

Machine Learning Práctico: Detección de Diabetes

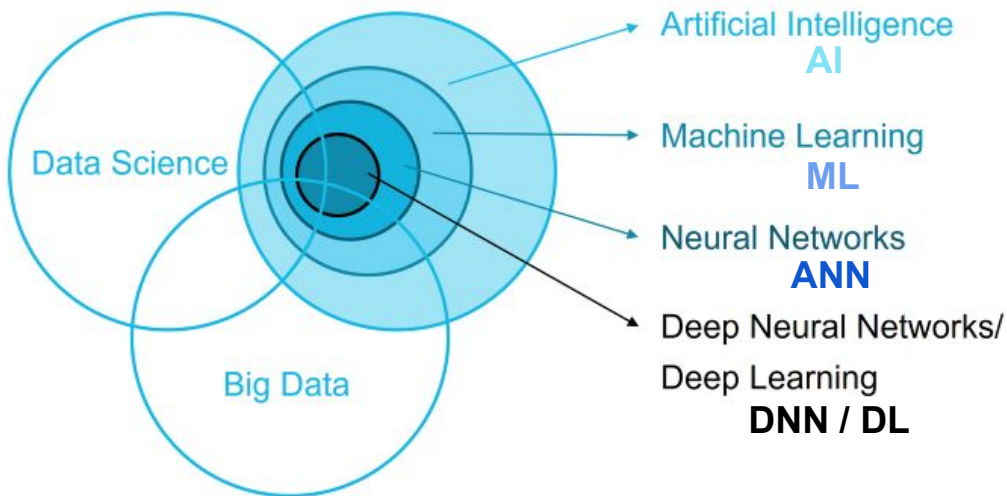


ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

¿Qué es machine learning?

¿Qué es Machine Learning?



Inteligencia Artificial (AI): Técnica con la que se pretende que las máquinas imiten el comportamiento humano. (¿Inteligencia?)

Aprendizaje de Máquina o Aprendizaje automático (ML): Rama de la Inteligencia Artificial que utiliza métodos estadísticos para que las máquinas aprendan de los datos (experiencia).

Redes Neuronales Artificiales (ANN): Rama del Aprendizaje de Máquina. Es un sistema computacional inspirado en neuronas biológicas.

Redes neuronales profundas o Aprendizaje profundo (DNN): Conjunto de técnicas poderosas para el aprendizaje de redes neuronales (Y sí, redes neuronales más profundas)

Machine Learning, Arthur Samuel (1959): Campo de estudio que dota a las computadoras con la habilidad de aprender sin ser explícitamente programadas.

Algoritmos de ML:

Algoritmos de Regresión

- Regresión Lineal (Linear regression)
- Regresión Logística (Logistic Regression)

Algoritmos basados en Instancia

- k vecinos más cercanos (k-Nearest Neighbor kNN)
- Mapa auto organizado (Self-Organizing Map)

Algoritmos de Árbol de Decisión

And Regression Tree CART)

learning)

- Árboles de Clasificación y Regresión (Classification And Regression Tree CART)
- Decisión de Árbol condicional (Decision tree learning)

Algoritmos Bayesianos

- Bosques Aleatorios (Random Forest)
- Clasificador bayesiano ingenuo (Naive Bayes)
- Gaussian Naive Bayes
- Multinomial Naive Bayes
- Red bayesiana (Bayesian Network)

Algoritmos de agrupación (Clustering)

- K-medias (K-Means)
- K-medianas (K-Medians)
- Agrupamiento jerárquico (Hierarchical Clustering)

Máquinas de vectores de soporte (SVM)

Algoritmos de Redes Neuronales

- Compuerta XOR
- Perceptrón
- Retro propagación (Back-Propagation)
- Red de Hopfield (Hopfield Network)
- Red perceptrón multicapa (MLP Multi Layered Perceptron)

Algoritmos de Aprendizaje Profundo

Networks CNN)

Memory Neural Networks LSTM)

Algoritmos de Reducción de Dimensión

- Análisis de componentes principales (Principal Component Analysis PCA)
- t-Incrustación estocástica de vecinos (T-distributed Stochastic Neighbor Embedding t-SNE)

Procesamiento del Lenguaje Natural (NLP)

Otros Algoritmos

- algoritmos de Aprendizaje por Reglas de Asociación
- Algoritmos de Conjunto
- Visión por computadora (Computer Vision)
- Sistemas de Recomendación
- Algoritmos genéticos (Genetic Algorithms)

Y MÁS 0_0

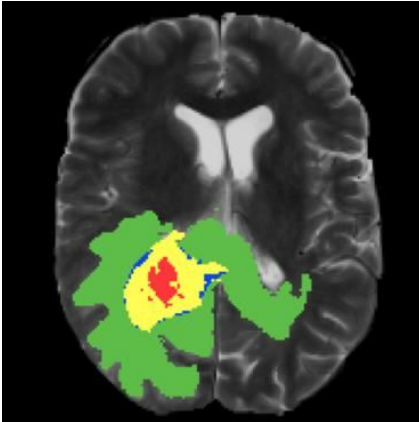
Machine Learning ¿usarlo o no usarlo?

Modelo tradicional

- Solución rápida que se puede implementar en pocos comandos
- Se conoce el problema y su solución
- Correctamente implementado, tiene una alta eficiencia

Suma de logaritmos de distinta base

$$\log_2 8 + \log_3 27 - \log_4 63 =$$



Modelo de machine learning

- ☐ Alta complejidad
- ☐ Se tienen **datos** referente al problema
- ☐ La solución puede tener un margen de error

Aprendizaje supervisado y no supervisado

Supervisado

Se cuenta con:

Un conjunto de datos X_m y de etiquetas Y_m

$$\{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$$

Se obtiene:

Un modelo predictivo

Predecir un comportamiento o clase para datos que no se han visto

$$Y = f(X)$$

Grupos de algoritmos:

- Clasificación
- Regresión

No supervisado

Se cuenta con:

Un conjunto de datos X_m

$$\{(x_1), (x_2), \dots, (x_m)\}$$

Se obtiene:

Un modelo descriptivo

Obtener más información sobre los datos:

- Patrones
- Estructuras
- Distribuciones

Grupos de algoritmos:

- Agrupamiento
- Reglas de asociación

Aprendizaje por refuerzo

Se cuenta con:

Agentes, ambientes, estados, acciones y recompensas

Se obtiene:

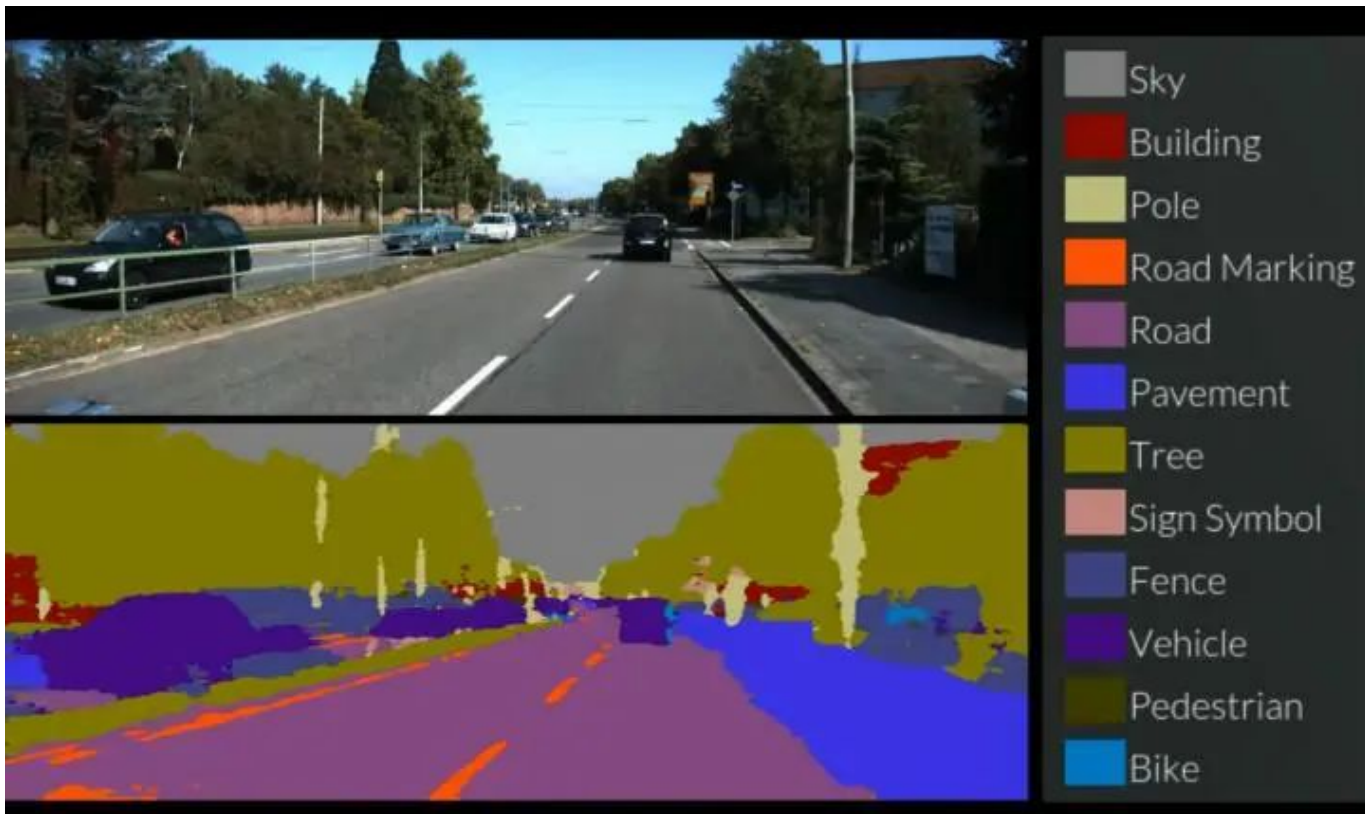
Un comportamiento del agente ante su ambiente.

