

Aprendizaje Automático

Dra. Cecilia Reyes Peña

Universidad Nacional Autónoma de México

Contenido

- 1 Introducción
- 2 Aprendizaje Supervisado
 - Regresión
 - Regresión lineal
 - Clasificación
- 3 k-Vecinos
- 4 Bayes
- 5 Árboles de decisión
- 6 SVM
- 7 Aprendizaje no supervisado
- 8 Reducción de dimensiones
- 9 PCA
- 10 t-SNE
- 11 Agrupamiento
- 12 K-Means
 - Selección del número óptimo de clústers
- 13 Redes



Introducción



Aprendizaje automático

El aprendizaje automático se encarga de extraer conocimiento de un conjunto de datos (dataset) a través de un proceso de entrenamiento. Las áreas de conocimiento necesarias para entender los algoritmos de aprendizaje automático son:

- Inteligencia artificial
- Estadística
- Computación
- Ciencia de datos



Aprendizaje automático

Para el aprendizaje automático es necesario tener una representación de los datos que pueda ser entendida por la computadora.

Nombre	Edad	Estatura	Peso	CaracterísticaN
"Juan"	"Gómez"	1.75	75	118
"Martha"	"López"	1.65	56	112
"Paris"	"Pérez"	1.23	35	122
....
....

Figura: Representación de datos

Aprendizaje Supervisado



Aprendizaje supervisado

Se conocen pares de datos que representan una entrada y su respectiva salida (conjunto de entrenamiento).

Al ingresar un dato nuevo se busca predecir su respectiva salida.



Ajuste del modelo

El propósito del aprendizaje supervisado es construir modelos a partir de un conjunto de datos de entrenamiento que sean capaces de realizar predicciones ante la entrada de datos nuevos.

Estos datos no vistos deben compartir las mismas características de los datos de entrenamiento.



Ajuste del modelo

La única medida de si un algoritmo funcionará bien con datos nuevos es la evaluación en el conjunto de pruebas.

Si el modelo logra predecir la salida, entonces está generalizado.



Sobreajuste

El sobreajuste se produce cuando se ajusta un modelo demasiado a las particularidades del conjunto de entrenamiento y se obtiene un modelo que funciona bien en el conjunto de entrenamiento, pero que no es capaz de generalizar a los nuevos datos.



Infraajuste

Por otra parte, si el modelo es demasiado simple, es posible que no pueda captar todos los aspectos y la variabilidad de los datos, y el modelo no funcionará bien ni siquiera en el conjunto de entrenamiento. La elección de un modelo demasiado simple se denomina infraajuste.



Regresión

EL propósito de la regresión es predecir un número real a partir de una nueva entrada de datos no vistos en el entrenamiento.

Por ejemplo: Predecir el salario de una persona a partir de su formación, edad, experiencia, entre otras.



Regresión lineal

La regresión lineal es una técnica de análisis de datos que predice el valor de datos desconocidos mediante el uso de otro valor de datos relacionado y conocido.

Modela matemáticamente la variable desconocida o dependiente y la variable conocida o independiente como una ecuación lineal.



Regresión lineal

La fórmula de la regresión lineal es la siguiente:

$$\hat{y} = w[0] * x[0] + w[1] * x[1] + \dots + w[p] * x[p] + b$$

Figura: Caption

Donde:

- \hat{y} es la predicción que realiza el modelo.
- $x[0]$ hasta $x[p]$ son características de un punto de dato.
- w y b son los parámetros que debe aprender el modelo



Regresión lineal simple

Si los datos tienen una sola característica, entonces la ecuación es la siguiente:

$$\hat{y} = w[0] * x[0] + b$$

Figura: Caption

Donde:

- $x[0]$ es la pendiente.
- b es el valor de la intersección con el eje y



Ejemplo de regresión lineal

Se tiene el caso de que si un alumno hizo más ejercicios, entonces obtiene mejor calificación en el examen.

Ejercicios realizados	Calificación examen
2	5
4	7
6	9
4	6
7	10

Table: Ejercicios realizados por alumno



Diagrama de dispersión

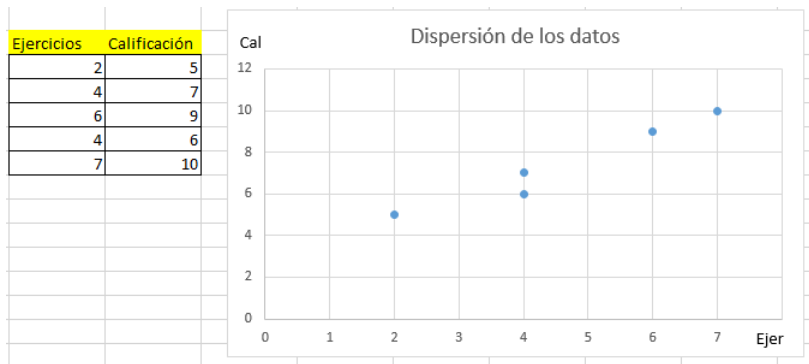


Figura: Caption

Diagrama de dispersión

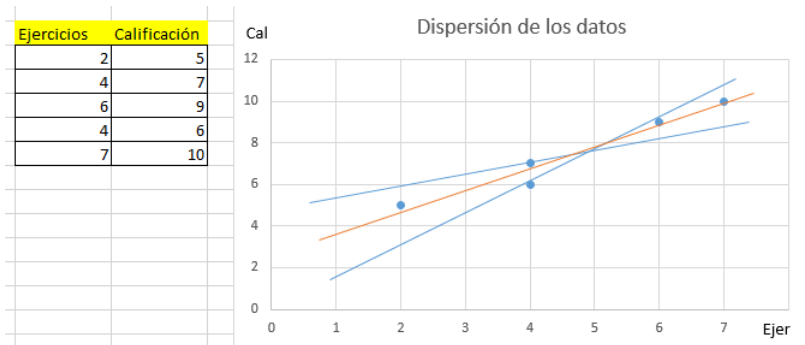


Figura: Caption

Diagrama de dispersión

Ejercicios	Calificación
2	5
4	7
6	9
4	6
7	10

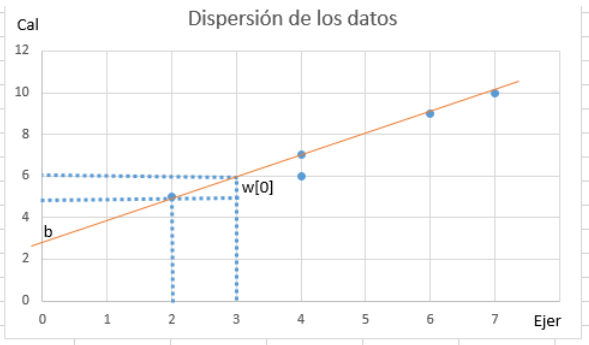


Figura: Caption

Resultados de la regresión

con $b=2.5$ y $w[0]=1.2$	
Ejercicios	Calificación
1	3.7
2	4.9
3	6.1
4	7.3
5	8.5
6	9.7
7	10.9

Figura: Caption

Clasificación

EL objetivo principal de la clasificación es asignar un etiqueta de clase a un nuevo ejemplo, no visto en el entrenamiento.

El conjunto posible de etiquetas provenientes de las clases siempre es conocido.



Tipos de clasificación

Muchas veces se habla de dos tipos principales de clasificación:

- clasificación binaria: solo se tienen dos clases y se puede interpretar como un si/no pertenece a. Por ejemplo: el correo y la clase spam.
- clasificación multiclase: se consideran más de dos clases para etiquetar nuevos datos. Por ejemplo: Clasificación de un texto acorde a su lenguaje.



k-Vecinos



K-Vecinos más cercanos

El algoritmo de los K-Vecinos más cercanos (K-Nearest Neighbor KNN) consiste en almacenar el conjunto de entrenamiento como una serie de puntos en un plano. Para hacer una predicción de un nuevo punto de datos, el algoritmo encuentra el punto del conjunto de entrenamiento más cercano al nuevo punto.

A continuación, asigna la etiqueta de este punto de entrenamiento al nuevo punto de datos.



KNN

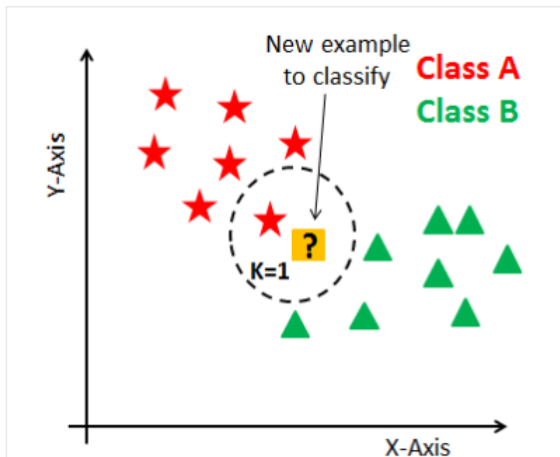
KNN es un algoritmo de aprendizaje perezoso y no paramétrico. No paramétrico significa que no hay suposiciones para la distribución de datos subyacente. En otras palabras, la estructura del modelo determinada a partir del conjunto de datos.

