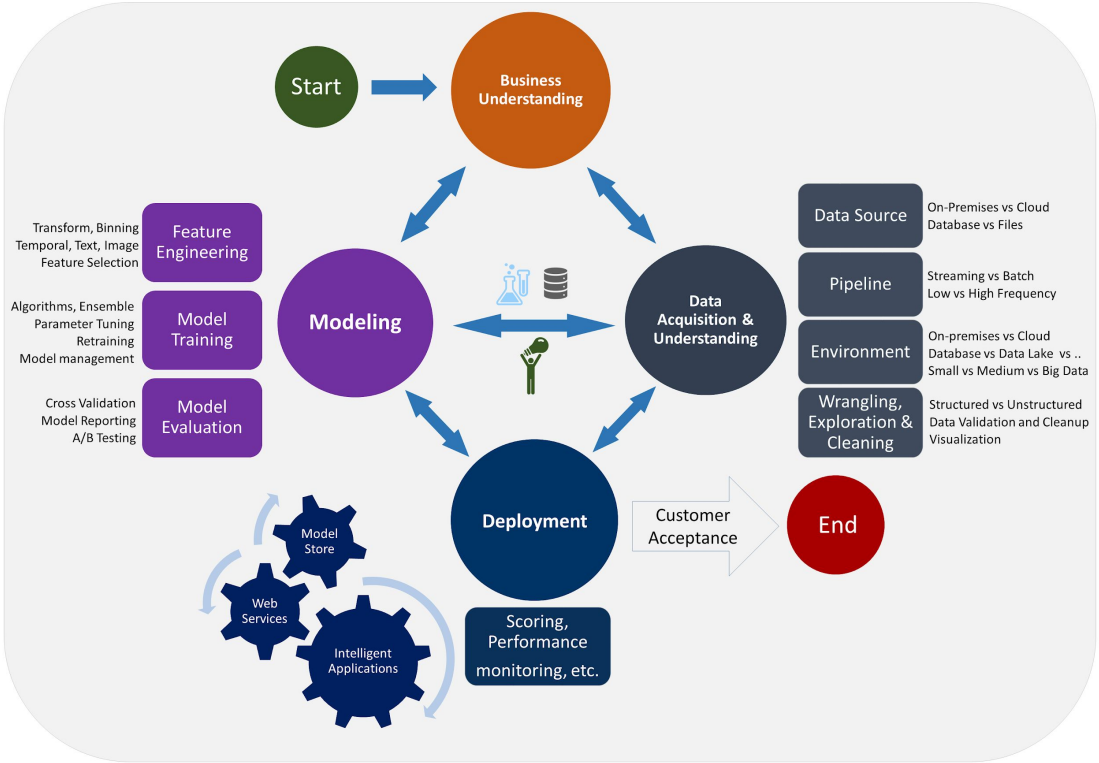
Ejemplo VI: Aspecto de la red neuronal



Proceso simple Machine Learning



Proceso completo Machine Learning



Aprendizaje Supervisado

- Aprendizaje Inductivo con variable objetivo definida y presente en los datos de entrada.
- Predicción numérica Vs Clasificación.
- Objetivo: Inferir un modelo que ante nuevas entradas haga predicción de la salida (numérica / etiqueta)

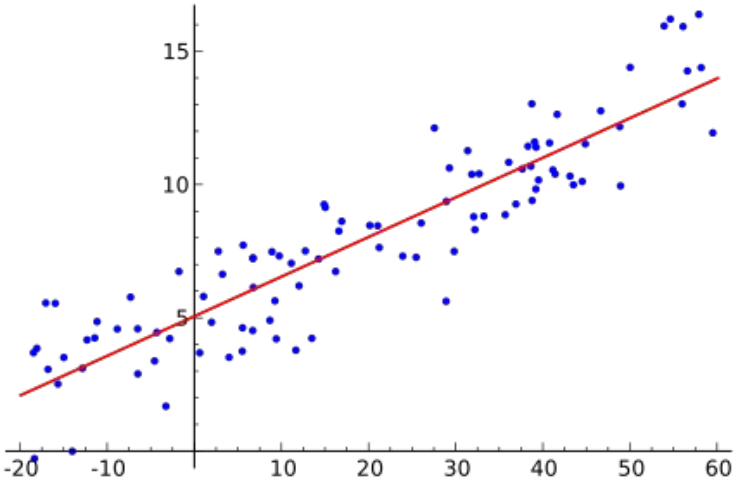
Supervisado Vs no Supervisado

Los sistemas de aprendizaje **supervisados** son aquellos en los que dentro del conjunto de datos se conoce el valor objetivo o **target**.