Controlado por políticas auto aprendidas que le dicen qué acciones tomar para maximizar recompensas (positivas) y/o minimizar riesgos (negativas)

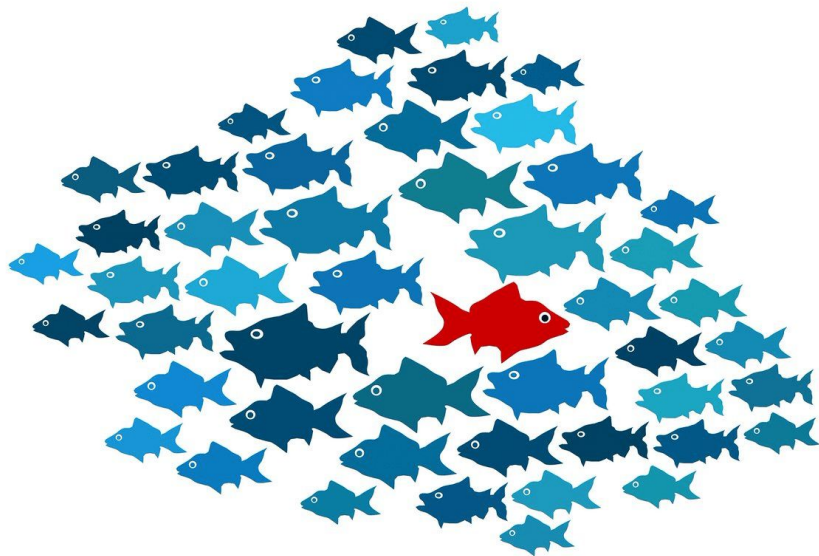
Aplicaciones:

- Juegos (de mesa, videojuegos)
- Brazos robóticos
- Vehículos autónomos
- Aprender a realizar una tarea compleja

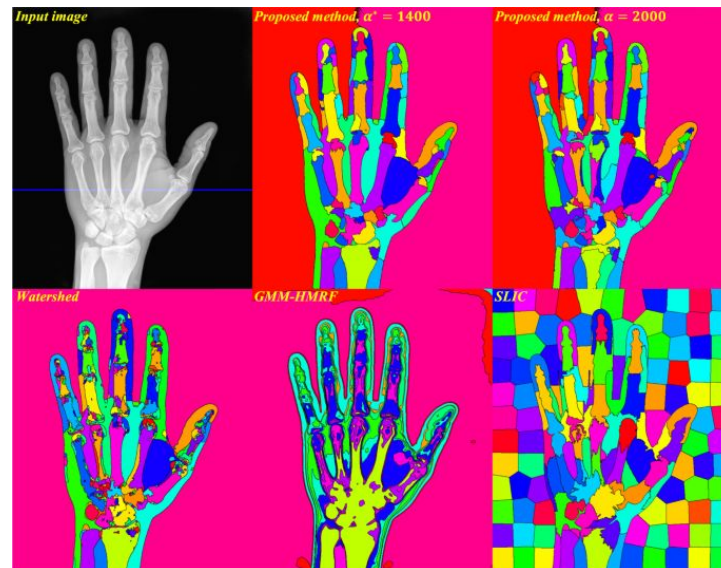
Ejemplo de aprendizaje supervisado



Ejemplo de aprendizaje no supervisado

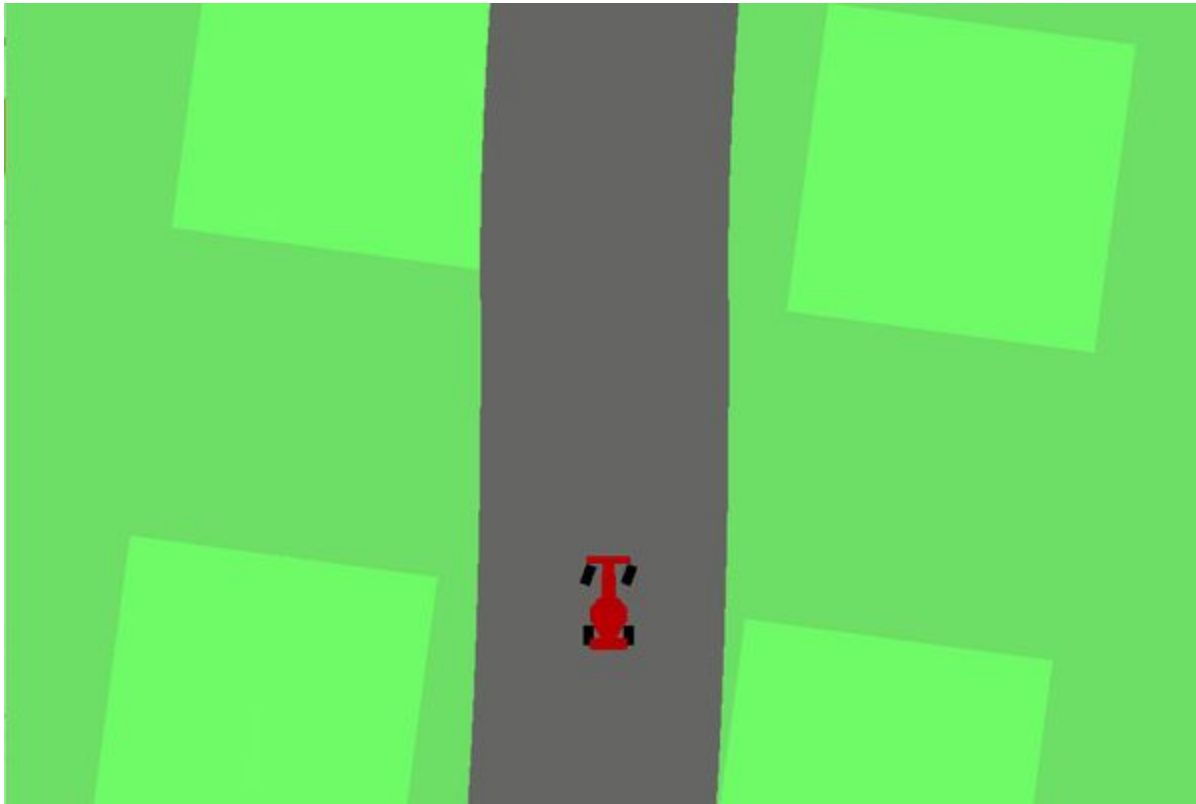


Detección de anomalías

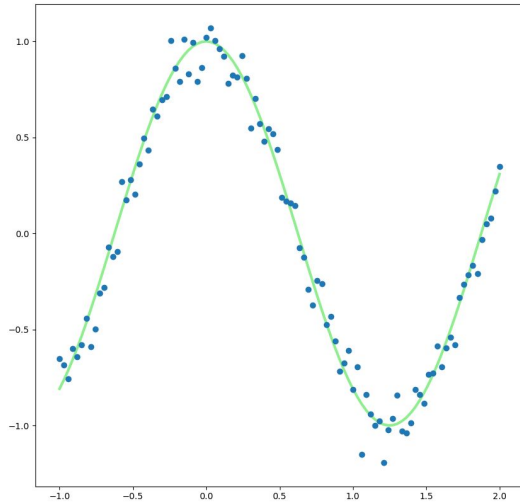


Segmentación y búsqueda de patrones

Ejemplo de aprendizaje por reforzamiento

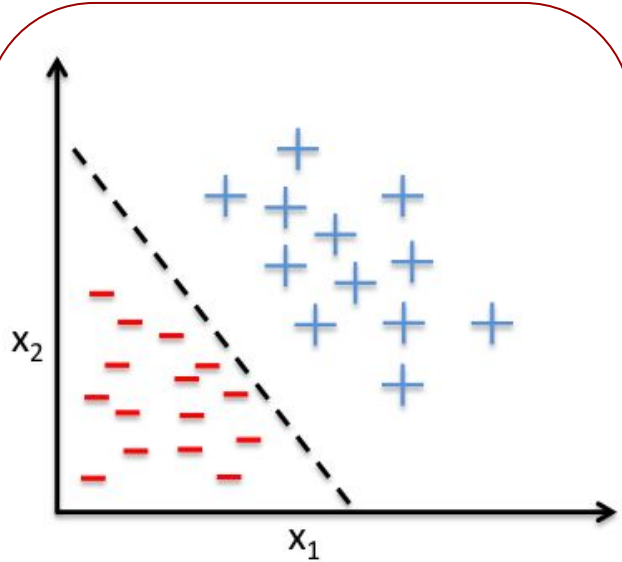


Regresión



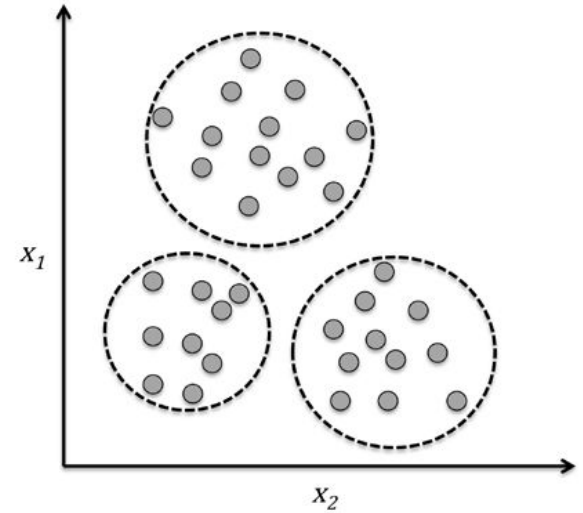
Tomamos el **conjunto de entrenamiento** para tratar de generar una función que se ajuste a los datos (predecir un valor numérico)

Clasificación



Separa en clases, esto es cuando la variable de salida es una categoría, como como rojo o azul o enfermedad y sin enfermedad.