El algoritmo perezoso significa que no necesita ningún punto de datos de entrenamiento para la generación del modelo. Todos los datos de entrenamiento son utilizados en la fase de prueba. Esto hace que el entrenamiento sea más rápido, pero la fase de prueba más lenta y costosa. En el peor de los casos, KNN necesita más tiempo para escanear todos los puntos de datos y escanear todos los puntos de datos requerirá más memoria para almacenar datos de entrenamiento.



KNN

En KNN, K es el número de vecinos más cercanos a considerar. K es generalmente un número impar si el número de clases es 2. Cuando $K = 1$, el algoritmo se conoce como el algoritmo del vecino más cercano.



KNN

Supongamos que P_1 es el punto para el que la etiqueta debe predecir. Primero, encuentra el k punto más cercano a P_1 y luego clasifica los puntos por mayoría de votos de sus k vecinos. Cada objeto vota por su clase y la clase con más votos se toma como predicción. Para encontrar puntos similares más cercanos, encuentre la distancia entre puntos utilizando medidas de distancia como la distancia euclidiana, la distancia de Hamming, la distancia de Manhattan y la distancia de Minkowski.



KNN

KNN tiene los siguientes pasos básicos:

- 1 Calcular distancia
- 2 Encuentra vecinos más cercanos
- 3 Vota por las etiquetas



KNN

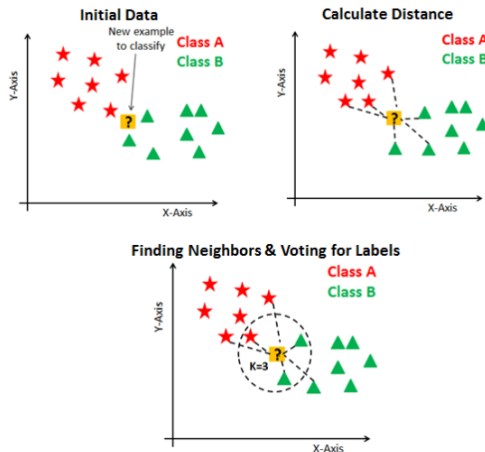


Figura: Caption

Frame Title

KNN funciona mejor con una cantidad menor de funciones que con una gran cantidad de funciones. Puede decir que cuando aumenta la cantidad de funciones, se requieren más datos. El aumento de la dimensión también conduce al problema del sobreajuste. Para evitar el sobreajuste, los datos necesarios deberán crecer exponencialmente a medida que aumente el número de dimensiones. Este problema de dimensión superior se conoce como la Maldición de la Dimensionalidad.

Para lidiar con el problema de la maldición de la dimensionalidad, debe realizar un análisis de componentes principales antes de aplicar cualquier algoritmo de aprendizaje automático, o también puede usar el enfoque de selección de características. La investigación ha demostrado que en grandes dimensiones la distancia euclidiana ya no es útil. Por lo tanto, puede preferir otras medidas, como la similitud del coseno, que se ven claramente menos afectadas por la dimensión alta.



K vecinos

El número de vecinos (K) en KNN es un hiperparámetro que debe elegir en el momento de la construcción del modelo. Puede pensar en K como una variable de control para el modelo de predicción.

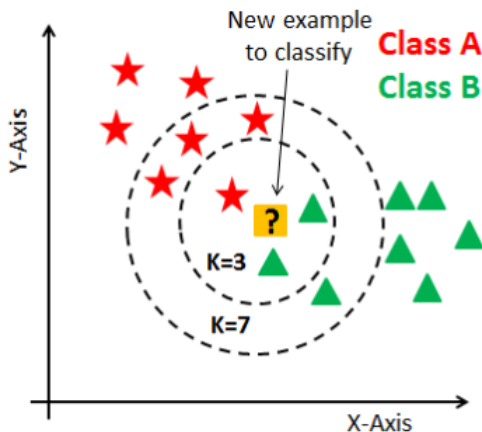
Ningún número óptimo de vecinos se adapta a todo tipo de conjuntos de datos. Cada conjunto de datos tiene sus propios requisitos.

En el caso de un pequeño número de vecinos, el ruido tendrá una mayor influencia en el resultado, y un gran número de vecinos lo encarecen computacionalmente.



K vecinos

Generalmente, k es un número impar si el número de clases es par. También puede verificar generando el modelo en diferentes valores de k y verificar su rendimiento.



Bayes



Teorema de Bayes

Teorema de Bayes (Thomas Bayes, 1763), el cual especifica que: Sea $A=A_1, A_2, \dots, A_n$ un conjunto de sucesos mutuamente excluyentes y exhaustivos, donde la probabilidad de cada uno de ellos es distinta de cero. Sea B un suceso cualquiera del que se conocen las probabilidades condicionales $P(B|A_i)$ y A_i un suceso perteneciente a A .

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

Figura: Caption



Teorema de Bayes

Donde $P(A_i)$ es la probabilidad a priori del suceso A_i , $P(B|A_i)$ es la probabilidad verosímil del suceso B supuesto que hubiera ocurrido en el suceso A_i y $P(A_i|B)$ es la probabilidad a posteriori del suceso A_i .

$$P(A_i|B) = \frac{P(A_i) \cdot P(B|A_i)}{\sum_{i=1}^n P(A_i) \cdot P(B|A_i)}$$

Figura: Caption



Naïve Bayes

Está basado en la simplificación del Teorema de Bayes y asume que cada una de las características de una clase, contribuyen de forma independiente a la probabilidad de que un objeto pertenezca a dicha clase. Su entorno de aprendizaje es supervisado y no necesita una gran cantidad de elementos para el entrenamiento.



Naïve Bayes

Problema: Estando en la oficina alguien paso vistiendo de rojo y no se sabe quien es En oficina se trabajan 4 días a la semana En esa oficina Alicia asiste 3 días a la semana y Bruno asiste solo 1 día a la semana Alicia viste de rojo en una probabilidad 0.4 Bruno viste de rojo en una probabilidad de 0.6



Naïve Bayes

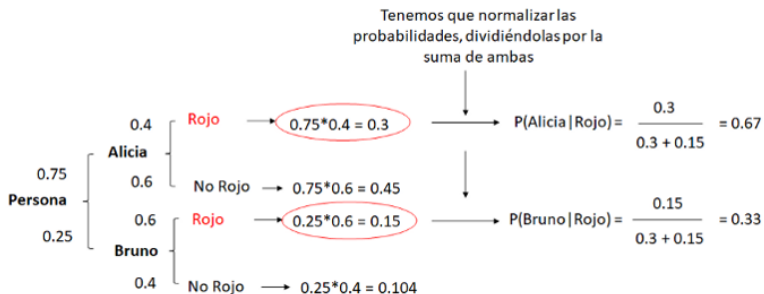
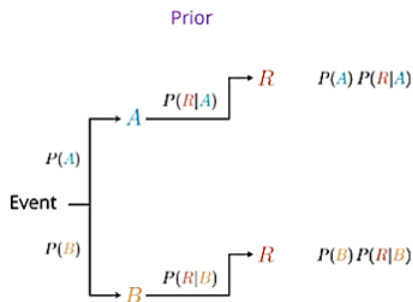


Figura: Caption

Naïve Bayes



Posterior

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

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

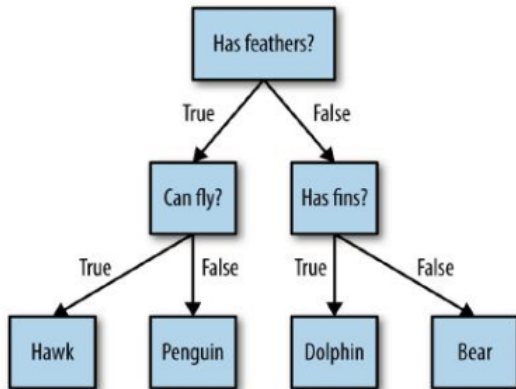
Figura: Caption

Árboles de decisión



Árboles de decisión

Los árboles de decisión son estructuras que aprenden de una jerarquía compuesta por condicionales *IF/ELSE* para tomar decisiones. Son muy útiles cuando la dispersión de los datos es compleja.