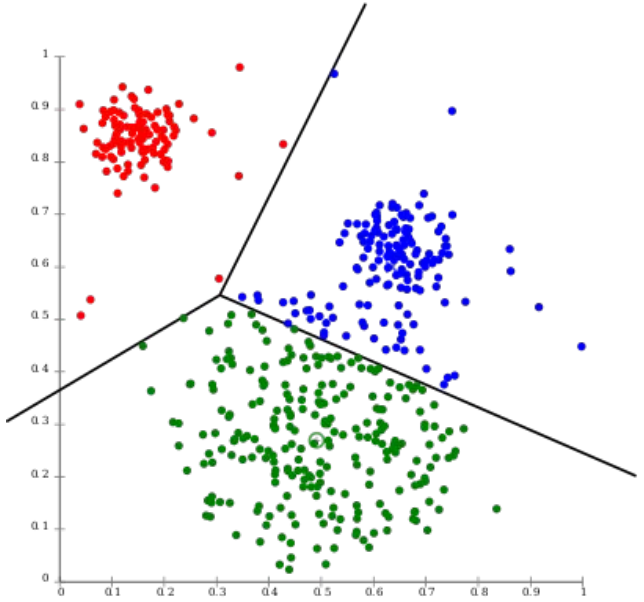
Los sistemas de aprendizaje **no supervisados** son aquellos en los que no se conoce su salida o **target.**

Aprendizaje Supervisado: Tipos

Regresión

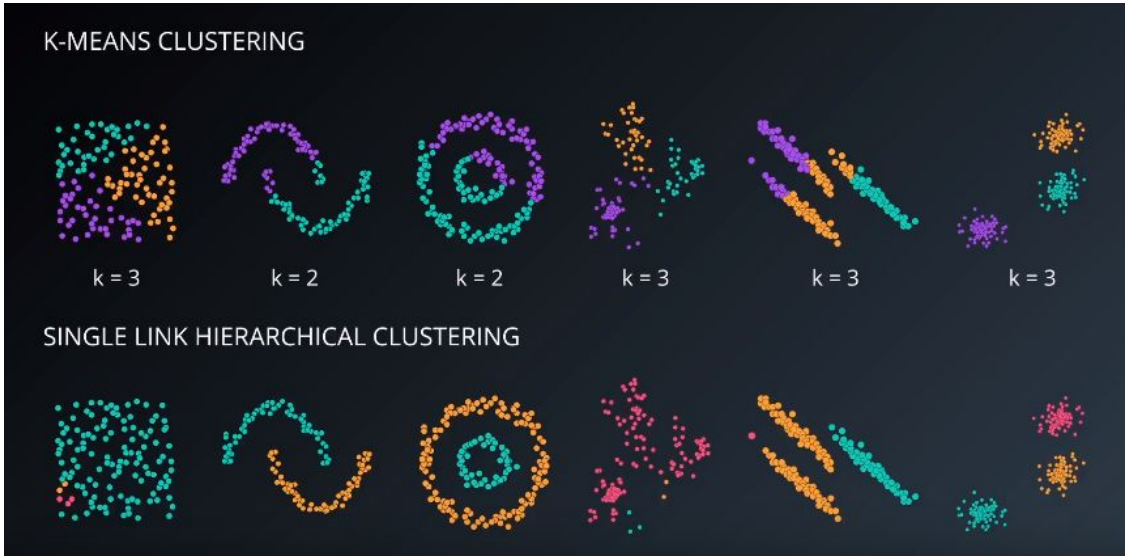


Clasificación



Aprendizaje no Supervisado: Tipos

Agrupamiento



Reducción de la dimensionalidad

Ámbitos de aplicación

Sanitario

- Detección de enfermedades.
- Predicción de aparición de enfermedades.
- Análisis de la actividad postural.
- Predicción de estancia hospitalaria.
- Análisis de señales cerebrales.
- Clasificación de secuencias de ADN.

