

Conceptos previos

Mariano Crosetti

Rosario, Argentina
Universidad Austral

Maestría en Explotación de Datos y Gestión del Conocimiento

“The measure of intelligence is the ability to change.” - Albert Einstein

“Intelligence is the ability to adapt to change.” - Stephen Hawking

Contenidos I

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Síganme en <https://mariancrosetti.com>



MARIANO CROSETTI

Software Engineer
NLP & Computer Vision
ICPC Coach & LATAM Champion



LEARN



WORK



READ



CHILL

Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Criterios de evaluación

La evaluación consistirá en 4 trabajos prácticos.

Serán asignados las classes: viernes 14/06, viernes 28/06, sábado 20/07.

Los trabajos pueden realizarse en grupo de hasta 4 personas.

Pueden ser entregados hasta el 04/08.

Large Language Models			
Fecha	Horario	Sesión	Modalidad
8-Jun	9 a 13	Clase 1	Presencial
14-Jun	17 a 21	Clase 2	Online
28-Jun	14:15 a 18	Clase 3	Presencial
29-Jun	9 a 13	Clase 4	Presencial
20-Jul	9 a 12:45	Clase 5	Presencial
26-Jul	9 a 13	Clase 6	Online
3-Aug	9 a 13	Clase 7	Online

La materia va a ser fácil de aprobar.

Los trabajos tendrán una parte obligatoria sencilla y una parte opcional que no influirá en la evaluación de la materia.

Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- **Objetivos**

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Objetivos

Los objetivos de esta materia son:

- Que todos puedan entenderla: independientemente de su background.
- Con aplicaciones prácticas: que permita resolver problemas reales.
- Que tengan una buena base: para poder construir sobre ella.
- Incluir material opcional: para quién quiera ampliar su formación.



Nos concentraremos en Natural Language Processing. No hablaremos de temas de ciencias de datos, Machine Learning¹ o Deep Learning² que no traten de procesar texto.

¹El 90 % de los problemas en la práctica son de tabular data y el 90 % puede ser resuelto usando métodos como XGBoost

²No trataremos por ejemplo otras modalidades como Computer Vision

Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

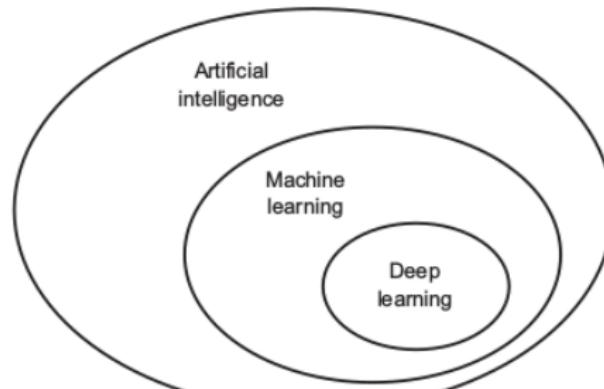
3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Definición IA, ML, DL

- **Inteligencia Artificial:** “una rama de las Ciencias de la Computación que tiene como objetivo simular comportamiento inteligente en una computadora.”
- **Machine Learning:** Estudio de algoritmos que mejoran automáticamente su rendimiento gracias a la experiencia. Por ejemplo viendo casos del problema resuelto. Aprenden de los **datos suministrados**.
- **Deep Learning:** Una familia de algoritmos de Machine Learning también llamada redes neuronales. Tienen la propiedad de poder manejar datos crudos (píxeles de imágenes, ondas de sonido) y aprender las representaciones que sean útiles para el problema a resolver.

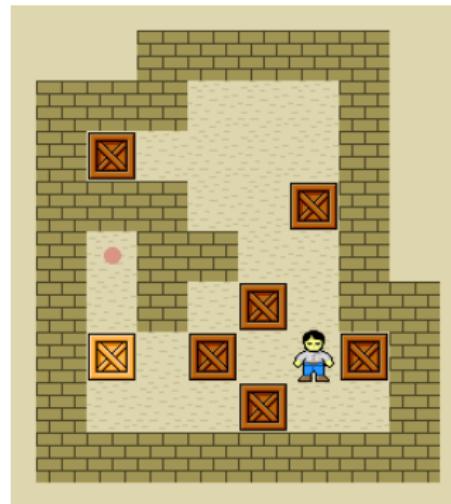


Ejemplo de Inteligencia Artificial

“Una rama de las Ciencias de la Computación que tiene como objetivo simular comportamiento inteligente en una computadora.” - Arthur Samuel, 1956

Ejemplos:

- Jugar al ta-te-ti probando todas las posibilidades.
- Resolver el cubo rubik.
- Resolver otros acertijos como SOKOBAN (juego donde hay que mover ciertas cajas a ciertas posiciones).