Árboles de decisión

- Los árboles comienzan con un nodo raíz.
- El conjunto de preguntas (test) son conocidos como nodos internos.
- Los nodos de los niveles más bajos de cada ramificación son conocidos como hojas.



Árboles de decisión

Los árboles buscar realizar particiones binarias del conjunto de datos para segmentarlos. Un conjunto de datos que ya no puede ser segmentado, tendrá el rol de una hoja. Una hoja que solo tiene valores pertenecientes a una misma clase se le considera como una hoja pura. La profundidad de un árbol está definida por la trayectoria más larga desde el nodo raíz hasta una hoja.



Árboles de decisión

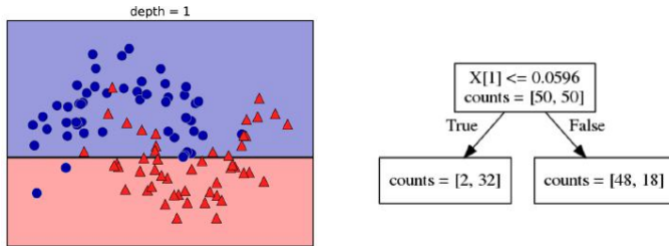


Figura: Caption

Árboles de decisión

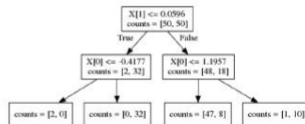
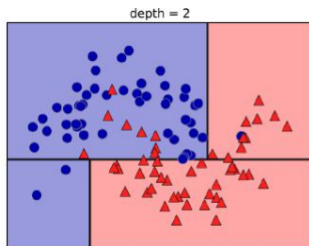


Figura: Caption

Árboles de decisión

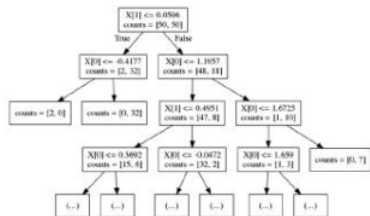
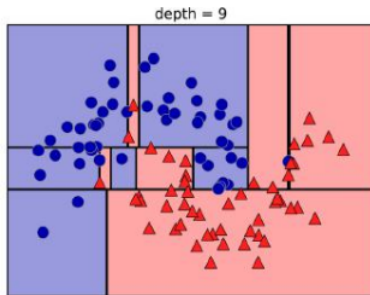


Figura: Caption

Construcción de árboles de decisión

La idea básica detrás de cualquier algoritmo de árbol de decisión es la siguiente:

- 1 Seleccione el mejor atributo utilizando Medidas de selección de atributos (ASM) para dividir los registros. Las medidas más populares son la ganancia de información, la relación de ganancia y el índice de Gini.
- 2 Convierta ese atributo en un nodo de decisión y divida el conjunto de datos en subconjuntos más pequeños.
- 3 Inicie la construcción del árbol repitiendo este proceso recursivamente para cada niño hasta que una de las condiciones coincida:
 - Todas las tuplas pertenecen al mismo valor de atributo.
 - No quedan más atributos.
 - No hay más instancias.



SVM



Support Vector Machine

SVM construye un hiperplano en un espacio multidimensional para separar diferentes clases. SVM genera un hiperplano óptimo de forma iterativa, que se utiliza para minimizar un error. La idea central de SVM es encontrar un hiperplano marginal máximo (MMH) que divida mejor el conjunto de datos en clases.

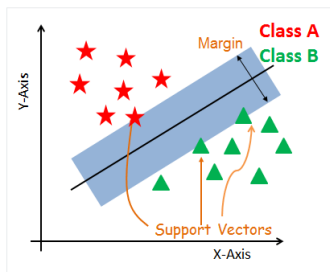


Figura: Caption

Support Vector Machine

- Los vectores de soporte son los puntos de datos que están más cerca del hiperplano. Estos puntos definirán mejor la línea de separación al calcular los márgenes. Estos puntos son más relevantes para la construcción del clasificador.
- Un hiperplano es un plano de decisión que separa un conjunto de objetos que pertenecen a diferentes clases.
- Un margen es un espacio entre las dos líneas en los puntos de clase más cercanos. Esto se calcula como la distancia perpendicular desde la línea hasta los vectores de apoyo o los puntos más cercanos. Si el margen es mayor entre las clases, entonces se considera un buen margen, un margen más pequeño es un mal margen.



SVM

El objetivo principal es segregar el conjunto de datos dado de la mejor manera posible. La distancia entre los puntos más cercanos se conoce como margen. El objetivo es seleccionar un hiperplano con el máximo margen posible entre los vectores de soporte en el conjunto de datos dado. SVM busca el hiperplano marginal máximo en los siguientes pasos:

- Genera hiperplanos que segregan las clases de la mejor manera. Figura del lado izquierdo que muestra tres hiperplanos negro, azul y naranja. Aquí, el azul y el naranja tienen un mayor error de clasificación, pero el negro separa las dos clases correctamente.
- Seleccione el hiperplano correcto con la segregación máxima desde los puntos de datos más cercanos, como se muestra en la figura del lado derecho.



SVM

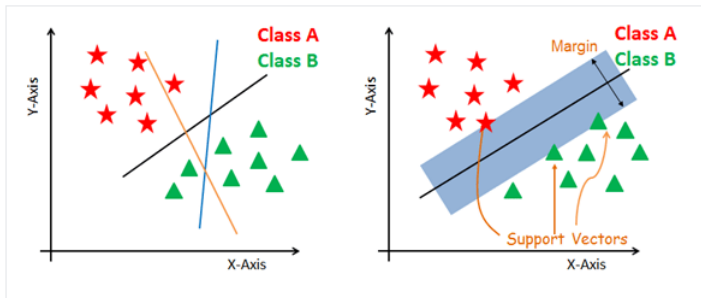
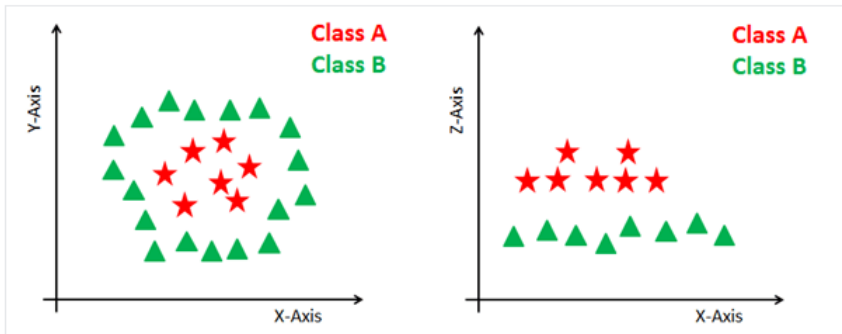


Figura: Caption

Manejo de planos no lineales e inseparables

Algunos problemas no se pueden resolver usando un hiperplano lineal, como se muestra en la siguiente figura (lado izquierdo).

En tal situación, SVM utiliza un truco del núcleo para transformar el espacio de entrada en un espacio dimensional superior, como se muestra a la derecha. Los puntos de datos se trazan en el eje x y el eje z (Z es la suma al cuadrado de x e y).



Núcleos SVM

El algoritmo SVM se implementa en la práctica usando un kernel. Un kernel transforma un espacio de datos de entrada en la forma requerida. SVM utiliza una técnica llamada truco del kernel. Aquí, el núcleo toma un espacio de entrada de baja dimensión y lo transforma en un espacio de mayor dimensión. En otras palabras, puede decir que convierte un problema no separable en un problema separable al agregarle más dimensión. Es más útil en problemas de separación no lineal. El truco del kernel te ayuda a construir un clasificador más preciso.



Tipos de núcleos SVM

- Núcleo lineal: se puede utilizar como producto escalar normal de dos observaciones dadas cualesquiera. El producto entre dos vectores es la suma de la multiplicación de cada par de valores de entrada. $K(x, x_i) = \sum(x * x_i)$
- Núcleo polinomial Un núcleo polinomial es una forma más generalizada del núcleo lineal. El núcleo polinomial puede distinguir el espacio de entrada curvo o no lineal. $K(x, x_i) = 1 + \sum(x * x_i)^d$.

Donde d es el grado del polinomio. $d=1$ es similar a la transformación lineal. El grado debe especificarse manualmente en el algoritmo de aprendizaje.



Tipos de núcleos SVM

- Núcleo de función de base radial: es una función de núcleo popular comúnmente utilizada en la clasificación de máquinas de vectores de soporte. RBF puede mapear un espacio de entrada en un espacio dimensional infinito. $K(x, x_i) = \exp(-\gamma \sum ((x - x_i)^2))$

Aquí, γ es un parámetro, que varía de 0 a 1. Un valor más alto de γ se ajustará perfectamente al conjunto de datos de entrenamiento, lo que provoca un ajuste excesivo. $\gamma=0.1$ se considera un buen valor predeterminado. El valor de γ debe especificarse manualmente en el algoritmo de aprendizaje.