Energético

- Estimación de demanda energética
- Predicción del clima.

Retail

- Estimación de la demanda.
- Fijación de precios.
- Predicción del comportamiento de los compradores.
- Segmentación de clientes.
- Búsqueda de clientes basándose en comportamientos en las redes sociales
- Optimización de la usabilidad Web/Móvil.
- Optimización de las horas que maximizan el impacto en redes sociales de una campaña de marketing.

Ámbitos de aplicación II

Logística

- Predecir de fallos en equipos tecnológicos.
- Aplicación en data analytics a partir de sensores.
- Mantenimiento predictivo en aeronáutica
- Análisis de telemetría en coches.
- Predicción de retrasos de aviones.
- Predecir el tráfico urbano.
- Vehículos autónomos.

RRHH

- Análisis de empleados más rentables.

Financiero

- Detección de fraude en transacciones electrónicas.
- Predicción de riesgos financieros.
- Predicción de recesión.

Seguridad

- Detectar intrusiones en redes.
- Detección de objetos.
- Sistemas Anti-spam.
- Detectar software malicioso.

Lab1 - Scikit-Learn

Pipelines

- Combinación de estimadores
- Ejecución secuencial
- Muy útil para reducir código

Ver notebook!

Evaluación y validación de modelos

Introducción

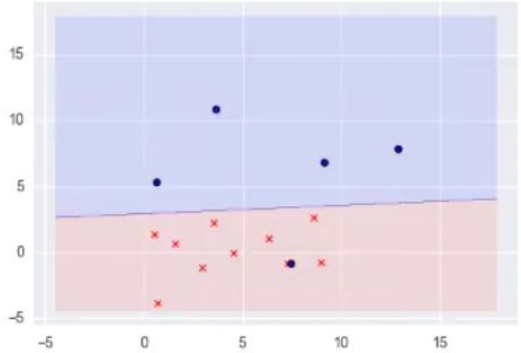
- Demostrar cómo de buenos son nuestros modelos
- Hemos aprendido modelos predictivos:
    - Regresión Lineal para problemas numéricos
    - Regresión Logística para Clasificación

Los algoritmos usados para la estimación del modelo en regresión lineal devuelven la configuración de parámetros **óptima**. La configuración aprendida para un modelo no ajusta a la perfección. Por ello siempre se obtendrá un **error predictivo.**

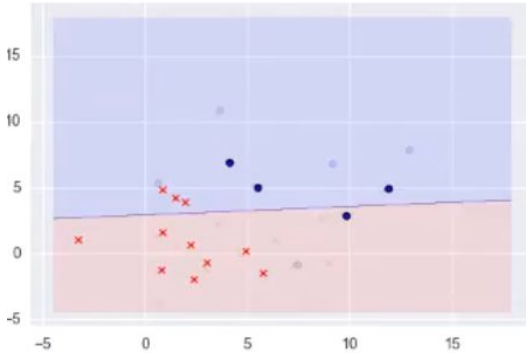
Cabe esperar, **no siempre es así,** el error aumenta cuando se usa el modelo para predecir datos que no han sido utilizados.

Introducción II

Ej. Regresión Logística



Error (datos originales) = 6.6 %



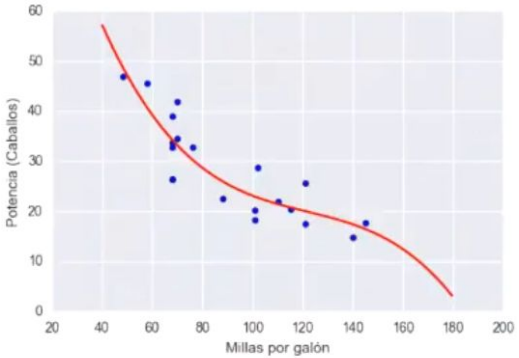
Error (nuevos datos) = 26.6 %

Introducción III

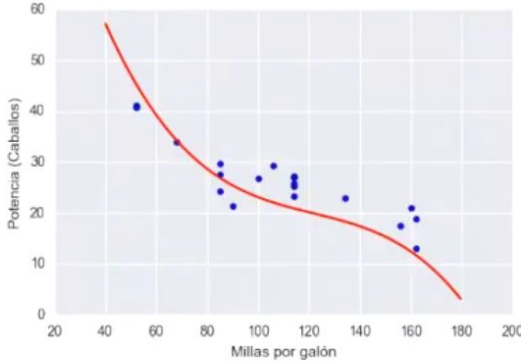
Podemos mejorar cuanto se ajusta el modelo a los datos de entrenamiento (ej. creando características polinómicas)