Limitaciones de los sistemas expertos

Los primeros sistemas de “*Inteligencia Artificial*”:

Gran cantidad de reglas que los hacían muy efectivos para resolver ciertos problemas

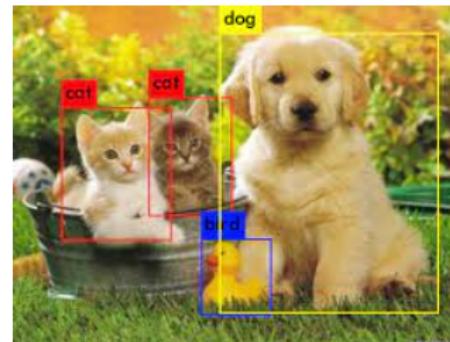
Requieren **pasos bien definidos y son lógicos.**

Entre el '50 y '80 se creyó que con un conjunto suficientemente grande de reglas se podría lograr una inteligencia al nivel de los humanos.

Este enfoque falló en problemas que no poseen pasos bien lógicos definidos:

- Reconocimiento de objetos en imágenes
- Transcripción de texto a partir de sonido

Son problemas que los humanos lo pueden realizar sin dificultad!



Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- **Terminología básica**
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

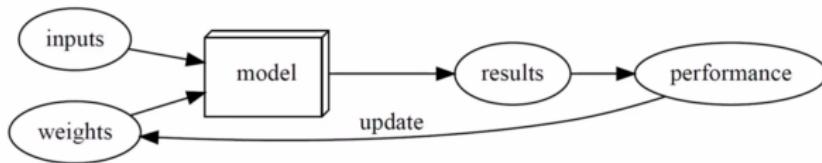
3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Machine Learning - Terminología

La idea del Machine Learning (Aprendizaje Automatizado) es crear un sistema configurable, que modifique su configuración para mejorar la performance sobre casos ejemplo del problema a resolver.



Terminología de Machine Learning:

- Al sistema lo llamaremos **modelo**. Al construirlo, se hacen suposiciones sobre la distribución de los datos, esto es lo que se denomina como **sesgo inductivo**.
- La configuración que irá cambiando suele referir como **parámetros, pesos** (weights).
- Los casos ejemplos están compuesto de **entradas** (inputs) para las cuales se conoce la respuesta deseada también llamada **objetivo** (target). Estos ejemplo constituye el denominado **conjunto de datos**.
- Las entradas suelen componerse por diversos valores denominados **características** (features).
- Los resultados también suele llamarse **predicciones**.
- La función que calcule la performance la llamaremos **métrica**.
- El algoritmo encargado de los *update* de modo que los pesos mejoren la performance del modelo se llama **algoritmo de optimización**.
- El algoritmo optimizará una **función de costo u objetivo**. La misma puede ser distinta a la métrica. Generalmente tiene propiedades matemáticas deseables para el algoritmo.
- A todo el proceso se lo llama **entrenamiento** del modelo.

Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- **Generalización y sobreajuste**
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

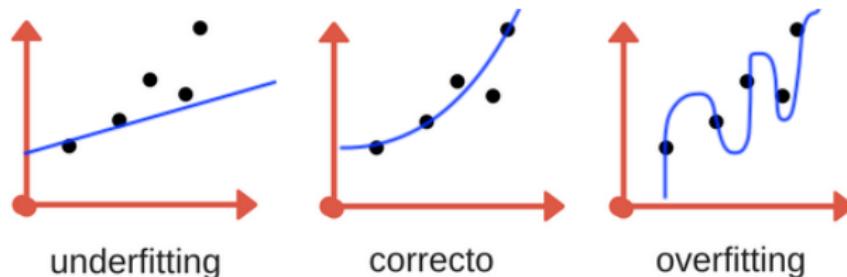
Machine Learning - Generalización

El objetivo es usar el modelo en casos futuros:

- La acción de aplicar el modelo a un ejemplo al cuál no se conoce su resultado suele denominarse **inferencia**.
- Uno desearía conocer la **performance esperada en los casos futuros**.
- Se reserva un conjunto de datos no utilizado durante el entrenamiento y se calcula la métrica sobre dicho conjunto. Esto se conoce como **evaluación**.
- **Nunca se evalúan las métricas sobre datos de entrenamiento.**

Si una buena performance en los casos de entrenamiento se extrae a buenos resultados sobre los casos no vistos se dice que el modelo **generaliza** adecuadamente.

Cuando un modelo se ajusta demasiado a los datos de entrenamiento y no generaliza bien se denominada **sobreajuste (o overfitting)**.



En general modelos simples tienden a subajustar. Modelos complejos, que pueden adaptarse mucho a los datos de entrenamiento tienden a sobreajustar. Esto es lo que se conoce como dilema sesgo-varianza.



Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test**
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

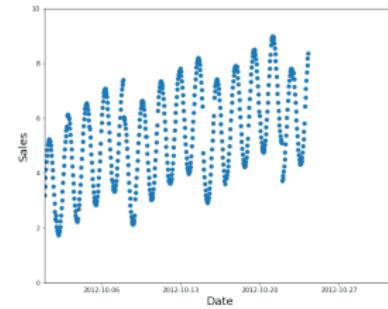
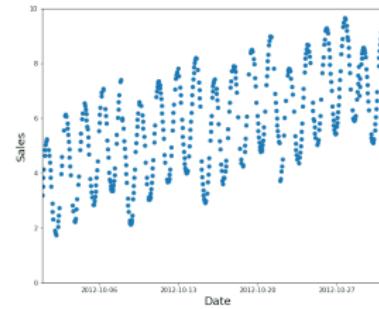
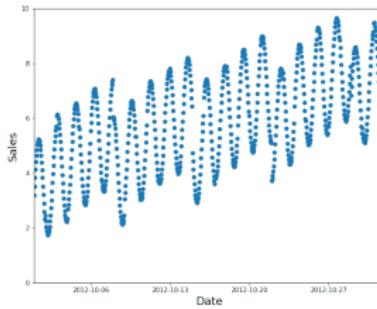
Machine Learning - Hiperparámetros y validación

Existen configuraciones que no son ajustadas por el algoritmo de optimización.

Son "parámetros" que actuarán sobre los parámetros.

- Se denominan **hiperparámetros**.
- Para elegirlos se evalúa el modelo en cada configuración y se elige el mejor.
- El conjunto de datos sobre el cuál se evalúa cada configuración se denominad **conjunto de validación**.
- Si las configuraciones son muchas corremos el riesgo de estar "sobreajustando" manualmente los hiperparámetros y reportar resultados sesgados: que mejores resultados en el conjunto de validación no se traduzca a mejores resultados en general.
- Lo metodológicamente correcto entonces es tener otro conjunto de datos, reservado para una evaluación final sobre la configuración final elegida para reportar los resultados.
- Este conjunto se lo denomina **conjunto de test**.
- A menudo se toma una partición aleatoria de los datos.

Esto no es suficiente, por ejemplo claramente en las series temporales En una aplicación real querremos predecir valores *futuros*. Por lo que una correcta partición tomaría un rango continuo de los últimos valores para confeccionar un conjunto de validación representativo.



Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Machine Learning - Ejemplo

Predecir el precio de una propiedad dado características como:

- m^2 : metros cuadrados.
- h : cantidad de habitaciones.
- t : tipo de propiedad (casa, ph, departamento, casa de pasillo)

Podríamos pensar que el precio es una fórmula:

$$\text{predice_precio}(m^2, h, t) = m^2 \times w_1 + h \times w_2 + t \times w_3$$

Nuestros datos serán una lista de $\{m_i^2, h_i, t_i, \text{precio}_i\}$ con $1 \leq i \leq N$.