Agrupamiento



Es la división de los datos en grupos con **rasgos** similares (sin conocer las clases).

Herramientas y ambientes de trabajo



ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

Python: Utilizamos Python 3 porque es un lenguaje fácil de aprender, fácil de entender y de utilizar, sobre todo para prototipado rápido

Colab: Colaboratory es un entorno de notebook de Jupyter gratuito que no requiere configuración y se ejecuta completamente en la nube.

Ambiente virtual miniconda: Conda es un sistema de gestión de paquetes de código abierto y un sistema de gestión del entorno que se ejecuta en Windows, macOS y Linux

Scikit-learn: Es una biblioteca de aprendizaje automático de software gratuito para el lenguaje de programación Python. implementa muchos algoritmos de Machine Learning eficientemente por lo que es un gran punto de entrada para aprender. Video.

Tensorflow: es una biblioteca más compleja para computación numérica distribuida utilizando grafos de flujo de datos.
TF - GPU

Deep Learning Framework	Release Year	Written in which language?	CUDA support
TensorFlow	2015	C++, Python	Yes
Keras	2015	Python	Yes
PyTorch	2016	Python, C	Yes
Caffe	2013	C++	Yes
Deeplearning4j	2014	C++, Java	Yes

[Numpy](#): Es un paquete para computación científica con Python, ver [tutorial](#). Es de utilidad práctica para manejo de arreglos de objetos N-dimensionales



[Pandas](#): Es una biblioteca de alto desempeño para Python, con manejo de estructuras de datos y herramientas de análisis de datos, ver [documentación](#).



[Matplotlib](#): Es una biblioteca para generar en pocas líneas: diagramas, histogramas, imágenes, espectros de potencia, diagramas de barras, diagramas de error, diagramas de dispersión y más. Ver [tutorial](#).



[GitHub](#): Es una plataforma para colaboración y control de versiones. Ver [ayuda](#).



Lectura recomendable: [Why You Need Python Environments and How to Manage Them with Conda](#)

Hardware - Software

Requisitos CPU:

- Laptop o CPU
- Instalar [miniconda](#)
- Python 3
- Instalar librerías para el fácil uso de ML

Requisitos GPU:

- Nvidia gpu
- Instalar miniconda
- Python 3
- Tensorflow gpu [TF - GPU](#)
- Cuda Toolkit
- CudNN

El gran desarrollo de machine learning ha dado paso a nuevas formas de ayuda en el aprendizaje

Colab - Google

¿Existe un modelo perfecto?



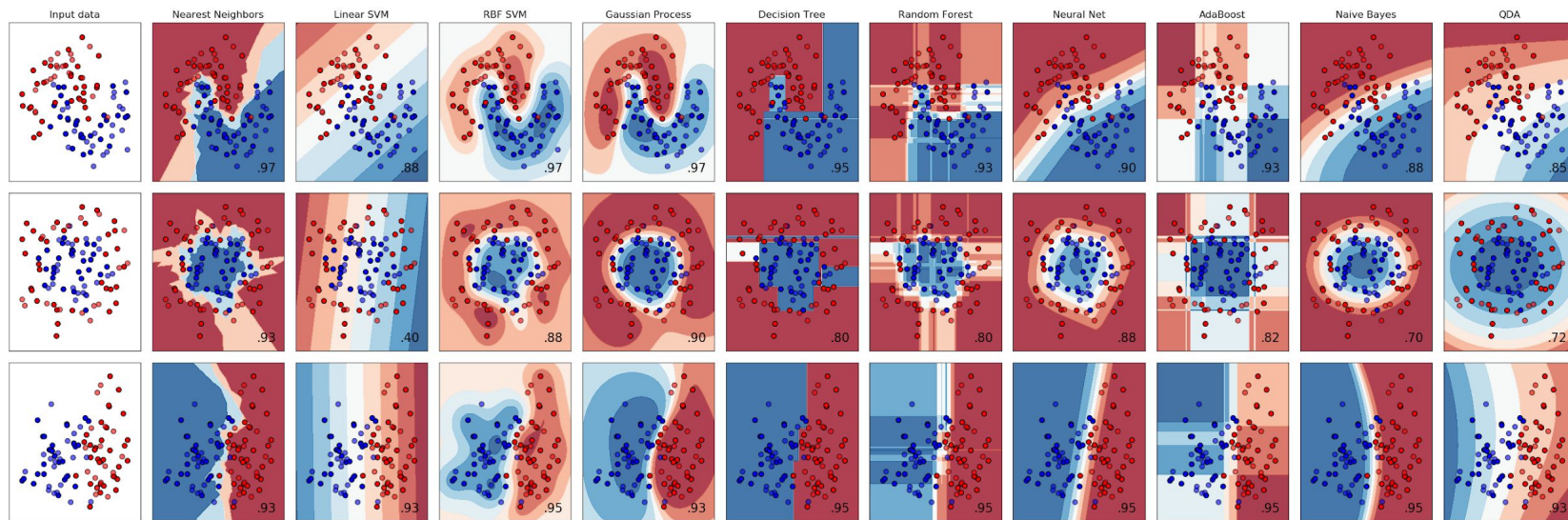
ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

Reto: Activando el ambiente de trabajo, usamos “sklearn Classifier comparison” dentro de los documentos y abrimos la página mediante Colab.

Ejecute el código y analicelo junto al instructor.

Resultado Esperado:



Primero, los datos

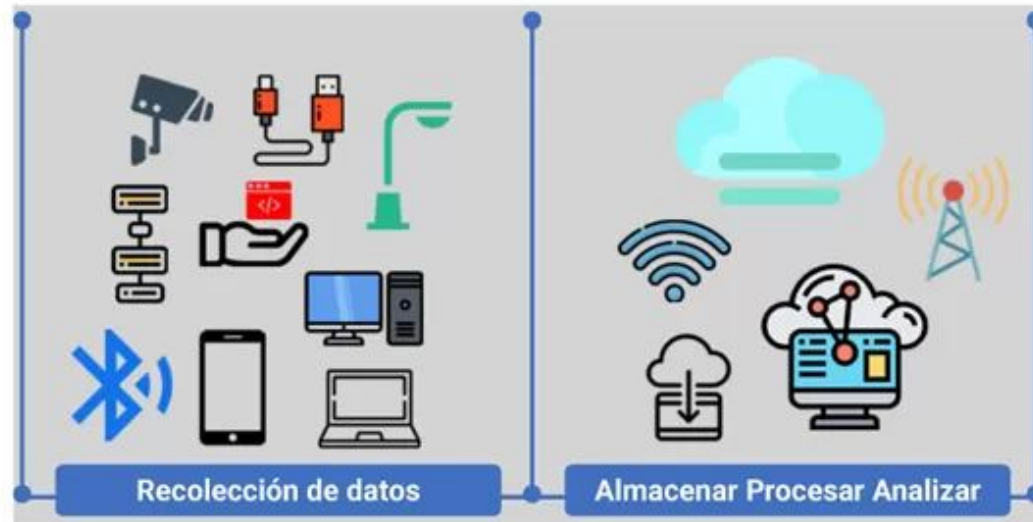
(Segundo, mas datos)



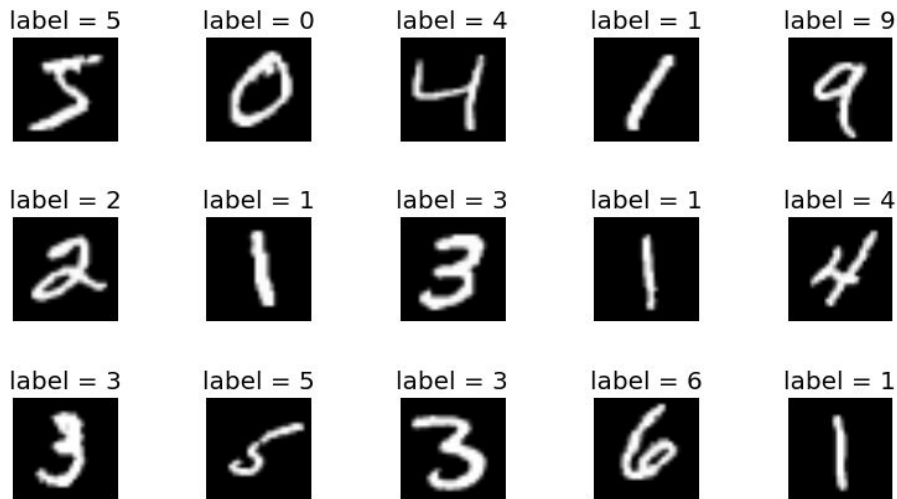
ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

Aprendizaje de Máquina o **Aprendizaje automático (ML)**: Rama de la Inteligencia Artificial que utiliza métodos estadísticos para que las máquinas aprendan de los datos (experiencia).