A medida que ajustamos, el modelo tiende a **disminuir el error**. Y posiblemente también en nuevos datos.

Ej. Regresión Lineal con características polinómicas (grado 3)



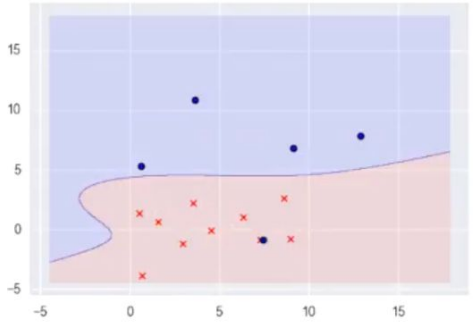
Error cuadrático medio = 19.13



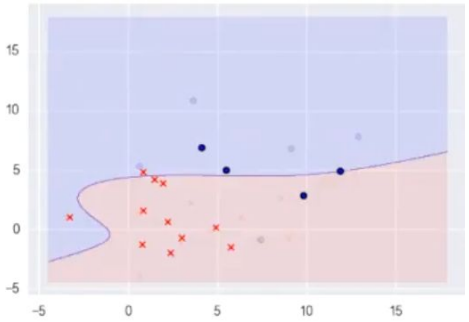
Error cuadrático medio = 23.28

Introducción IV

Ej. Regresión logística con características polinómicas (grado 3)



Error = 6.6 %



Error = 20 %

Evaluar y Validar: Objetivo

- **Conjunto de entrenamiento**, dataset utilizado para aprender el modelo.
- En general el rendimiento de un modelo sobre datos nuevos es **peor.**
- La capacidad de un modelo de no degradar su rendimiento se denomina **generalización.**
- **Rendimiento del modelo**, capacidad predictiva del mismo al aplicarlo sobre nuevos casos.
- La **evaluación** del modelo ha de hacerse sobre datos no utilizados en el entrenamiento.

Evaluar un modelo

Se cuantifica **el rendimiento** de un modelo de aprendizaje supervisado respecto a un conjunto de datos mediante: (las métricas más comunes)

- Regresión: Error cuadrático medio.

$$MSE(m, X) = \frac{1}{N} \sum_{i=1}^{N} \left(m(x^{(i)}) - y^{(i)} \right)^2$$

  - nota: En scikit es R^2 que es un factor entre 0 y 1 que evalúa cómo de bien captura la variación de datos.

- Clasificación: ERROR o 1 - *Tasa de aciertos*

$$ERROR(m, X) = \frac{1}{N} \sum_{i=1}^{N} fallo\left(m(x^{(i)}), y^{(i)} \right)$$

$$fallo\left(m(x^{(i)}), y^{(i)} \right) = \begin{cases} 0 & \text{si } m(x^{(i)}) = y^{(i)} \\ 1 & \text{si } m(x^{(i)}) \neq y^{(i)} \end{cases}$$

Evaluar un modelo: Test (Holdout)

Técnica más recomendable cuando hay suficientes datos.

Consiste en dividir el conjunto original de datos en:

- Conjunto de **entrenamiento** a partir del cual se estima el modelo
- El conjunto de **test** evalúa el mismo

Esta división suele ser en proporciones 70 - 30 | 80 - 20

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, y, test_size=0.3, random_state=11)
model.fit(X_train, Y_train)
acc = model.score(X_test,Y_test)
```

Evaluar un modelo: Buenas prácticas

Importante **barajar** o desordenar el conjunto original antes de hacer la partición (variables ordenadas)

En Scikit-Learn se baraja por defecto (`shuffle=True`).

También **estratificar** las particiones. Es decir la distribución de los valores de la variable clase sea la misma en las particiones. Es importante para otros métodos de validación con más particiones.

Scikit-Learn proporciona la función
`sklearn.model_selection.StratifiedShuffleSplit`

Evaluar un modelo: Cross Validation

Modelo de regresión logística (ej. wisconsin) la semilla siempre tendré la misma división.