Podríamos medir la performance calculando la suma de los valores absolutos de las diferencias:

$$\text{performance} = \sum_{i=1}^N |\text{predice_precio}(m_i^2, h_i, t_i) - \text{precio}_i|$$

- La ecuación es nuestro **modelo**.
- Asumir que el precio es lineal a dichas variables es parte de nuestro **sesgo inductivo**.
- m^2, h, t son las **características** o features.
- w_1, w_2 y w_3 serán los **pesos**. Encontraremos valores de los mismo que hagan que la predicción sea buena para ciertos datos conocidos.
- Los valores de m^2, h, t de las casas que tenemos en nuestros datos son las **entradas**.
- Los valores que arrojen nuestra fórmula son las **predicciones**.
- Los valores de precios reales (según los datos) de las casas son los **objetivos**.

Machine Learning - Ejemplo (cont.)

$$\text{performance} = \sum_{i=1}^N |\text{predice_precio}(m_i^2, h_i, t_i) - \text{precio}_i|$$

- *performance* será nuestra **métrica**.
- Podríamos utilizar el algoritmo de mínimos cuadrados para encontrar pesos adecuados. Este será el **algoritmo de optimización**.
- El mismo optimiza el error cuadrático medio:

$$MCE = \frac{1}{N} \sum_{i=1}^N (\text{predice_precio}(m_i^2, h_i, t_i) - \text{precio}_i)^2$$

Este será nuestra **función de costo**.

- Podemos probar con distintas decisiones de diseño:
 - Normalizar o no las características.
 - Dividir t en varias características booleanas “es casa”, “es ph”, “es departamento”, etc.
 - Incluir características de interrelación como $m^2 \times h$.

Estas decisiones presentan diferentes configuraciones que constituyen los **hiperparámetros** a probar.

Machine Learning - Más terminología

Es común hacer la diferencia de Aprendizaje Automatizado:

- **Supervisado:** se suministran los resultados esperados para las entradas ejemplo.
- **No supervisado:** sólo hay entradas (no hay salidas) y el objetivo es entender la estructura fundamental de los datos y aprender relaciones ocultas importantes. Ejemplo: dada una colección de textos agruparlos según si son del mismo tópico (pero no tenemos los tópicos!).

Dependiendo de la naturaleza de las predicciones hablaremos de:

- **Regresión:** si las predicciones son valores continuos (ejemplo: precio).
- **Clasificación:** Si los objetivos pertenece a un dominio de valores discreto. A las salidas esperadas los llamaremos también **etiquetas** (labels). Al conjunto de salidas esperadas se lo denomina **clases**. Las predicciones suelen ser la probabilidad de pertenecer a cada clase.

Hablaremos de **clasificación binaria** si las etiquetas poseen sólo dos valores posibles.



Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Redes neuronales

El éxito de *predice_precio* depende fuertemente de la elección de las características (m^2 , h , t). Para elegirlas utilizamos nuestro conocimiento (o sentido común) acerca de qué características de una casa puede influir en el precio.

No es tan fácil elegir características para el problema de reconocimiento de imágenes (incluso problemas fáciles como reconocer dígitos). Algunas podrían ser:

- Cantidad de “agujeros” (el 0 y el 8 poseen “agujeros”, el 7 y el 5 no).
- Cantidad de bordes en dirección vertical, horizontal, diagonal.
- Cantidad de píxeles blancos (hay números como el 1 que parecen que poseen menos píxeles que otros como el 8)



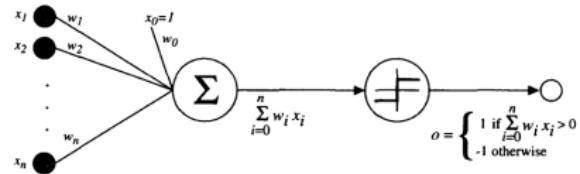
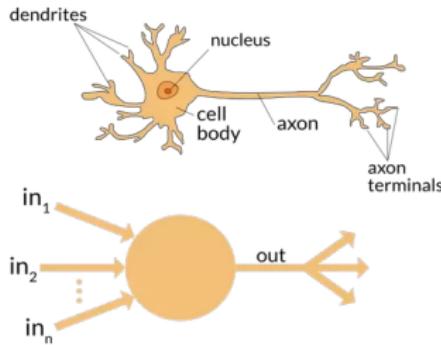
Desventajas:

- Las características dependen del problema. No podemos reutilizarlas para otro problema de reconocimiento (por ejemplo reconocer fotos gatos de perros).
- Necesitamos programar programas que extraigan cada una de esas características.