Datos muy conocidos



MNIST dataset

- 70000 datos de entrenamiento y validación
- 10000 datos de prueba
- Modelo de uso multiple
 - Clasificación
 - Generación
 - Denoising
 -

Datos muy conocidos



CIFAR10 dataset

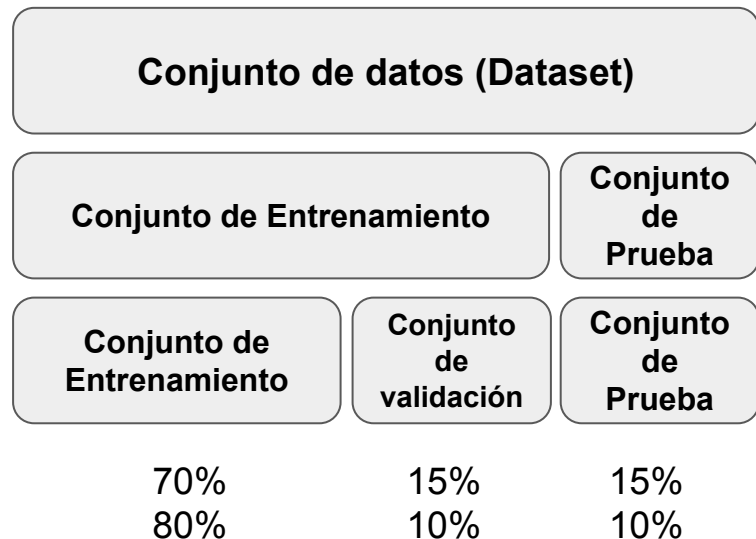
- 60000 datos totales
- Modelo de uso multiple
 - Clasificación
 - Generación
 - Denoising
 -

Conjunto de datos y cómo manejarlos

Representación de los datos

Los conjuntos de datos se separan en tres grupos:

- Conjunto de entrenamiento: Utilizados para determinar los parámetros del estimador (modelo).
- Conjunto de validación: Estima el error del modelo, sirve para ajustar hiper-parámetros
- Conjunto de prueba: Utilizado para medir que tan buena es la generalización del modelo.

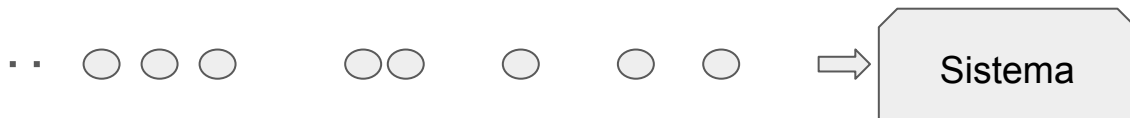


Formas de aprender

Aprendizaje por lotes (Batch learning): El sistema aprende de los datos mediante lotes (grupos de muestras), hasta presentarle todo el conjunto de entrenamiento. El aprendizaje se realiza fuera de línea.



Aprendizaje en línea (Online learning): El sistema va aprendiendo incrementalmente al momento que se le presenta la información (en línea), pero suele hacerse cuando el sistema no está en producción (fuera de servicio), aprendizaje en línea puede ser un nombre confuso.



Diabetes

La diabetes es una enfermedad en la que los niveles de **glucosa (azúcar) de la sangre** están muy altos. La glucosa proviene de los alimentos que consume. La insulina es una hormona que ayuda a que la glucosa entre a las células para suministrarles energía. En la **diabetes tipo 1**, el cuerpo no produce insulina. En la **diabetes tipo 2**, la más común, el cuerpo no produce o no usa la insulina de manera adecuada. Sin suficiente insulina, la glucosa permanece en la sangre.

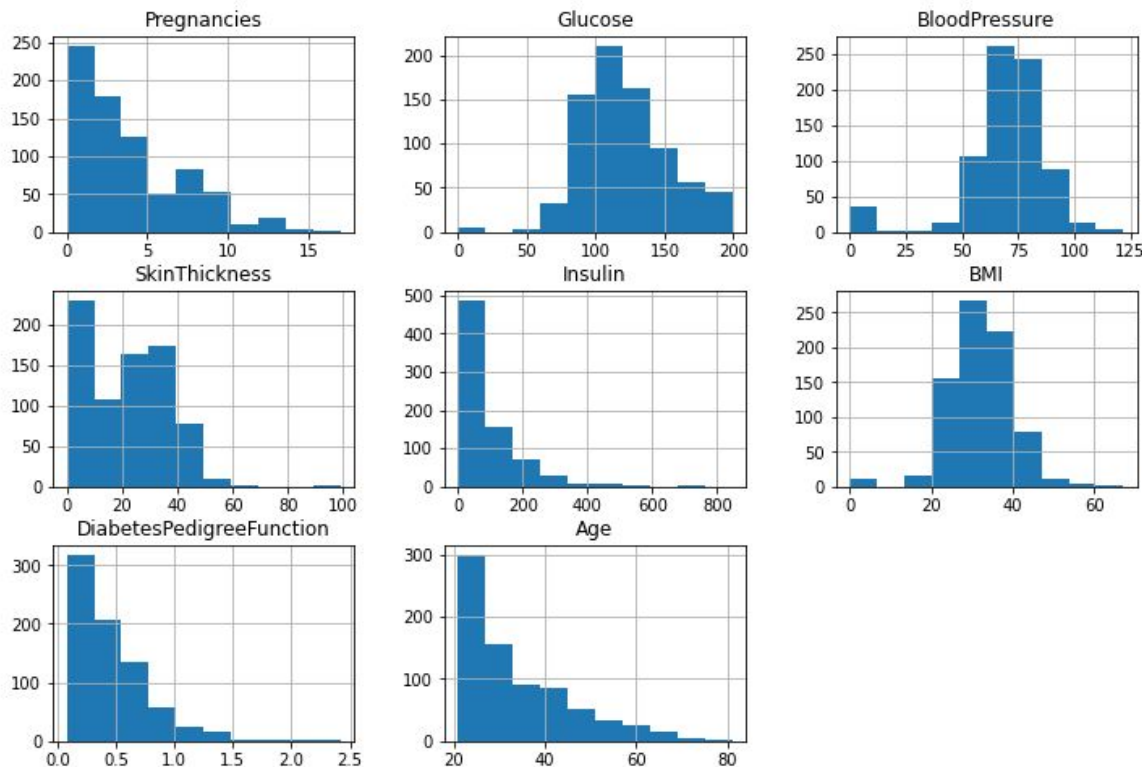


Detección de diabetes

- Como ejemplo utilizaremos un dataset que nos permite detectar si una persona tiene diabetes o no
- El dataset cuenta con 767 muestras con parámetros como:
 - Edad
 - Glucosa
 - Presión arterial
 - etc.
- Es necesario procesar los datos antes de utilizar cualquier modelo

Detección de diabetes

Siguiendo el código del archivo: **Detector de diabetes.ipynb**, observe los datos y prepararlos para su uso



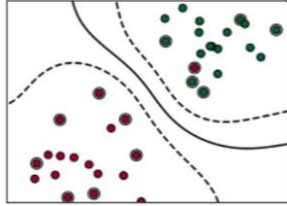
Modelos clasificadores



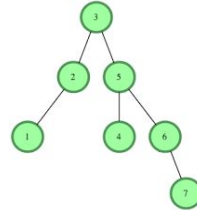
ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

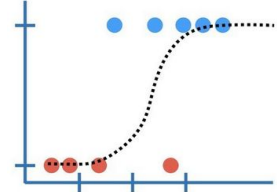
Modelos de machine learning



SVM



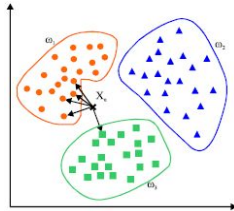
Tree



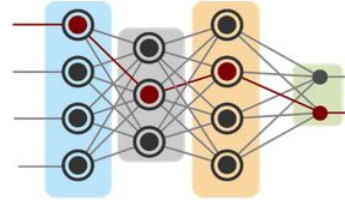
Logistic

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

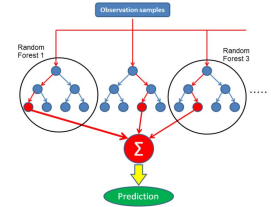
Bayes



K-NN

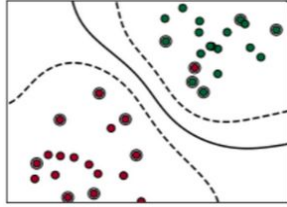


ANN

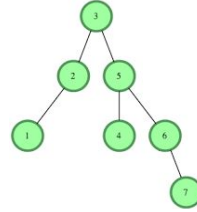


Forest

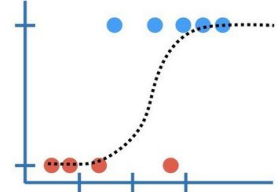
Modelos de machine learning



SVM



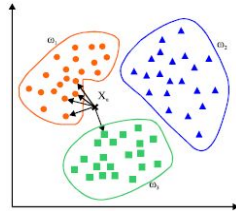
Tree



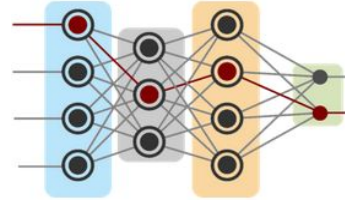
Logistic

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

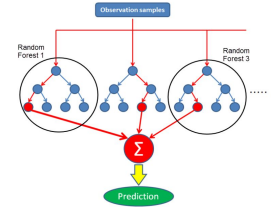
Bayes



K-NN



ANN



Forest

Máquinas de soporte vectorial



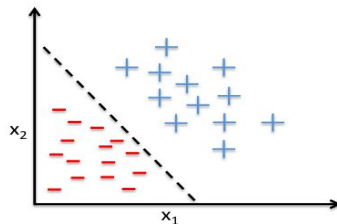
ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