```
lr = LogisticRegression()
acc = np.empty(10)

for i in range(0,10):
    X_train, X_test, Y_train, Y_test =
    train_test_split(X, y, test_size=0.3,
        random_state=i*i)
    lr.fit(X_train, Y_train)
    acc[i] = lr.score(X_test, Y_test)
    print('{:.4f}' .format(acc[i]))

print('Acc: {:.4f} Std: {:.4f}'
      .format(np.mean(acc), np.std(acc)))
```

```
0.9649
0.9532
0.9181
0.9415
0.9298
0.9298
0.9298
0.9298
0.9240
0.9240
Acc: 0.9345 Std: 0.0138
```

Evaluar un modelo: Cross Validation II

K-Validación cruzada (Cross Validation, K-CV)

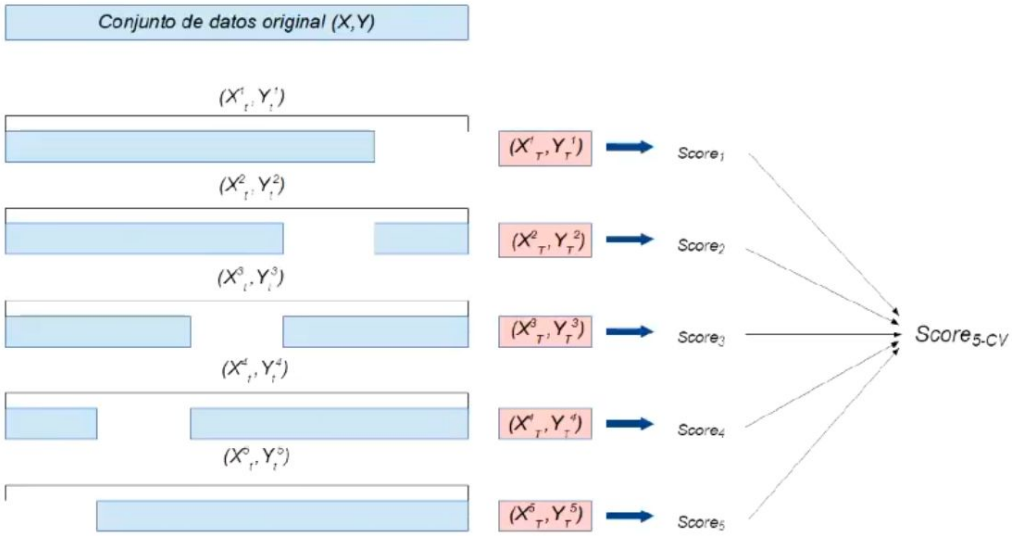
- Obtiene *K* conjuntos del mismo tamaño y disjuntos de test
- Devuelve la media del error obtenido en las *K* iteraciones

En K-CV es más importante hacer la **validación cruzada estratificada** para asegurar que la proporción de elementos es homogénea a cada subconjunto.

El caso de K-CV en el *K* es igual al número de registros *X* es conocida como **leave-one-out** o LOOCV

Cross Validation III: Ejemplo & Ejercicio

Ejemplo gráfico, 5 conjuntos de train disjuntos y 5 de test



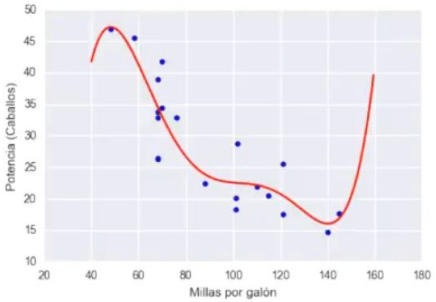
Overfitting: Sobreajuste

Cuanto más forcemos que el modelo ajuste los datos de entrenamiento:

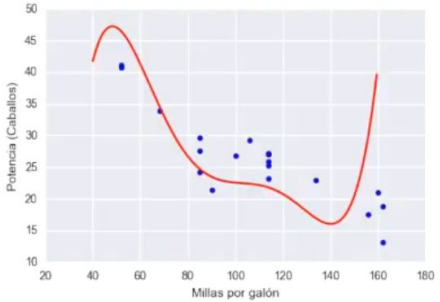
1 la generalización del clasificador será mejor sobre los datos de entrenamiento pero… 2 al aplicarlo sobre nuevos datos, podría degradarse mucho.

Ej. R. logística forzando ajuste características polinómicas (grado 5)

Sobreajuste, cuando el modelo se ajusta demasiado al conjunto de datos de entrenamiento y por ello no **generaliza** bien la predicción de nuevos casos



Error cuadrático medio = 18.19