Redes neuronales - Historia

Si no podemos ganarle al cerebro... imitémoslo!

En 1943 Walter Pitts propone un modelo simplificado de la neurona cerebral:



$$o(x_1, \dots, x_n) = \begin{cases} 1 & \text{si } w_0 + w_1 x_1 + w_2 x_2 + \dots + w_n x_n > 0 \\ -1 & \text{en caso contrario} \end{cases}$$

Redes neuronales - Historia (cont.)

En 1986 en “*Parallel Distributed Processing (PDP)*” se decía que un sistema con la capacidad de imitar el comportamiento del cerebro poseería:

- ① Un conjunto de unidades de procesamiento.
- ② Un estado de activación de las mismas.
- ③ Una función de salida de cada unidad de procesamiento.
- ④ Un patrón de conectividad de dichas unidades.
- ⑤ Una regla de propagación de las activaciones a través de la red de conexiones (forma de transformar las salidas de una unidad en entradas de la otra).
- ⑥ Una regla de activación para combinar las entradas de las unidades, el estado actual y producir las salidas de la unidad.
- ⑦ Una regla de aprendizaje donde se modifican los patrones de conectividad por medio de la experiencia.
- ⑧ Un entorno en donde el sistema opera.

Todas estas características aplican a la formulación actual de **Redes Neuronales Artificiales**.

Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- **Formulación actual**
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Redes neuronales feed forward

Queremos aproximar \hat{f} con n_{in} atributos de entrada y n_{out} de salida:

$$\hat{f} : \mathbb{R}^{n_{in}} \rightarrow \mathbb{R}^{n_{out}}$$

Para ello aplicaremos sucesivas funciones denominadas **capas**:

$$f^j : \mathbb{R}^{w_{j-1}} \rightarrow \mathbb{R}^{w_j}$$

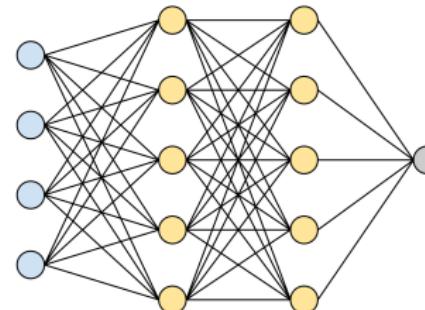
Los resultados \hat{h}^j de cada capa son denominados **activaciones**:

$$\hat{h}^{j+1} = f^{j+1}(\hat{h}^j)$$

$\hat{h}^0 = x$ a menudo se la llama **capa de entrada**. Son las entradas del modelo.

A la dimensión w_j de la capa \hat{h}^j se la conoce como **anchura de la capa**.

El número m de capas se denomina **profundidad**.



Entrada

Capas ocultas

Salida

Redes neuronales - Activaciones

Cada una de dichas funciones consiste en una transformación afín de la salida de la capa anterior, seguida de una función no lineal ϕ^j (denominada **función de activación**):

$$f^j(\bar{h}^{j-1}) = \phi^j(W^{j\top} \bar{h}^{j-1} + b^j)$$

El vector de las $\{h_i^m\}_{i=1}^{n_{out}}$ constituyen la salida de nuestra red:

$$out_i = h_i^m \quad 0 \leq i < n_{out}$$

Los $W^j \in \mathbb{R}^{w_{j-1} \times w_j}$, $b^j \in \mathbb{R}^{w_j}$ se denominan respectivamente **pesos** y **biases**.

Son los parámetros a entrenar.

La profundidad m, las anchuras w_j , las funciones de activación, son hiperparámetros.
Constituyen lo que se conoce como **arquitectura de la red**.

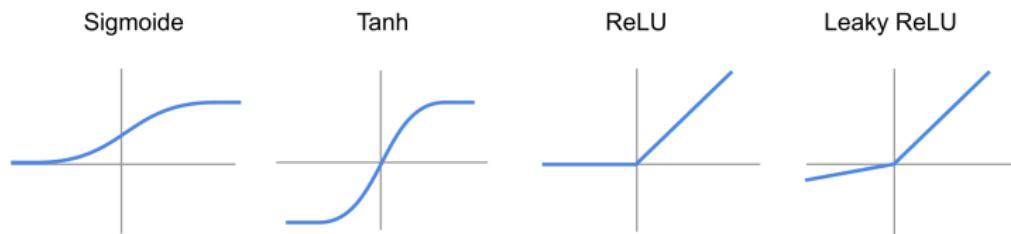


Figura: Diferentes alternativas para la función de activación.

Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico**
- Algoritmo de backpropagation
- Por qué todo esto funciona?!

4 Natural Language Processings

Descenso por gradiente

Como elegir W^j, b^j óptimos? (que ajusten nuestros datos)

Llamaremos W al conjunto de todos los parámetros.

Supongamos que tenemos una función de costo J :

$$J : \mathbb{R}^{|W|} \rightarrow \mathbb{R}_0^+$$

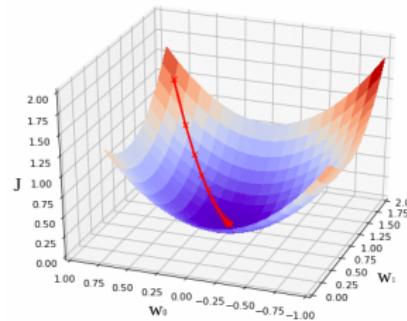
J calculará las predicciones para cada ejemplo y lo comparará de *alguna forma* con las salidas.

Nota: notar que J recibe como entrada los *parámetros* del modelo. El conjunto de datos no es una entrada de la función J , se considera fijo para el análisis de J .

Para optimizar J se utilizará el método de descenso por gradiente³:

$$W_i \leftarrow W_i - \eta \nabla J_i(W)$$

∇J es la dirección (en el espacio de pesos) donde ocurre el mayor incremento de J de manera local (dentro de un entorno infinitamente pequeño).



³Recordar que el concepto de *gradiente* para una función $\mathbb{R}^n \rightarrow \mathbb{R}$ es un vector donde cada componente es la derivada de la *-única-* salida respecto de cada componente de la entrada.

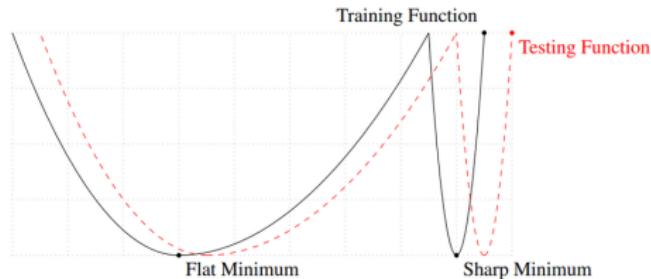


Descenso por gradiente (cont.)

Notar que en nuestro análisis de J los datos se consideraban fijos.

Luego tendremos diferentes J_{train} y J_{test} .

Optimizaremos en J_{train} y esperaremos que eso nos de valores pequeños de J_{test} .



Los mínimos suaves supondrán, probablemente, cambios menores en el resultado del costo producto de las variaciones entre funciones de costo al variar los datos, y por ende presentan un **mayor poder de generalización** que los mínimos afilados.

El método de descenso por gradiente requiere hacer muchos pasos de entrenamiento o **iteraciones** que consiste en:

- ① Se calculan las predicciones.
- ② Se calcula el gradiente del costo.
- ③ Se actualizan los pesos.

Funciones de costo

Por función de costo J necesitamos que sea diferenciable.

Métricas tales como tasa de ciertos son útiles para evaluar la performance pero no para optimizar: son funciones escalonadas y sus gradientes son nulos en gran parte de su dominio. En cambio se utilizan como funciones de costo:

- En problemas de **regresión** el ya definido **Error Cuadrático Medio (MSE)**. Además de ser una función diferenciable, presenta buenas propiedades de concavidad.
- En problemas de clasificación binaria **Binary Cross Entropy**. En caso de clasificación binaria tenemos:

$$BCE(y, p) = \sum_{x_i \in D_v} y_i \log(p_i) + (1 - y_i) \log(1 - p_i)$$

Como $y_i \in \{0, 1\}$ (clasificación binaria), las expresiones y_i y $1 - y_i$ simplemente cumplen el rol de "elegir" que término sumar.