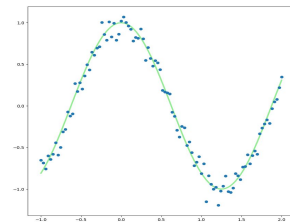
Las máquinas de soporte vectorial (SVM) son un algoritmo de aprendizaje automático bastante popular desde la década de los 90's. En principio, es un modelo lineal que crea un hiperplano de separación

En aprendizaje supervisado:

SVC Support Vector Classifier
(Clasificación)

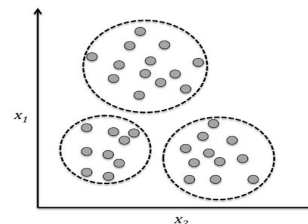


SVR Support Vector Regression
(Regresión)



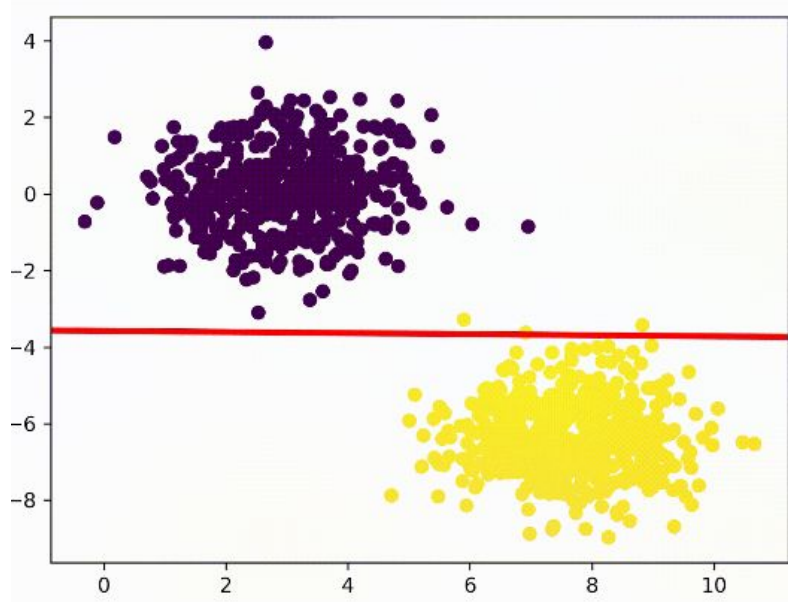
En aprendizaje no supervisado:

Agrupamiento

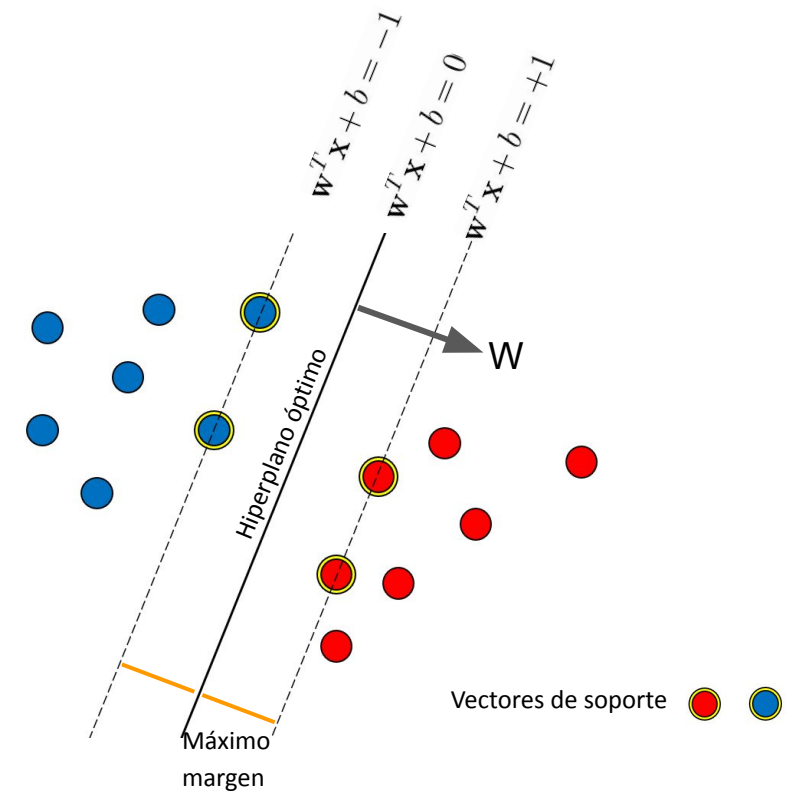


SVM tienen su origen en los trabajos sobre la teoría del aprendizaje estadístico y fueron introducidas en los años 90 por Vapnik y sus colaboradores.

Hiperplanos de separación



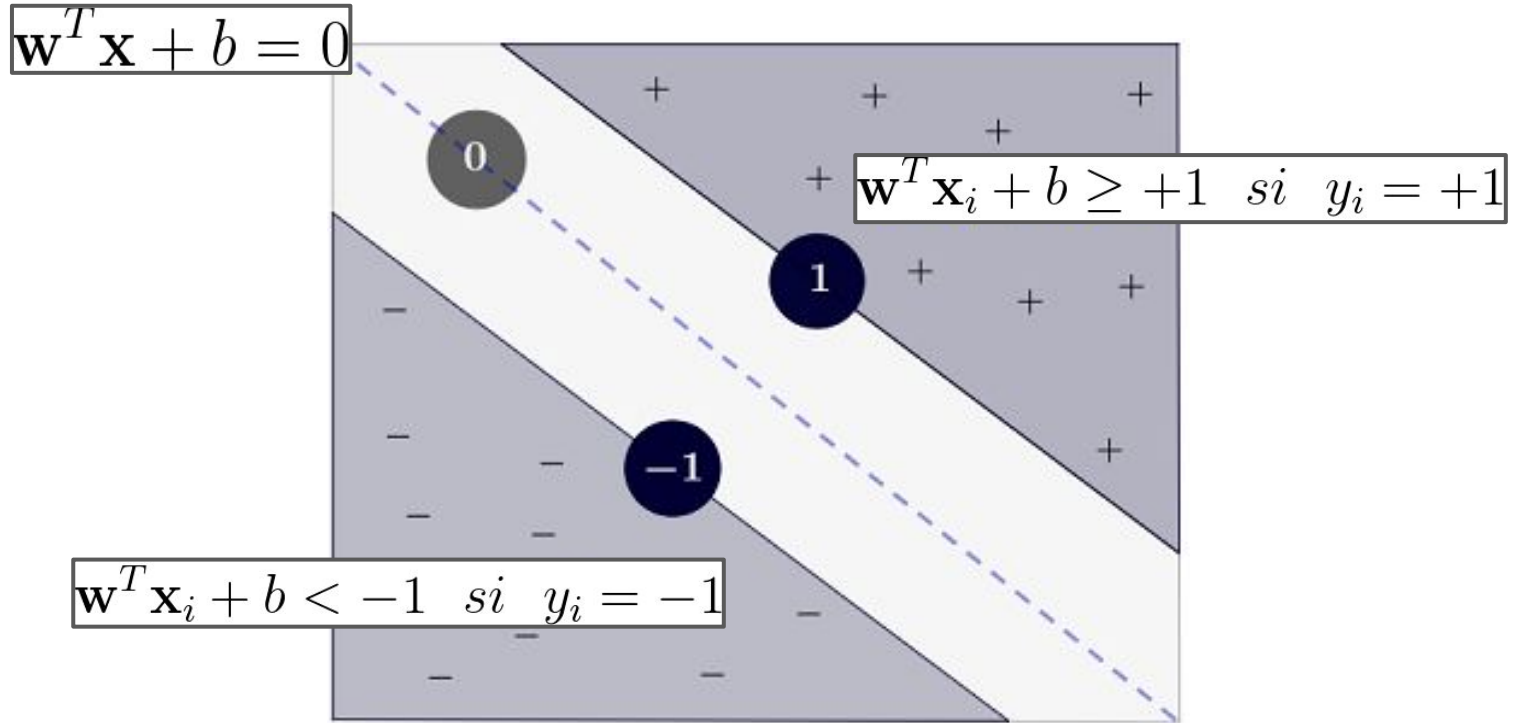
Algoritmo del perceptrón



Support Vector Machine

Aplicaciones de SVM: visión artificial, reconocimiento de caracteres, categorización de texto e hipertexto, clasificación de proteínas, procesamiento de lenguaje natural, análisis de series temporales.