Error cuadrático medio = 152.99

Validar modelo: Tasa de acierto

Evaluación sensible a la distribución:

- A veces clases no balanceadas, si el tamaño del conjunto de datos es *N* y número de clases es *c* no tiene porque haber ***N/c*** instancias de cada clase
- Puede suponer un problema en la evaluación final

Ejemplo:

- Se han estudiado 10000 pacientes, de los cuales 9900 son sanos, y 100 tienen una enfermedad grave.
- Un clasificador c predice que todos los pacientes están sanos.
- La tasa de acierto es 9900/1000 = 99%

¿Es c un buen clasificador?

Validar modelo: Matriz de confusión

		Clase real	
		Sí	No
Clasificado como	Sí	Verdadero Positivo (TP)	Falso Positivo (FP)
	No	Falso Negativo (FN)	Verdadero Negativo (TN)
Total		Ej. Positivos (P)	Ej. Negativos (N)

Se define:

- Tasa de acierto (*accuracy*): $accuracy = \frac{TP+TN}{P+N}$
- Tasa de verdaderos positivos TPR (*sensitivity* o **recall**): $TPR = \frac{TP}{TP+FN} = \frac{TP}{P}$
- Tasa de verdaderos negativos TNR (*specifity*): $TNR = \frac{TN}{TN+FP} = \frac{TN}{N}$
- Tasa de falsos positivos FPR (*negative error*): $FPR = \frac{FP}{TN+FP} = \frac{FP}{N}$
- Valor positivo predictivo (**precision**): $PPV = \frac{TP}{TP+FP}$

Se cumple que FPR = (1 - specificity)

Obtener métricas Scikit Learn

El método **score()** devuelve la tasa de acierto/coeficiente de determinación (R^2) de los modelos de clasificación/regresión implementados en Scikit Learn.

El paquete sklearn.metrics contiene funciones que reciben los valores verdaderos y los predecidos por el modelo (entre otros) y calculan las medias correspondientes.

Ejemplo de Métricas para clasificación