Tipos de núcleos SVM

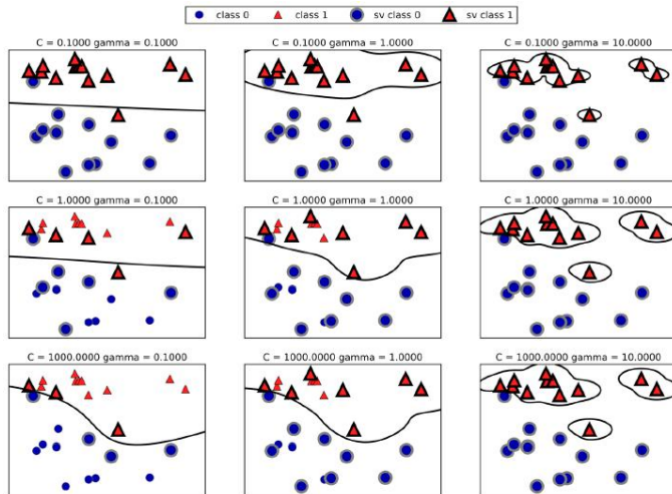


Figura: Caption
Aprendizaje Automático

Aprendizaje no supervisado



Aprendizaje no supervisado

Se conocen los datos pero no las salidas. Por lo que los algoritmos intentarán entender y evaluar dichos datos. En lugar de tener una salida, los datos solo tienen una entrada que serían múltiples variables que describen los datos.



Reducción de dimensiones



Reducción de dimensiones

Una aplicación común de las transformaciones no supervisadas es la reducción de la dimensionalidad, que toma una representación de datos que consiste en muchas características, y encuentra una nueva forma de representar estos datos resumiendo las características esenciales con menos características.

Una aplicación común de la reducción de la dimensionalidad es la reducción a dos dimensiones con fines de visualización.



PCA



Principal Component Analysis

El análisis de componentes principales (PCA) es una técnica de reducción de dimensionalidad lineal que se puede utilizar para extraer información de un espacio de alta dimensión proyectándola en un subespacio de menor dimensión.

Intenta preservar las partes esenciales que tienen más variación de los datos y eliminar las partes no esenciales con menos variación.



Principal Component Analysis

Las dimensiones no son más que características que representan los datos. Una cosa importante a tener en cuenta sobre PCA es que es una técnica de reducción de dimensionalidad no supervisada, puede agrupar los puntos de datos similares en función de la correlación de características entre ellos sin supervisión (o etiquetas).



Principal Component Analysis

PCA es un procedimiento estadístico que utiliza una transformación ortogonal para convertir un conjunto de observaciones de variables posiblemente correlacionadas (entidades cada una de las cuales toma varios valores numéricos) en un conjunto de valores de variables linealmente no correlacionadas llamadas componentes principales.



Aplicación de PCA

- Visualización de datos: cuando se trabaja en cualquier problema relacionado con datos, el desafío en el mundo actual es el gran volumen de datos y las variables/características que definen esos datos. Para resolver un problema, necesita una amplia exploración de datos, como descubrir cómo se correlacionan las variables o comprender la distribución de algunas variables.
- Aceleración del algoritmo de aprendizaje automático (ML): dado que la idea principal de PCA es la reducción de la dimensionalidad, puede aprovecharla para acelerar el entrenamiento y el tiempo de prueba de su algoritmo de aprendizaje automático, teniendo en cuenta que sus datos tienen muchas características y que el aprendizaje del algoritmo ML es demasiado lento. En un nivel abstracto, toma un conjunto de datos que tiene muchas características y simplifica ese conjunto de datos seleccionando algunas Principales Componentes de las características originales.



¿Qué es un componente principal?

Los componentes principales representan lo que hay debajo del capó de sus datos. En términos sencillos, cuando los datos se proyectan en una dimensión más baja (suponga tres dimensiones) desde un espacio más alto, las tres dimensiones no son más que los tres componentes principales que capturan (o contienen) la mayor parte de la variación (información) de sus datos .



¿Qué es un componente principal?

Los componentes principales tienen dirección y magnitud. La dirección representa a través de qué ejes principales se distribuyen principalmente los datos o tienen la mayor variación y la magnitud indica la cantidad de variación que el Componente principal captura de los datos cuando se proyecta en ese eje. Los componentes principales son una línea recta y el primer componente principal tiene la mayor variación en los datos. Cada componente principal posterior es ortogonal al último y tiene una varianza menor. De esta forma, dado un conjunto de x variables correlacionadas sobre y muestras, se obtiene un conjunto de u componentes principales no correlacionados sobre las mismas y muestras.



¿Qué es un componente principal?

La razón por la que logra componentes principales no correlacionados a partir de las características originales es que las características correlacionadas contribuyen al mismo componente principal, lo que reduce las características de datos originales a componentes principales no correlacionados; cada uno representa un conjunto diferente de características correlacionadas con diferentes cantidades de variación. Cada componente principal representa un porcentaje de la variación total capturada de los datos.



PCA

Funcionamiento de PCA:

- Encontrar la dirección de la máxima varianza. AL ser está una dirección se representa con un vector, dicho vector contiene información sobre las características que presentan mayor correlación (CP). En dos dimensiones se debe encontrar un vector ortogonal para comparar dos varianzas y seleccionar la más grande. Para datos de tres o más dimensiones suele haber una infinidad de vectores ortogonales.



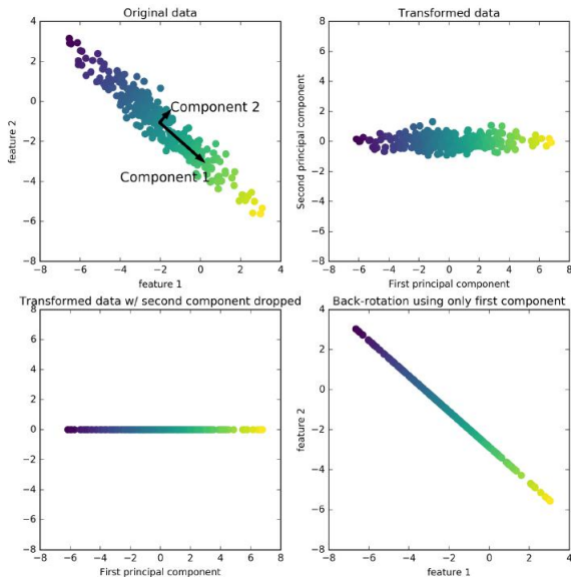
PCA

Funcionamiento de PCA:

- Después se realiza una rotación de los datos para alinear los componentes principales con los ejes originales. Antes de la rotación, se restó la media de los datos, de modo que los datos transformados se centran en torno a cero. En la representación rotada encontrada por el CP, los dos ejes no están correlacionados, lo que significa que la matriz de correlación de los datos en esta representación es cero excepto en la diagonal.
- Eliminación del segundo CP.
- Rotación de los datos hacia la dirección original.



PCA



t-SNE



T-Distribute Stochastic Neighbor Embedding

T-SNE consiste en crear una distribución de probabilidad que represente las similitudes entre vecinos en un espacio de gran dimensión y en un espacio de menor dimensión. Por similitud, intentaremos convertir las distancias en probabilidades.



T-SNE

El procedimiento consta de tres pasos:

- 1 Cálculo de similitudes
- 2 Creación del espacio dimensional
- 3 Representación de los datos



Cálculo de similitudes

Calcular las similitudes de puntos en el espacio inicial de grandes dimensiones. Para cada punto x_i centramos una distribución gaussiana alrededor de este punto. Luego medimos, para cada punto x_j (i diferente de j), la densidad bajo esta distribución gaussiana definida previamente. Finalmente, normalizamos para cada uno de los puntos. De este modo obtenemos una lista de probabilidades condicionales observadas: La desviación estándar se define por un valor llamado perplejidad que corresponde al número de vecinos alrededor de cada punto. Este valor lo establece el usuario de antemano y permite estimar la desviación estándar de las distribuciones gaussianas definidas para cada punto x_i . Cuanto mayor es la perplejidad, mayor es la variación.