Maximizar el margen bajo la restricción de clasificar correctamente las muestras



Detección de diabetes

Siguiendo con nuestro ejemplo prepararemos una máquina de vector soporte y se entrenará para poder predecir si una persona tiene diabetes

```
from sklearn import svm # Importar la libreria necesaria para usar las SVM
```

```
diabetes_svm = svm.SVC()
```

```
diabetes_svm.fit(train_set,y_train)
```

Árboles de decisión



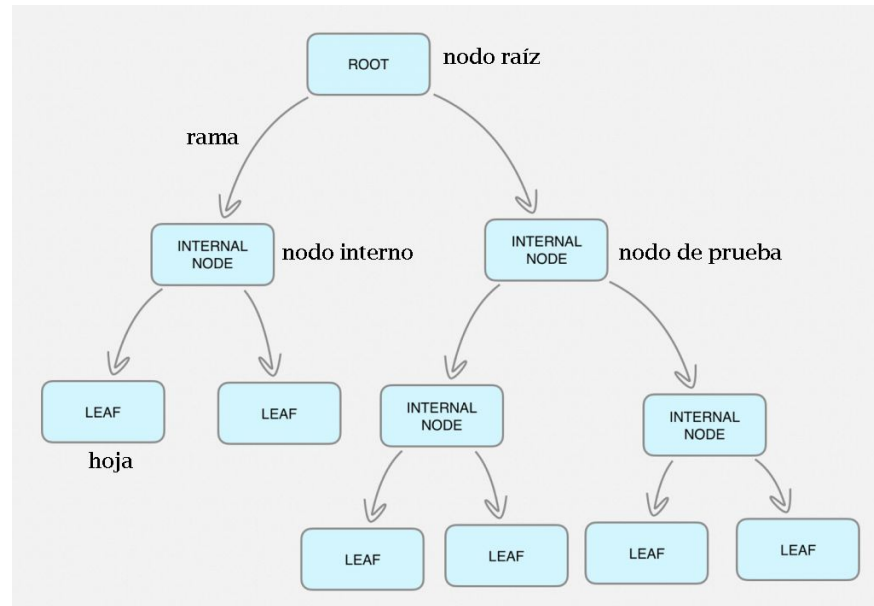
ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

CART (Classification And Regression Trees), Leo Breiman-1984:

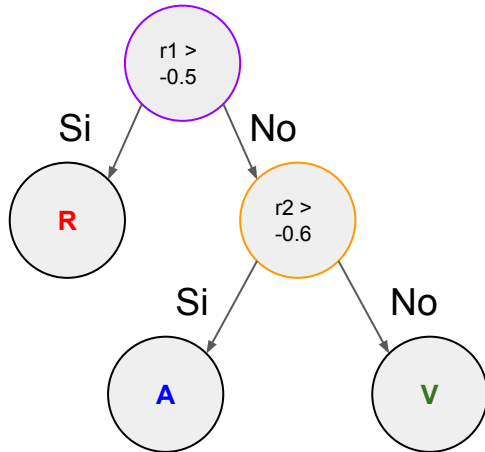
Los árboles de decisión son una técnica simple pero importante en ML

- Pueden utilizarse para regresión y clasificación
- Las variables de entrada y salida pueden ser categóricas o continuas
- Divide iterativamente el conjunto de datos en subconjuntos no superpuestos hasta que los nodos finales (hojas) sean puros
- Por lo general su implementación es un árbol binario



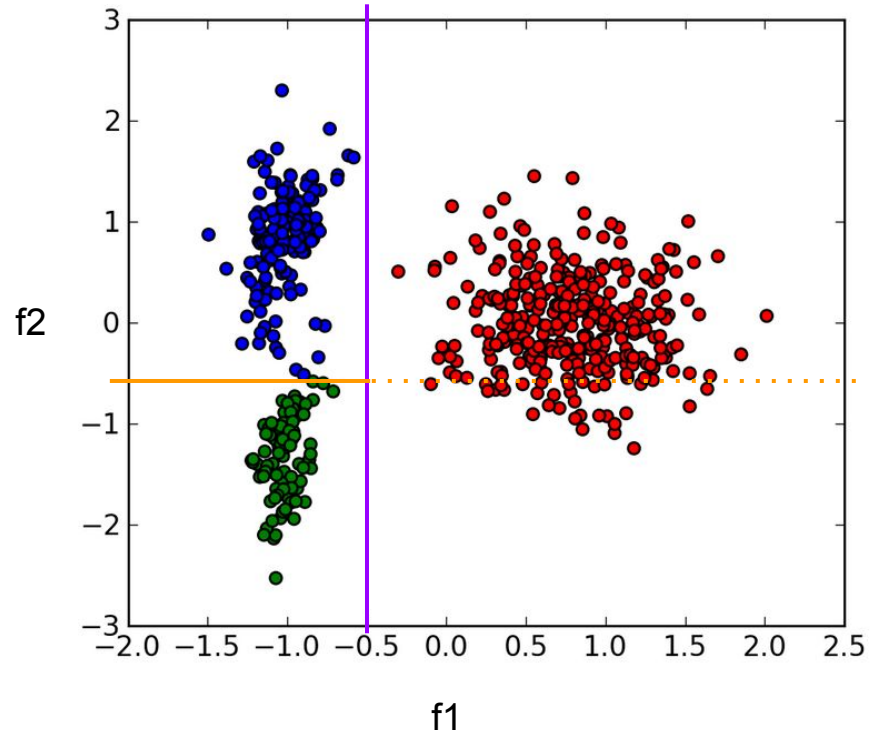
Leo Breiman: CART (1984), Bagging predictors (1996), Random Forests (2001).

Organigrama del buen ingeniero



Reto del algoritmo:

- Dividir correctamente cada clase con los mínimos errores



Detección de diabetes

Siguiendo con nuestro ejemplo prepararemos un árbol de decisión y se entrenará para poder predecir si una persona tiene diabetes

```
from sklearn.tree import DecisionTreeClassifier

diabetes_dt = DecisionTreeClassifier(random_state=0)

diabetes_dt.fit(train_set,y_train)
```

Redes neuronales multicapa

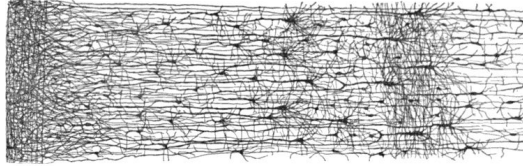
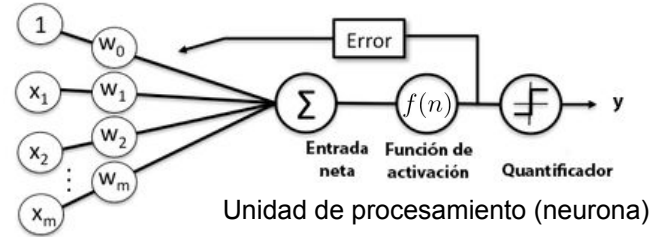
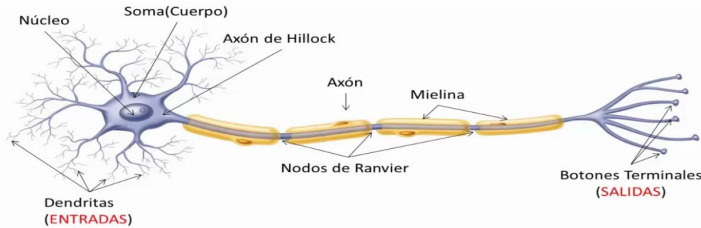


ACTUMLOGOS

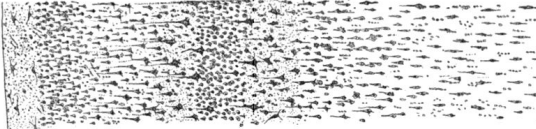
DESARROLLANDO HABILIDADES TECNOLÓGICAS

Neuronas artificiales

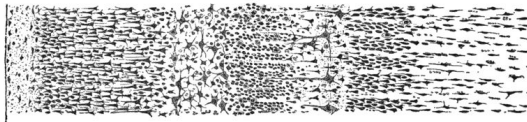
Son modelos simples inspirados en las neuronas biológicas (corteza cerebral).



Corteza motora (niño)

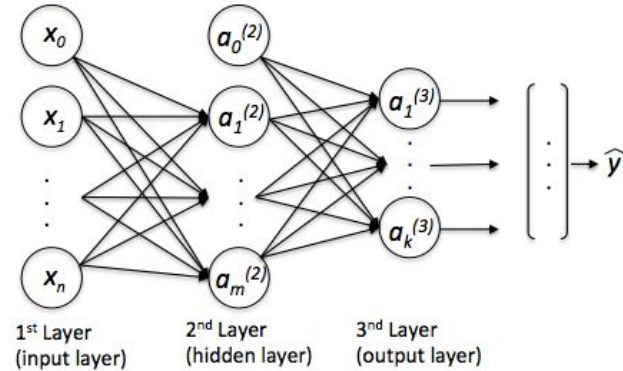


Corteza motora (adulto)



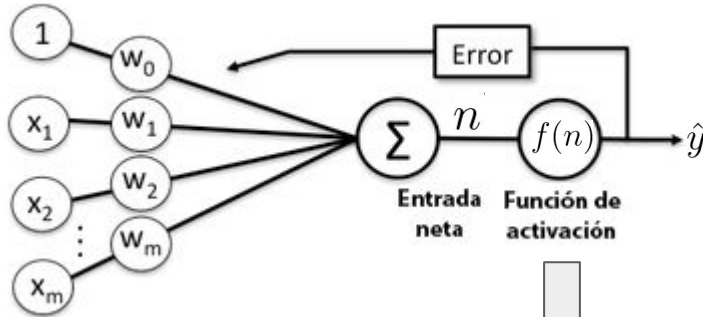
Corteza visual (adulto)

[Fuente](#)