```
import sklearn.metrics as metrics
print('Precisión:',  metrics.precision_score(y,y_pred))
print('Recall: ', metrics.recall_score(y,y_pred))
print('F-Score: ', metrics.f1_score(y,y_pred))
print('Matriz de confusión: ', metrics.confusion_matrix(y,y_pred))
```

Métricas: |c| > 2

Las métricas vistas son para casos binomiales, puede haber variables clase que tengan más de 2 posibles estados. |c| > 2

todas son vistas como resultados de interés.

Existen la media Micro y la media Macro, para resolver esta casuística disponemos del parámetro *average*.

Resumen:

El esfuerzo en mejorar nuestro clasificador puede crear sobreajuste y llevarnos al desastre.

Para estimar la **generalización** de un modelo podemos usar **K-CV.**

Existen muchas métricas de clasificación.

Selección de variables: Objetivo

- Reducción de la dimensionalidad
- Obtener el subconjunto mas optimo de variables
- Menor varianza, porque se aprende un modelo que descarta variables redundantes e irrelevantes
- que permita predecir la clase mejor que cuando se utilizaban todas las variables
- sino es posible que sea mejor, con la misma tasa de aciertos o métrica para validar el modelo
- Si pierde un poco, que merezca la pena quedarnos con un 10% de variables

Selección de variables: Definición

- Regularización es sencillo de aplicar en Regresión lineal y logística.
- Existen modelos **no regularizables.**
- Estos modelos también pueden beneficiarse de **disminución del sobreajuste.**
- Ejecutan algoritmos de selección de variables que ayudan a descartar características innecesarias.
- Estos métodos también son aplicables a **modelos regularizables**

Selección de Variables Supervisada (FSS)

Proceso por el cual se identifican variables de entrada relevantes para la predicción de la clase u objetivo. Un buen algoritmo FSS eliminará los atributos más irrelevantes y/o redundantes para la clase objetivo

Selección de variables: Ventajas

Al identificar el subconjunto (casi) óptimo de variables predictivas se obtienen:

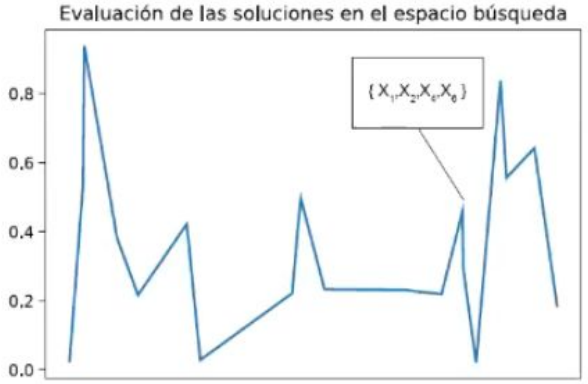
1. Simplicidad del modelo, reducción **sobreajuste**
2. Aumenta poder de **generalización**
3. Entrenamiento modelo **más rápido**
4. Modelo creado es más **interpretable**, ayudando así al experto del dominio
5. No se altera la **representación** original, mantiene semántica de variables

Selección de variables: Espacio búsqueda

Si tenemos ***n*** variables, entonces posibles combinaciones es 2^n

El conjunto de todas las combinaciones es denominado espacio de búsqueda y la **misión** del FSS es encontrar la mejor **combinación.**

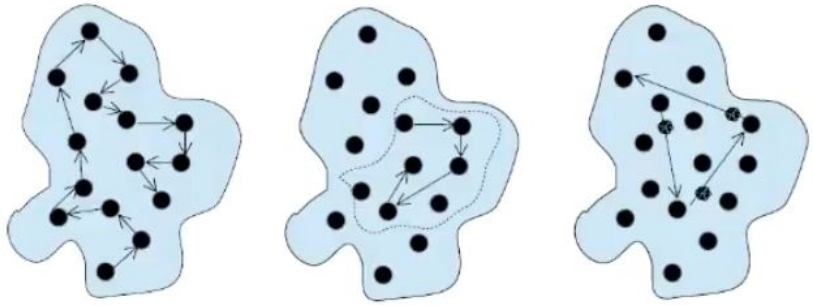
Si se encuentra la mejor, hemos encontrado el **óptimo global**, sino, **óptimo local**