Creación del espacio dimensional

Se necesita crear un espacio dimensional más pequeño en el que se representan los datos. Al principio no se conocen las coordenadas ideales en este espacio. Por tanto, los puntos se distribuyen de forma aleatoria en este nuevo espacio. El resto es bastante similar al paso 1, se calculan las similitudes de los puntos en el espacio recién creado, pero usando una distribución t-Student y no una gaussiana. De la misma forma obtenemos una lista de probabilidades :



Representación de los datos

Para representar fielmente los puntos en el espacio dimensional más pequeño, lo ideal sería que las medidas de similitud en los dos espacios coincidieran. Entonces, necesitamos comparar las similitudes de los puntos en los dos espacios usando la medida Kullback-Leibler (KL). Luego tratamos de minimizarlo mediante la gradiente descendente para obtener el mejor y_i posible en un espacio de dimensiones pequeñas. Esto equivale a minimizar la diferencia entre las distribuciones de probabilidad entre el espacio original y el espacio de menor dimensión.



PCA VS T-SNE

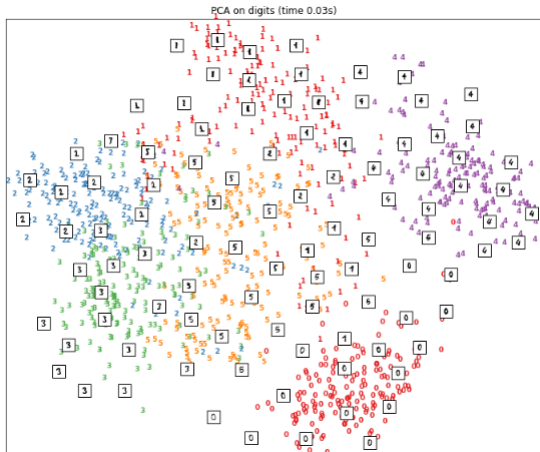
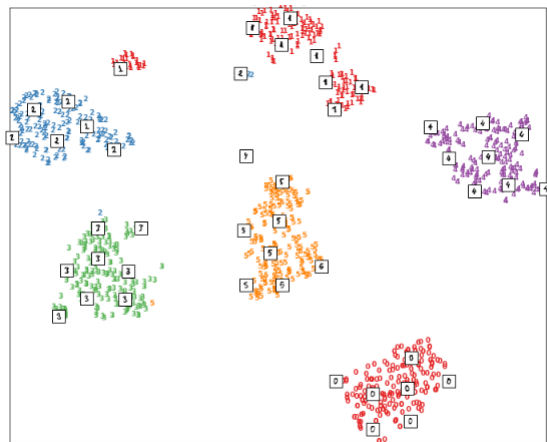


Figura: Caption

PCA VS T-SNE



Reducción de dimensiones con el método t-SNE

Figura: Caption

Agrupamiento



Agrupamiento

El agrupamiento es la tarea de agrupar un conjunto de objetos de manera que los objetos en el mismo grupo sean más similares entre sí que con los objetos en otros grupos. La similitud es una métrica que refleja la fuerza de la relación entre dos objetos de datos.



Tipos de agrupamiento

El agrupamiento se puede dividir ampliamente en dos subgrupos:

- Agrupamiento duro: en el agrupamiento duro, cada objeto o punto de datos pertenece por completo a un clúster o no.
- Agrupamiento suave: en el agrupamiento suave, un punto de datos puede pertenecer a más de un clúster con algún valor de probabilidad o verosimilitud. Por ejemplo, podría identificar algunas ubicaciones como puntos fronterizos pertenecientes a dos o más distritos.



Algoritmos de agrupamiento

Los algoritmos de agrupamiento se pueden categorizar en función de su modelo de clúster, es decir, en función de cómo forman clústeres o grupos.

- Basado en conectividad
- Basado en centroide
- Basado en distribuciones
- Basados en la densidad



Agrupamiento basado en conectividad

La idea principal detrás de este agrupamiento es que los puntos de datos que están más cerca en el espacio de datos están más relacionados (similares) que los puntos de datos más alejados. Los clústeres se forman conectando puntos de datos según su distancia. A diferentes distancias, se formarán diferentes grupos y se pueden representar usando un dendrograma, lo que revela por qué también se les llama comúnmente "agrupamiento jerárquico".

Estos métodos no producen una partición única del conjunto de datos, sino una jerarquía de la que el usuario aún debe elegir los clústeres apropiados eligiendo el nivel en el que desea agrupar. Tampoco son muy resistentes a los valores atípicos, que pueden aparecer como clústeres adicionales o incluso hacer que otros clústeres se fusionen.



Dendogramas

Un dendrograma es un tipo de representación gráfica en forma de árbol que organiza y agrupa los datos en subcategorías según su similitud; dada por alguna medida de distancia. Los objetos similares se representan en el dendrograma por medio de un enlace cuya posición está determinada por el nivel de similitud entre los objetos o grupos de objetos. Dadas estas características, hace que los dendrogramas sean un tipo de diagrama muy útil para estudiar las agrupaciones de los datos.



Dendogramas

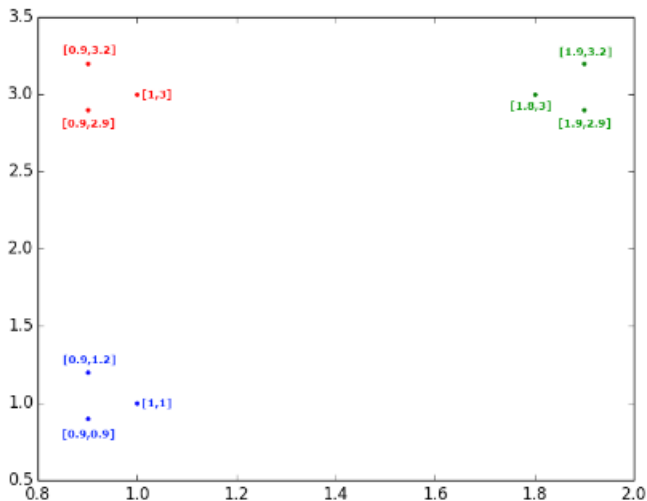
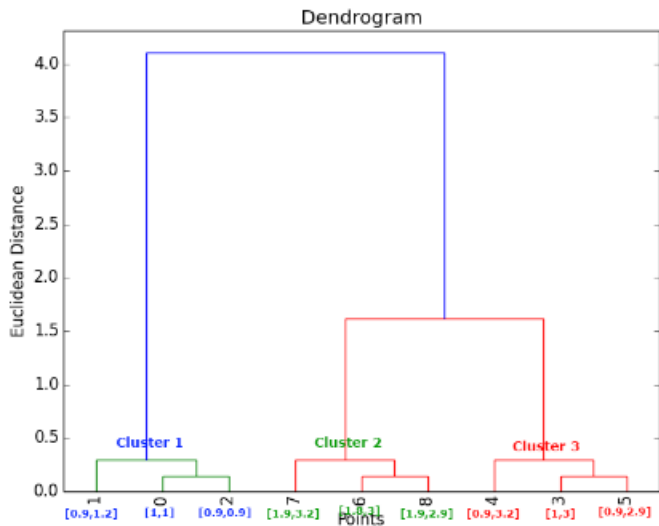


Figura: Caption

Dendrogramas

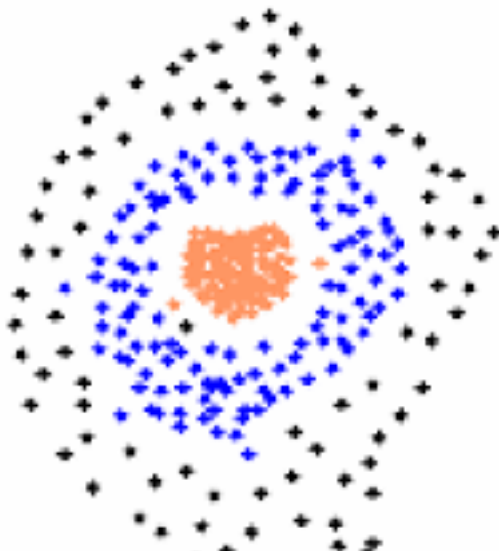


Algoritmos basados en la densidad

Los métodos basados en la densidad buscan en el espacio de datos áreas de densidad variada de puntos de datos. Los clústeres se definen como áreas de mayor densidad dentro del espacio de datos en comparación con otras regiones. Los puntos de datos en las áreas dispersas generalmente se consideran puntos de ruido y/o borde. El inconveniente de estos métodos es que esperan algún tipo de guía de densidad o parámetros para detectar los bordes de los grupos. DBSCAN y OPTICS son algunos agrupamientos basados en densidad destacados.



Algoritmos basados en la densidad



Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

DBSCAN es capaz de encontrar clústers de forma arbitraria y grupos con ruido (es decir, valores atípicos), teniendo como principal ventaja que se puede aplicar sin indicar la cantidad de clústers a generar.

La idea principal detrás de DBSCAN es que un punto pertenece a un clúster si está cerca de muchos puntos de ese grupo.