Red neuronal

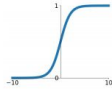
Unidad de procesamiento (neurona)



$x_0 = 1$
 $w_0 = b$
 n = entrada neta
 \hat{y} = predicción

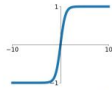
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



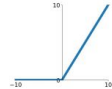
tanh

$$\tanh(x)$$



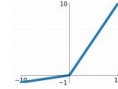
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

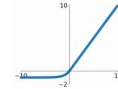


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

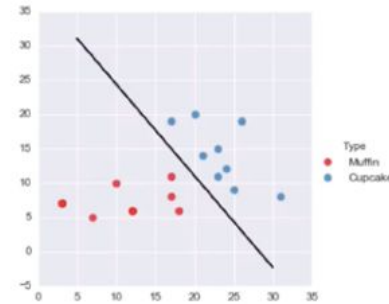


$$\hat{y} = f(n) = f(\mathbf{w}^T \mathbf{x} + b)$$

La frontera de decisión

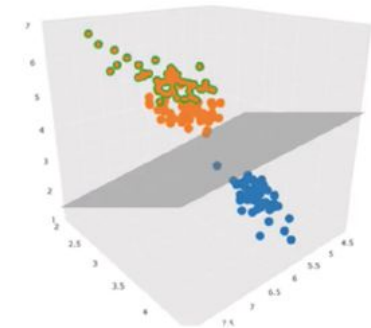
2D: Con 2 rasgos

Línea recta



3D: Con 3 rasgos

Un Plano

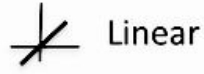


En N dimensiones: Un Hiperplano

¿Qué hace una neurona con el espacio de entrada?

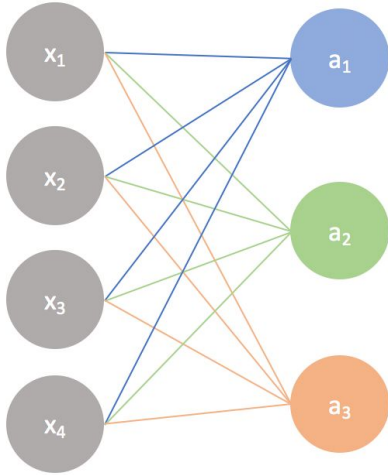
separa el espacio de dimensiones en 2 partes

Red monocapa



Input layer

Output layer



$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \end{bmatrix} = \begin{bmatrix} w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \end{bmatrix}$$

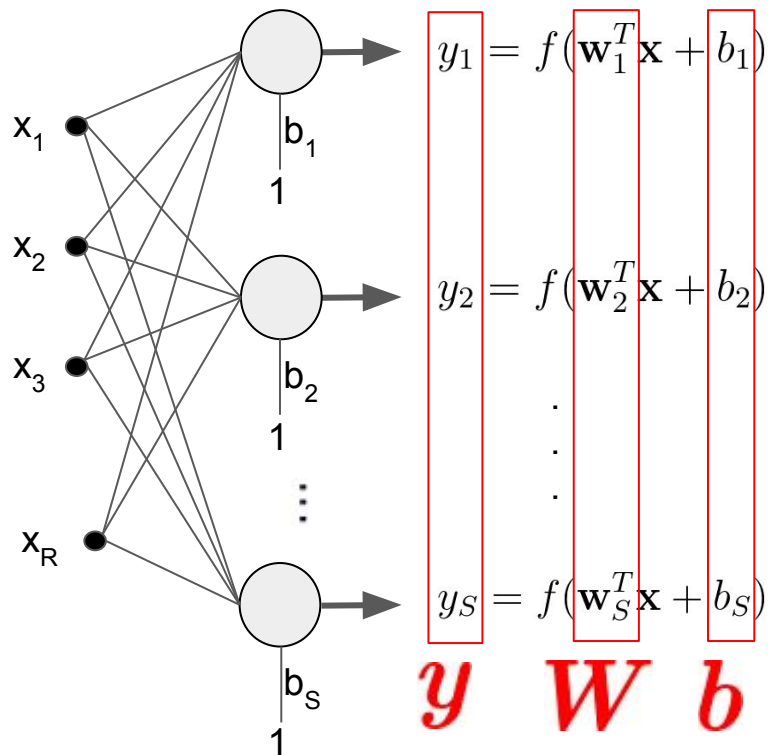
$$w_1 = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix}, w_2 = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix}, w_3 = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix}$$

$$\begin{bmatrix} \leftarrow w_1^T \rightarrow \\ \leftarrow w_2^T \rightarrow \\ \leftarrow w_3^T \rightarrow \end{bmatrix} \begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix}$$

Por simplicidad no se dibujaron las polarizaciones

$$y_i = f(\mathbf{w}_i^T \mathbf{x} + b_i) \quad \forall_i \in [1, S]$$

Red monocapa



$$y_1 = f(\mathbf{w}_1^T \mathbf{x} + b_1)$$

$$y_2 = f(\mathbf{w}_2^T \mathbf{x} + b_2)$$

\vdots

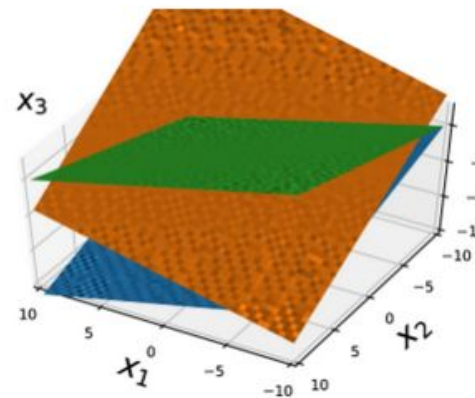
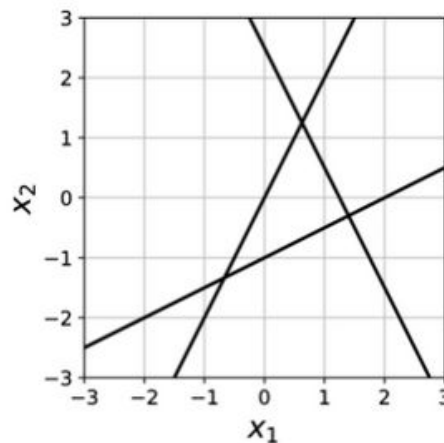
$$y_S = f(\mathbf{w}_S^T \mathbf{x} + b_S)$$

y W b

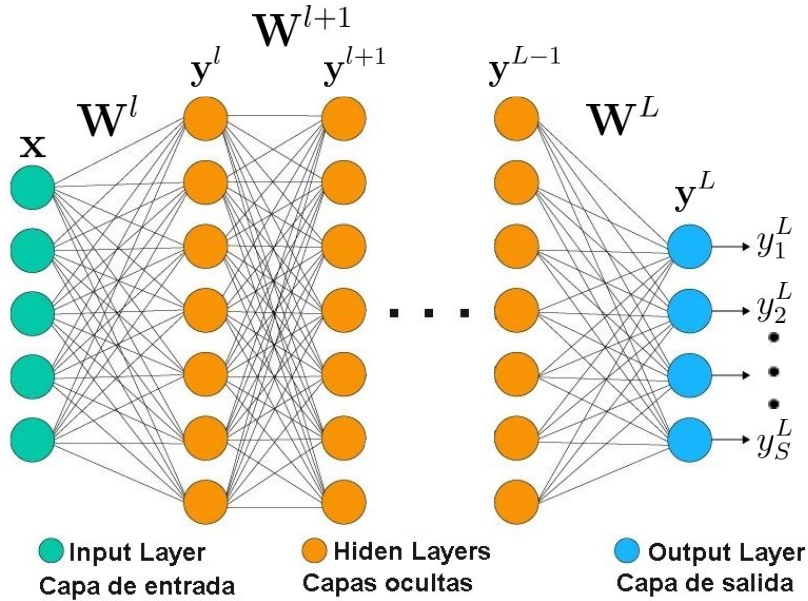
$$\mathbf{y} = f(\mathbf{W} \mathbf{x} + \mathbf{b})$$

¿Cuántos hiperplanos hay?

S hiperplanos



Red multicapa (multi-layer neural network)



$$\mathbf{y}^l = f(\mathbf{W}^l \mathbf{y}^{l-1} + \mathbf{b}^l) \quad \forall l \in [1, L]$$

$$\boxed{\mathbf{y}^1 = f(\mathbf{W}^1 \mathbf{x} + \mathbf{b}^1)} \Rightarrow \boxed{\mathbf{y}^2 = f(\mathbf{W}^2 \mathbf{y}^1 + \mathbf{b}^2)} \Rightarrow \dots \Rightarrow \boxed{\mathbf{y}^L = f(\mathbf{W}^L \mathbf{y}^{L-1} + \mathbf{b}^L)}$$

Nota: la notación \mathbf{y}^l no es un exponencial, se refiere al número de capa de la red, se puede encontrar l indicado como subíndice o superíndice

Detección de diabetes

Siguiendo con nuestro ejemplo prepararemos un perceptrón multicapa y se entrenará para poder predecir si una persona tiene diabetes

```
from sklearn.neural_network import MLPClassifier
```

```
diabetes_mlp = MLPClassifier(random_state=1, max_iter=300)  
diabetes_mlp.fit(train_set, y_train)
```

```
MLPClassifier(max_iter=300, random_state=1)
```

Comprobar el modelo



ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

¿Los resultados mostrados son reales?



¿Es un 2?