Selección de variables: Tipos FSS

Dependiendo de la **exploración del espacio**, se definen 3 tipos:

- **Exhaustiva**: Evaluar las 2^n combinaciones posibles en cualquier dirección (anchura y profundidad) Sino se para el algoritmo, termina explorando todo el espacio de búsqueda
- **Secuencial**: Basado en heurística determinista que elimina o añade variables candidatas al subconjunto final según la función de evaluación estime si merece la pena o no.
- **Estocástica**: Basado en una meta heurística no determinista que aplica aleatoriedad al espacio de búsqueda.

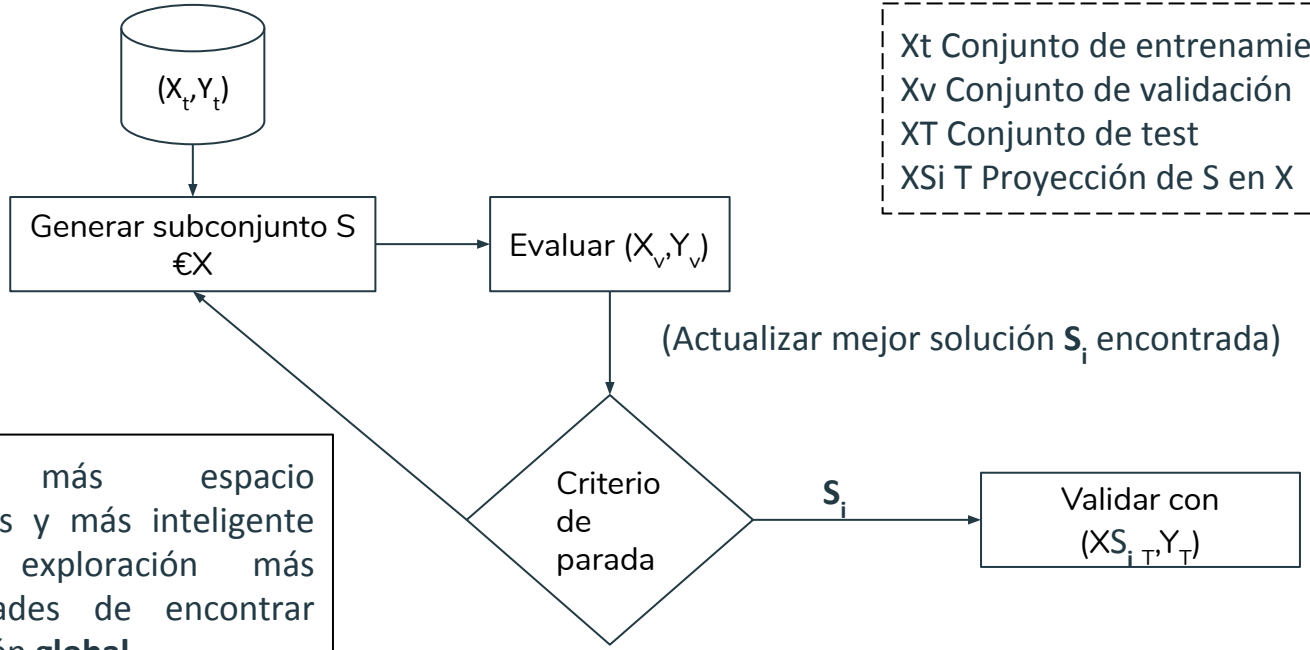


Selección de variables: Proceso Genérico

Independientemente del tipo de recorrido, las **fases** para algoritmos FSS son:

1. **Generar una solución**: A partir de la solución anterior, crear otra combinación de variables
2. **Evaluar la solución**: Saber si esta solución es mejor que la anterior
3. **Criterio de parada**: Decidir si es el momento de finalizar la búsqueda.
4. **Evaluar**: Testear el modelo con nuevos datos usando solo atributos seleccionados

Selección de variables: Algoritmo



Xt Conjunto de entrenamiento
Xv Conjunto de validación
XT Conjunto de test
XSi T Proyección de S en X

Cuanto más espacio exploremos y más inteligente sea la exploración más probabilidades de encontrar una solución **global**

Selección de variables: Evaluación

Según el **tipo** de evaluación:

1. **Filter**: Evalúa propiedades intrínsecas de los datos
2. **Wrapper**: Mucho más costosos en tiempo con mejores resultados
3. **Híbrido**: Utilizan evaluación subconjuntos wrapper y generan candidatas mediante filter. Buen compromiso entre velocidad y calidad final.
4. **Embebido**: La selección se realiza como parte del proceso de aprendizaje del modelo.

Lab - Selección de Variables

Aprendizaje Inductivo

Se basa en el descubrimiento de patrones a partir de ejemplos. Parte de casos particulares y obtiene casos generales (modelos o reglas) y es el tipo de aprendizaje más común en el aprendizaje automático.

Estos ejemplos pueden ser **positivos y negativos**

De entre todos los mecanismos en el aprendizaje inductivo, destacan los **árboles de decisión.**

Aprendizaje Inductivo: Árboles decisión

Un árbol de decisión está formado por un conjunto de **nodos de decisión** y de **nodos-respuesta** (hojas):

- Un **nodo de decisión** está asociado a uno de los atributos y tiene 2 o más ramas que salen de él, cada una de ellas representando los posibles valores que puede tomar el atributo asociado.

- Un **nodo-respuesta** está asociado a la clasificación que se quiere proporcionar, y devuelve la decisión del árbol con respecto al ejemplo de entrada.

Lab - Árboles de Decisión