DBSCAN

Hay dos parámetros principales de DBSCAN:

- **eps (epsilon):** La distancia que especifica los vecindarios. Se considera que dos puntos son vecinos si la distancia entre ellos es menor o igual a eps.
- **minPts:** Número mínimo de puntos de datos para definir un clúster.



DBSCAN

Con base en estos dos parámetros, los puntos se clasifican como punto central, punto límite o valor atípico:

- Punto central: un punto es un punto central si hay al menos minPts número de puntos (incluido el punto en sí) en su área circundante con radio eps .
- Punto fronterizo: un punto es un punto fronterizo si es accesible desde un punto central y hay menos de minPts número de puntos dentro de su área circundante.
- Valor atípico: un punto es un valor atípico si no es un punto central y no se puede alcanzar desde ningún punto central.



DBSCAN

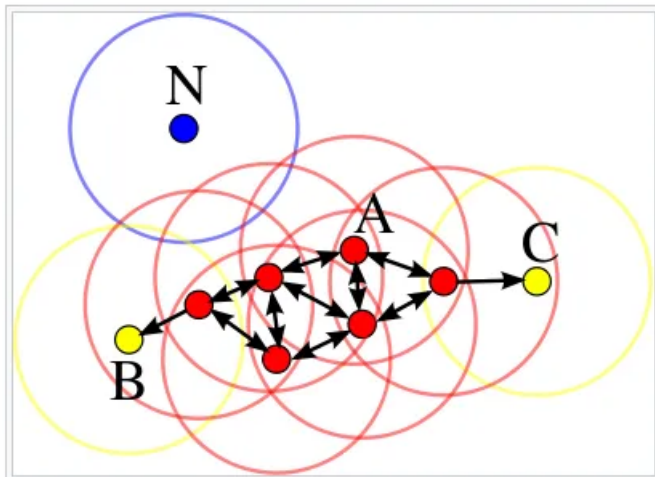


Figura: Caption

DBSCAN Funcionamiento

- Se determinan minPts y eps .
- Se selecciona un punto de partida arbitrario que no ha sido visitado y se calcula el área de vecindad usando el radio eps .
- Si hay al menos minPts número de puntos en la vecindad, el punto se marca como punto central y comienza la formación de un grupo. Si no, el punto se marca como ruido. En ambos casos ese punto se marca como visitado.
- Para este primer punto en el nuevo grupo, los puntos dentro de su vecindad de distancia eps también se vuelven parte del mismo grupo. Este procedimiento de hacer que todos los puntos en la vecindad pertenezcan al mismo conglomerado se repite luego para todos los puntos nuevos que acaban de agregarse al grupo de conglomerados.



DBSCAN Funcionamiento

- Este proceso se repite hasta que se determinan todos los puntos del grupo, es decir, se han visitado y etiquetado todos los puntos dentro de la vecindad ϵ del grupo.
- Una vez terminado el grupo actual, se recupera y procesa un nuevo punto no visitado, lo que conduce al descubrimiento de otro grupo o ruido. Este proceso se repite hasta que todos los puntos se marcan como visitados.
- Dado que al final de esto se han visitado todos los puntos, cada punto se habrá marcado como perteneciente a un grupo o como ruido.



Agrupamiento basado en centroide

En este tipo de agrupamiento, los clústers están representados por un vector central o un centroide. Este centroide podría no ser necesariamente un miembro del conjunto de datos. Este es un algoritmo de agrupamiento iterativo en el que la noción de similitud se deriva de cuán cerca está un punto de datos del centroide del agrupamiento.



K-Means



K-Means

K-means es un algoritmo basado en centroides que trata de encontrar centros de conglomerados que sean representativos de determinadas regiones de los datos.

El algoritmo alterna entre dos pasos: asignar cada punto de datos al centro de conglomerado más cercano y, a continuación, establecer cada centro de conglomerado como la media de los puntos de datos que se le asignan. El algoritmo termina cuando la asignación de instancias a los clusters ya no cambia.



K-Means

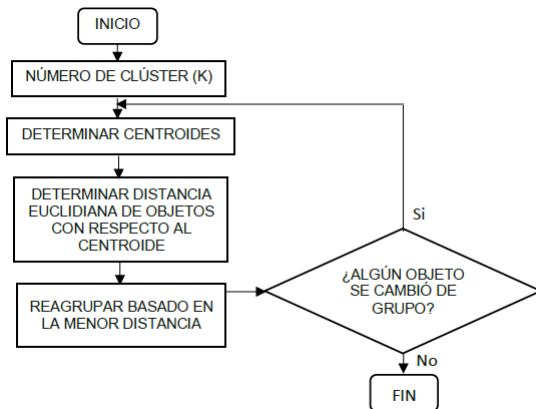


Figura: Caption

K-Means

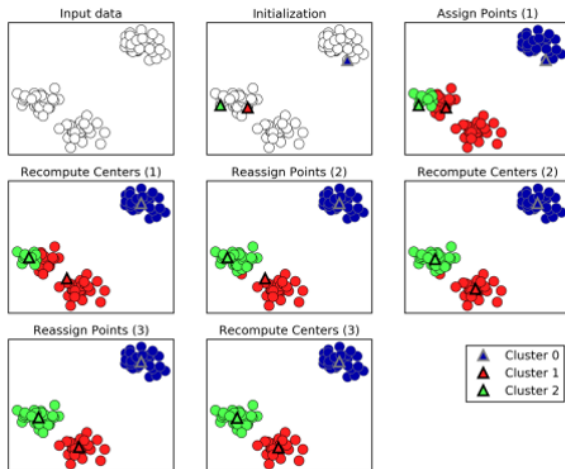


Figura: Caption

Método Elbow

El método Elbow ayuda a elegir el número óptimo de clústers, según la distribución de los datos.

Este método utiliza los valores de la inercia obtenidos tras aplicar el K-means a diferente número de Clusters (desde 1 a N Clusters), siendo la inercia la suma de las distancias al cuadrado de cada objeto del Cluster a su centroide:

$$Inercia = \sum_{i=0}^N \|x_i - \mu\|^2$$

Figura: Caption



Método Elbow

Una vez obtenidos los valores de la inercia tras aplicar el K-means de 1 a N Clusters, se representa en una gráfica lineal la inercia respecto del número de Clusters. En esta gráfica se debería de apreciar un cambio brusco en la evolución de la inercia, teniendo la línea representada una forma similar a la de un brazo y su codo. El punto en el que se observa ese cambio brusco en la inercia indica el número óptimo de Clusters a seleccionar para ese conjunto de datos.



Método Elbow

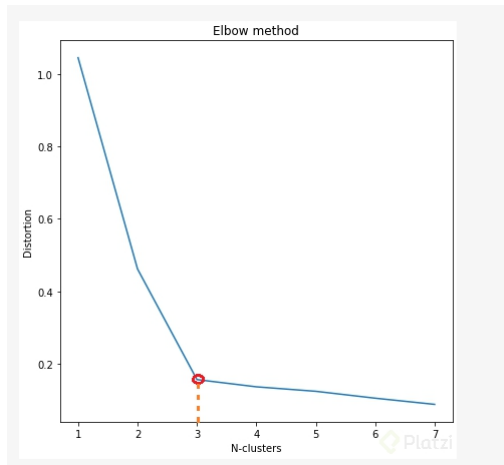


Figura: Caption

Gap

El método Gap (brecha) es similar al método del codo, cuya finalidad es la de encontrar la mayor diferencia o distancia que hay entre los diferentes grupos de objetos que vamos formando para representarlos en un dendrograma. Para ello se toma las distancias que hay de cada uno de los enlaces que forman el dendrograma y vemos cual es la mayor diferencia que hay entre cada uno de estos enlaces. Las distancias son representadas en una gráfica lineal, siendo el punto máximo, el número de Clusters óptimo para ese conjunto de datos.



Gap

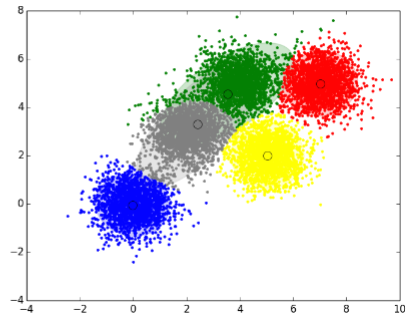
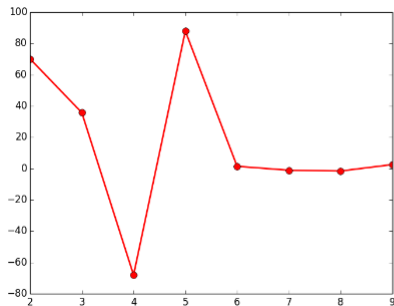


Figura: Caption

Agrupamiento basado en distribuciones

Este agrupamiento está muy relacionado con un modelado distribucional. La agrupación se basa en la noción de cuán probable es que un punto de datos pertenezca a una determinada distribución, como la distribución gaussiana, por ejemplo. Los puntos de datos en un clúster pertenecen a la misma distribución. Estos modelos tienen una sólida base teórica, sin embargo, a menudo sufren de sobreajuste. Los modelos de mezcla gaussiana, que utilizan el algoritmo de maximización de expectativas, son un famoso método de agrupamiento basado en distribución.