Evaluar el desempeño de los modelos



ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

Proceso para la selección de modelos

- Estimar el desempeño del modelo en **datos no vistos** por el modelo
- Aumentar el desempeño ajustando el algoritmo (**hiper-parámetros**)
- Comparar diferentes algoritmos e identificar el modelo que mejor resuelve el problema

Detección de diabetes

Utilizando el set de prueba analicemos nuevamente los tres modelos entrenados y verifiquemos su verdadera fiabilidad

```
# SVM

resultado_svm = diabetes_svm.predict(test_set)
result = 0
error = 0
for i in range(len(resultado_svm)):
    if resultado_svm[i] == y_test[i]:
        result += 1
    else:
        error += 1
print('De las ', len(resultado_svm), ' muestras, ', result, ' son correctas y ', error, ' son erroneas')
print('Porcentaje de acierto: ', result/len(resultado_svm))
```

```
# Arbol de decision

resultado_dt = diabetes_dt.predict(test_set)
result = 0
error = 0
for i in range(len(resultado_dt)):
    if resultado_dt[i] == y_test[i]:
        result += 1
    else:
        error += 1
print('De las ', len(resultado_dt), ' muestras, ', result, ' son correctas y ', error, ' son erroneas')
print('Porcentaje de acierto: ', result/len(resultado_dt))
```

Resumen

¿Machine Learning es útil?, ¿Cuáles son sus características?:

- En aprendizaje automático existen tres principales ramas: supervisado, no supervisado y por refuerzo
- Los principales tipos de problemas son: regresión, clasificación y agrupamiento
- En aprendizaje automático es importante tratar los datos: eliminar ruido, corregir errores, normalizar, quitar rasgos irrelevantes y más.
- Se busca que los modelos tengan un balance entre sesgo y varianza
- Existen métodos para validar el aprendizaje de los modelos: holdout, k-fold, leave one out, train-validation-test
- Y mucho mas ...

Conclusión Final

- Aplicar técnicas de Aprendizaje Automático para profundizar en grandes cantidades de datos puede ayudar a descubrir patrones que no son inmediatamente visibles. Esto se llama minería de datos.
- Los tipos de datos con los que se trabaja y los datos que se recopilan, dependen del problema que esté tratando de resolver:
 - Sonido (reconocimiento de voz)
 - Texto (Clasificación de comentarios)
 - Imágenes (visión artificial)
 - Series temporales (datos del sensor, actividad web)
 - Video (detección de movimiento)

Machine Learning Practico

Curso completo

- 42 Horas de curso
- Martes y viernes de 19 a 22 horas
- Inicio 18 de Marzo
 - Fin 13 de Mayo
- Dias sin clase
 - 15 de abril
 - 10 de mayo



ACTUMLOGOS

DESARROLLANDO HABILIDADES TECNOLÓGICAS

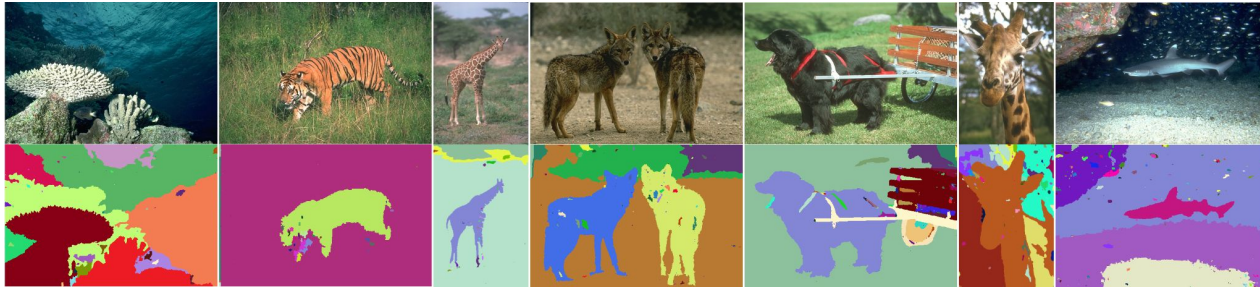


Machine Learning Practico

Curso completo

Casos de estudio:

- Predicción de precios de casas
- Detección de cáncer
- Detector de noticias falsas
- Análisis de sentimientos - Detectar si un texto es positivo o negativo
- Clasificación de tipo de ropa
- Agrupamiento no supervisado de números
- Segmentación no supervisada de imágenes

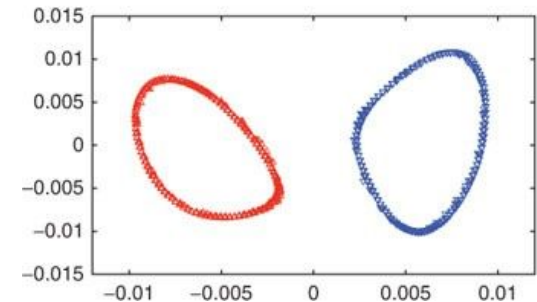
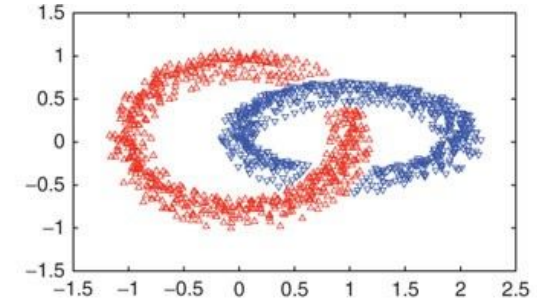


Machine Learning Practico

Curso completo

Temario:

- ☐ Instalación de un ambiente de trabajo para machine learning
- ☐ Panorama general y conceptos básicos
- ☐ Proceso del aprendizaje automático
- ☐ Modelos y métodos
 - ☐ Curvas de aprendizaje
 - ☐ Regularización
 - ☐ Maquinas de vector soporte
 - ☐ Árboles de decisión
 - ☐ Regresión logística
 - ☐ Redes neuronales
 - ☐ Naive Bayes
 - ☐ Bagging
 - ☐ Boosting
 - ☐ Modelos ensamblados
- ☐ Aprendizaje no supervisado
 - ☐ Agrupamiento
 - ☐ Reducción de dimensionalidad



Glosario

Aprendizaje automático (Machine Learning): es una área del conocimiento que busca crear modelos matemáticos y algoritmos efectivos que aprendan a realizar tareas a partir la experiencia representada por datos numéricos. Dichas tareas no deben ser programadas directamente.

Aprendizaje supervisado (Supervised Learning): el modelo matemático debe aprender a asociar las entradas con las salidas. Los ejemplos (entradas, salidas) son dados por un ser humano o algún medio externo, por eso se llama supervisado.

Aprendizaje NO supervisado (Unsupervised Learning): el modelo matemático debe aprender a agrupar los datos de entrada o a representarlos basado en similitudes y diferencias entre ellos. Es deseable que los datos sean clasificados o representados en grupos con el menor traslape y en el menor número de grupos posible. No se conocen las clases ni los grupos al inicio.

Aprendizaje por refuerzo (Reinforcement Learning): el modelo matemático debe aprender a elegir las acciones más convenientes de un agente de acuerdo al estado del ambiente y el estado del agente, con el fin de maximizar una función de recompensa. Por ejemplo, aprender a jugar fútbol, donde quieres maximizar el número de goles anotados y minimizar los recibidos, el ambiente es el campo de fútbol y los agentes son los jugadores.

Regresión: Tarea de aproximar una función para mapear las variables de entrada X a salidas continuas Y , que son valores reales asociado a cantidades (La función predice un valor numérico real)

Glosario

Clasificación: Tarea de aproximar una función para mapear las variables de entrada X a salidas discretas categóricas Y conocidas como etiquetas o clases. (La función predice clases)

Agrupamiento: Tarea de encontrar similitudes en los datos y particionarlos (agruparlos) en clusters cuando se tienen datos sin etiquetar (no se sabe la respuesta correcta).

Precisión (precision): La dispersión del conjunto de valores obtenidos de mediciones repetidas de una magnitud. Cuanto menor es la dispersión mayor la precisión

Exactitud (accuracy): Grado de concordancia entre los resultados de una medición y el valor verdadero

Estimador: Un estimador es cualquier objeto que aprende de los datos; Puede ser un algoritmo de clasificación, regresión o agrupamiento o un transformador que extrae / filtra características útiles de los datos sin procesar

Conjunto de entrenamiento (training set): Los ejemplos que utiliza el sistema para aprender se denominan conjunto de entrenamiento. Cada ejemplo de entrenamiento se llama una instancia de entrenamiento (o muestra)

Rasgos (features): Un rasgo es una propiedad medible individual o **característica** de un fenómeno que se está observando.

Glosario

Sobreajuste (overfitting): Significa que el modelo funciona bien en los datos de entrenamiento, pero no generaliza bien

Subajuste (underfitting): Es lo opuesto al sobreajuste, cuando el modelo es demasiado simple para aprender la estructura de los datos

Hipótesis (hypothesis): Es una determinada función que esperamos sea similar a la función verdadera (la función objetivo) que queremos modelar

Hiperparámetro (hyperparameter): Es un parámetro del algoritmo de aprendizaje y no del modelo, el ajuste de hiperparámetros es realizado la persona

Generalización (generalization): Es la capacidad de un modelo matemático para desempeñarse igual con ejemplos diferentes a los ejemplos de entrenamiento (o validación). Comúnmente, los ejemplos diferentes deben tener la misma naturaleza de origen que los ejemplos de entrenamiento (o validación).