Agrupamiento basado en distribuciones

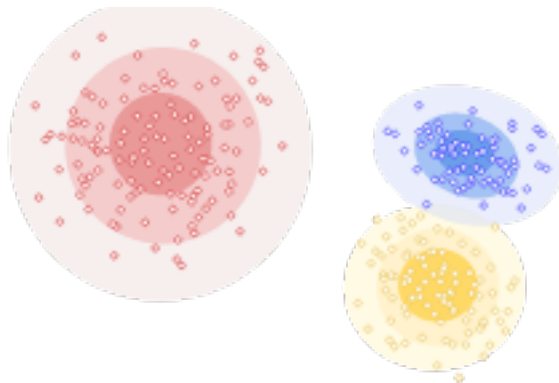


Figura: Caption

Redes



Neuronas

El cerebro es un órgano tremendamente complicado. En su interior millones de neuronas se comunican entre ellas con un mecanismo químico esencial: la sinapsis.

La sinapsis es el impulso nervioso que se produce a través de las neuronas y que posibilita su comunicación. Consiste, en esencia, en una descarga química traducida en una señal eléctrica que viaja a través de las redes neuronales de nuestro encéfalo.



Neuronas

El tipo más común de sinapsis es la sinapsis química, que funciona como sigue. Una señal neural eléctrica pre-sináptica, llega al botón sináptico. Allí, ésta hace que las vesículas sinápticas se rompan, liberándose así una sustancia llamada neurotransmisor.

Esta sustancia química se difunde a través del espacio entre las neuronas. Luego, es captada por la dendrita, en donde estimula la emisión de un nuevo impulso eléctrico, postsináptico, que se propaga hacia la derecha. Así vemos que las dendritas son las zonas receptivas de una neurona, siendo el axón una línea de transmisión, y los botones terminales comunican los impulsos a otras neuronas.



Neuronas

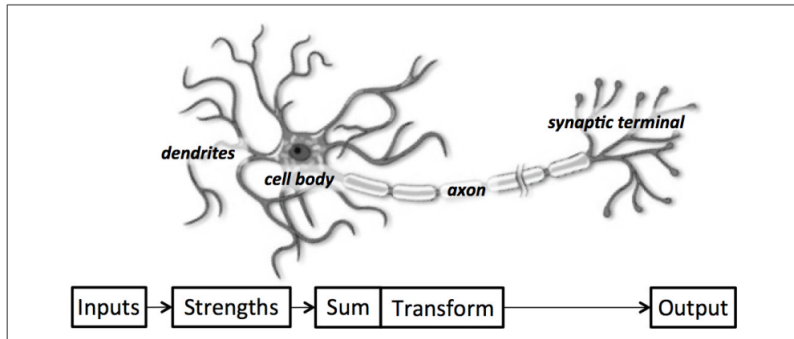


Figura: Caption

Neuronas

Cada decisión, cada estímulo, cada movimiento genera un torrente de neurotransmisores químicos en distintas partes del cerebro. Las neuronas emisoras, según el mensaje que se quiera enviar, libera un neurotransmisor en particular (adrenalina, noradrenalina, dopamina...) que atraviesa el espacio sináptico (el que hay entre neurona y neurona, el cual nunca llegan a tocar) para llegar a la receptora.

Las neuronas receptoras cuentan con una estructura química diseñada para unirse únicamente a determinados receptores, como si fuesen llaves que encajan con una única cerradura. De este modo interpretan el mensaje que les llega y es capaz de transmitirlo a la siguiente neurona.



Neuronas

Las neuronas son 6 ó 5 órdenes de magnitud más lentas que una compuerta lógica de silicio, los eventos en un chip de silicio toman alrededor de nanosegundos (10^9 s), mientras que en una neurona este tiempo es del orden de los milisegundos (10^3). Sin embargo, el cerebro compensa en forma excepcional la lentitud relativa en el funcionamiento neuronal con un número inmenso de neuronas con interconexiones masivas entre ellas.

La red resultante que es el cerebro es una estructura enormemente eficiente. La mayoría de las neuronas codifican sus salidas como una serie de breves pulsos periódicos, llamados potenciales de acción, que se originan cercanos al soma de la célula y se propagan a través del axón. Luego, este pulso llega a las sinapsis y de ahí a las dendritas de la neurona siguiente.



Perceptron



Perceptron

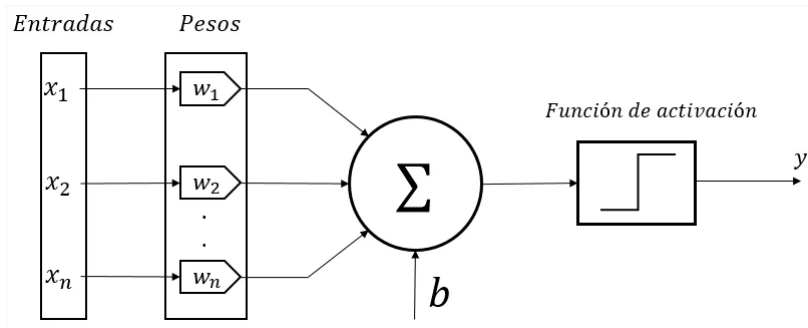


Figura: Caption

Perceptron

El perceptron se compone por:

- Entradas: Es la información que recibe el perceptron.
- Pesos: Son valores numéricos que se encargan de establecer la influencia de una entrada en la salida deseada.
- Bias: Es un parámetro que permite encontrar fácilmente la separación entre posibilidades de salida de una red neuronal. Controla qué tan predispuesta está la neurona a disparar un 1 o un 0 independiente de los pesos. Un sesgo alto hace que la neurona requiera una entrada más alta para generar una salida de 1. Un sesgo bajo lo hace más fácil.
- Función de activación: Es una función matemática que se encarga de determinar un valor de salida una vez se han procesado cada una de las entradas.



Regresión lineal

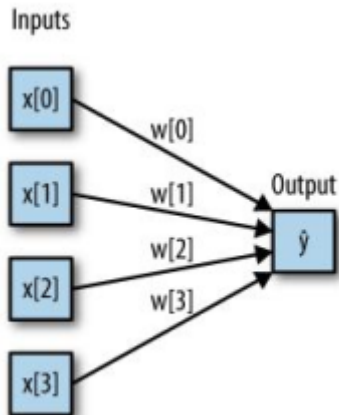


Figura: Caption

Regresión lineal

$$f(x_1, x_2) = b + w_1x_1 + w_2x_2$$

$$f(X) = b + \sum_i w_i x_i$$

$$\text{classification} = \begin{cases} 1 & \text{if } f(X) > 0 \\ 0 & \text{if } f(X) \leq 0 \end{cases}$$



Función de activación

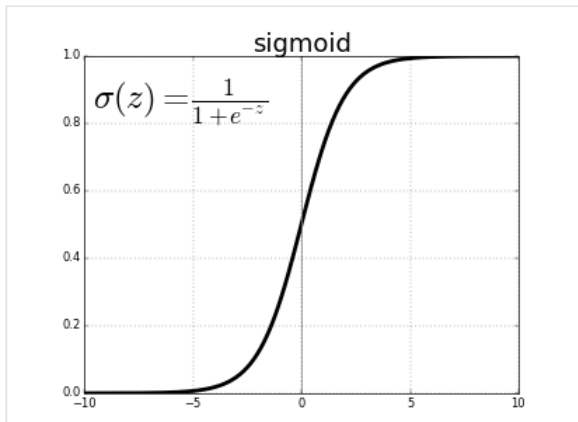
Tanto en las redes neuronales artificiales como biológicas, una neurona no sólo transmite la entrada que recibe. Existe un paso adicional, una función de activación, que es análoga a la tasa de potencial de acción disparando en el cerebro. La función de activación utiliza la misma suma ponderada de la entrada anterior, y la transforma una vez más como salida.

$$z = \sum_i w_i x_i$$



Función de activación sigmoide

Una neurona que utiliza la sigmoide como función de activación se llama neurona sigmoide. Primero establecemos que la variable z equivale a nuestra suma ponderada de entrada y después la pasamos a través de la función sigmoide.



Función de activación sigmoide

Podemos ver que $\sigma(z)$ actúa como una especie de función “aplastadora”, comprimiendo nuestra salida a un rango de 0 a 1. En el centro, donde $z=0$, $\sigma(0)=1/(1+e^0)=1/2$.

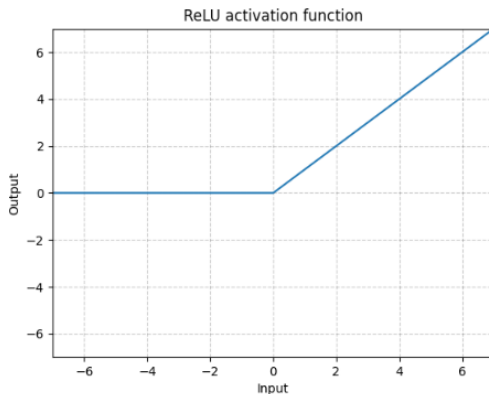
Para valores negativos grandes de z , el término e^{-z} en el denominador crece exponencialmente, y $\sigma(z)$ se aproxima a 0.

Al contrario, valores positivos grandes de z , reducen e^{-z} hacia 0, y $\sigma(z)$ se aproxima a 1.



Función de activación ReLU

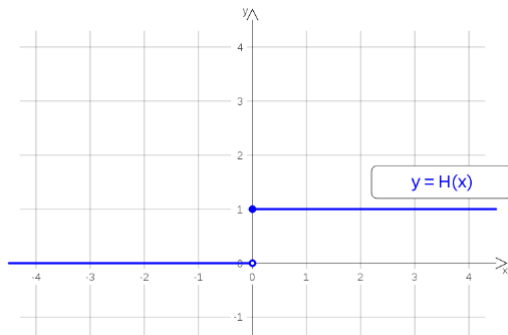
La función de activación llamada Rectified Linear Unit o ReLU, se define simplemente como $R(z) = \max(0, z)$. En otras palabras, las ReLUs permiten el paso de todos los valores positivos sin cambiarlos, pero asigna todos los valores negativos a 0.



Función de activación escalón

La función escalón de Heaviside, también llamada función escalón unitario, es una función discontinua cuyo valor es 0 para cualquier argumento negativo, y 1 para cualquier argumento positivo, incluido el cero.

$$u(x) = \begin{cases} 0 & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$$



Entrenamiento del perceptron

Para ilustrar como se entrena un perceptron, se utilizará la simulación de una compuerta AND.

Compuerta AND		
Entradas		Salidas
x_1	x_2	$y = x_1 * x_2$
0	0	0
0	1	0
1	0	0
1	1	1



Función de activación del perceptron


$$\begin{array}{ll} y = 1 & \text{net} > 0 \\ y = 0 & \text{net} \leq 0 \end{array}$$


Figura: Caption

Paso 1: Inicializar los pesos y el bias

Cada entrada del perceptron debe tener un peso. Estos valores pueden ser aleatorios. Sin embargo, es mejor empezar con valores pequeños.

- $w_1 = 10$
- $w_2 = 10$
- $bias = -8$



Paso 2 Calcular las salidas (net) con los pesos y el bias

En este paso se calcula cada salida que teniendo en cuenta los posibles valores pueden tomar las entradas del perceptron. Para este caso se realiza 4 veces debido a que la compuerta AND solo tiene ese número de posibles entradas.

$$net_n = w_1x_1 + w_2x_2 + b$$

Compuerta AND			
Entradas		Salidas	<i>n</i>
x_1	x_2	$y = x_1 * x_2$	
0	0	0	1
0	1	0	2
1	0	0	3
1	1	1	4



Entrada 1

Para la primera entrada:

- $net_1 = 10 * (0) + 10 * (0) + (-8) = -8$
- Por lo que la salida es: $y_1 = 0$ (debido a la función de activación $-8 < 0$)

Ahora hay que calcular el error $e = y' - y_1$. Donde:

$e \rightarrow$ Valor del error

$y' \rightarrow$ Salida deseada

$y \rightarrow$ Salida obtenida

- $e_1 = 0 - 0 = 0$



Cálculo del error

Hay que interpretar el error:

- Si $e_n = 0$ continuar con el cálculo para las entradas faltantes.
- Si $e_n \neq 0$ hay que corregir los pesos y el bias.
- Si todos los $e_n = 0$ y ningún valor ha cambiado en la iteración actual, se finaliza el entrenamiento.



Corrección del bias y de los pesos

Para realizar las correcciones se utilizan las siguientes ecuaciones. Donde:

Corrección de los pesos	$w_i(k + 1) = w(k) + e * x_i$
Corrección del bias	$b(k + 1) = b(k) + e$

$w_n(k + 1) \rightarrow$ *Nuevo peso*

$w(k) \rightarrow$ *Peso actual*

$e \rightarrow$ *error*

$b(k + 1) \rightarrow$ *nuevo bias*

$b(k) \rightarrow$ *bias actual*

$x_i \rightarrow$ *entrada*



Entrada 2

- $net_2 = 10 * (0) + 10 * (1) + (-8) = 2$
- Por lo que la salida es: $y_2 = 1$ (debido a la función de activación $0 > 1$)

Ahora hay que calcular el error $e = y' - y_n$. Donde:

- $e_2 = 0 - 1 = -1$



Entrada 2: Corrección

- Corrección de pesos

$$\mathbf{w}_i(\mathbf{k} + \mathbf{1}) = \mathbf{w}_i(\mathbf{k}) + \mathbf{e} * \mathbf{x}_i$$

$$w_1(k + 1) = 10 + (-1 * 0) \rightarrow w_1(k + 1) = 10$$

$$w_2(k + 1) = 10 + (-1 * 1) \rightarrow w_2(k + 1) = 9$$

Figura: Caption

- Corrección de bias

$$\mathbf{b}(\mathbf{k} + \mathbf{1}) = \mathbf{b}(\mathbf{k}) + \mathbf{e}$$

$$b(k + 1) = -8 - 1 \rightarrow b(k + 1) = -9$$

Figura: Caption



Entrada 3

$$net_3 = 10 * (1) + 9 * (0) - 9 \rightarrow net_3 = 1$$

$$net_3 = 1 \rightarrow y_3 = 1$$

$$e_3 = 0 - 1 \rightarrow e_3 = -1$$

Figura: Caption



Entrada 4

Continuar con el cálculo...



RNA



Red neuronal artificial

Una red neuronal consiste en una serie de capas de neuronas.

Específicamente, todas las neuronas de una capa se conectan a las neuronas de la siguiente capa.

En una red neuronal de 2 capas, sólo contamos las capas con entradas (omitimos la primera capa de entrada).



RNA

Nuestro cálculo comienza con la capa de entrada a la izquierda, de la cual pasamos valores a la capa oculta. De ahí, la capa oculta envía valores de salida a la última capa, que contiene el valor final.

Aunque pareciera que cada una de las tres neuronas de entrada envía múltiples valores de salida a la capa oculta, en realidad solamente hay un valor de salida por neurona. Las neuronas siempre producen un valor, independientemente de cuántas conexiones de salida tengan.



Regresión

Llamamos propagación hacia delante (en inglés, forward propagation o forward pass) al proceso por el cual una red neuronal envía su entrada a través de sus capas hacia la salida. A las redes neuronales que funcionan de esta manera se les llama red neuronal prealimentada (en inglés, feedforward neural network). Ya pronto veremos que algunas redes neuronales permiten que los datos fluyan en círculos.

Demo: Redes Neuronales [demo page] [view source]prevnext Suppose we have a dataset with one point: $x = [2.1 \ 1.7 \ 1.0]$ $y = 0.62$



Arquitectura

Más capas, más capacidad de expresión ¿Por qué son tan útiles las capas ocultas? La razón es que si no tuviéramos capas ocultas y tuviéramos que trazar una conexión directa entre nuestras entradas y nuestra salida, la contribución de cada entrada hacia el valor de salida sería independiente de las otras entradas. En la mayoría de los problemas del mundo real, las variables de entrada tienden a ser altamente interdependientes y afectan la salida de forma combinatoria y compleja. Las neuronas de las capas ocultas nos permiten capturar interacciones sutiles entre nuestras entradas. Otra manera de interpretar esta idea es que las capas ocultas representan “características” a nivel superior o atributos de nuestros datos. Cada una de las neuronas de una capa oculta sopesa sus entradas de forma diferente, y de esta manera aprende características diferentes de los datos. Nuestra neurona de salida logra capturar estas características intermediarias, no sólo las entradas originales. Al incluir más de una capa oculta, permitimos que la red neuronal pueda aprender sobre varios niveles de abstracción de los datos. En el próximo capítulo aprenderemos más



Redes convolucionales



Back Propagation