Intuitivamente:

- Si el \log no estuviera es una "tasa de aciertos" continua.
- Mide que distancia hay entre la probabilidad predecida y la predicción correcta.
- La diferencia entre un error de 0,001 y 0,01 es poca en términos absolutos. Pero significa un error $10\times$ menor. El \log hace que se ponderen las diferencias relativas.

Sustento matemático: las fórmulas se derivan de los estimadores de máxima probabilidad:

$$\hat{\theta} = \operatorname{argmax}_{\theta} P(\text{datos}|\theta) = \operatorname{argmax}_{\theta} \prod_{\text{dato}_i} P(\text{dato}_i|\theta) = \operatorname{argmax}_{\theta} \sum_{\text{dato}_i} \log(P(\text{dato}_i|\theta))$$

Descenso por gradiente estocástico

Computar el gradiente del error sobre todo el conjunto de entrenamiento es costoso.

Solución: se partitionan los datos y para cada subconjunto se ejecuta un paso de entrenamiento.

Ventajas respecto a usar un sólo ejemplo por iteración:

- Gradientes más estables. Mejor aproximación del gradiente real.
- Paralelizar en **GPU** los cómputos

Dichas particiones se las denominan ***mini-batch***.

Las iteraciones de todas las particiones constituyen lo que se denomina **época**.

La cantidad de épocas es la cantidad de veces que el modelo ajusta el conjunto de datos de entrenamiento, por lo que al aumentarlo hará más probable que el modelo comience a sobreajustar.

La elección del **tamaño del *mini-batch*** supone una relación costo beneficio, al incrementarlo:

- Cada iteración de entrenamiento consumirá más tiempo y memoria.
- Tendremos menos iteraciones por época, o sea menos “pasos” de descenso.
Probablemente necesitaremos entrenar más épocas.
- El gradiente calculado $\nabla J_{minibatch}(w)$ será más estable y representará una mejor estimación del gradiente verdadero $\nabla J(w)$ (sobre todo el conjunto de datos).

Las particiones son aleatorias y cambian entre diferentes épocas. Éstas variaciones son útiles para evitar mínimos filosos que no generalizan bien.

El método recibe el nombre de **descenso por gradiente estocástico**.

Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation**
- Por qué todo esto funciona?!

4 Natural Language Processings

Algoritmo de backpropagation

Las redes neuronales son composición de sucesivas funciones. Es natural que los gradientes se calculen usando la **regla de la cadena**:

$$E_1(w) = f(E_2(w))$$

En cada iteración del entrenamiento, sólo necesitamos la derivada del error respecto a cada peso **aplicada en la asignación actual de pesos \hat{w}** :

$$\frac{\delta E_1}{\delta w}(\hat{w}) = \frac{\delta f}{\delta E_2}(E_2(\hat{w})).\frac{\delta E_2}{\delta w}(\hat{w})$$

La regla de la cadena nos da una forma inductiva de calcular los gradientes del error respecto de cada capa, haciendo uso de los gradientes ya calculados de capas posteriores.

Recordamos la formulación de nuestra red neuronal feed forward:

$$\bar{h}^0 = \bar{x}$$

$$\bar{h}^{j+1} = f^{j+1}(\bar{h}^j)$$

Luego podemos calcular los gradientes $\frac{\delta J}{\delta h^j}$ como:

$$\frac{\delta J}{\delta h^j} = \frac{\delta J}{\delta h^{j+1}}(h^{j+1}).\frac{\delta h^{j+1}}{\delta h^j}(h^j)$$

Una vez que los gradientes respecto a cada capa h^j han sido calculados, se pueden calcular $\frac{\delta J}{\delta W^j}$ y $\frac{\delta J}{\delta b^j}$ si conocemos la derivada de ϕ^j aplicando la regla de la cadena sobre la fórmula de la capa:

$$\bar{h}^j = f^j(\bar{h}^{j-1}) = \phi^j(W^{j\top} \bar{h}^{j-1} + b^j)$$

Algoritmo de backpropagation (cont.)

Podemos calcular los gradientes respecto de cada una de las activaciones, empezando por las capas más profundas y conociendo la derivada respecto a capas anteriores.

En éste cálculo la información fluye “*hacia atrás*” y es por eso que se denomina **algoritmo de backpropagation**.

En contraste a éste concepto se llama ***forward propagation*** al cálculo de las predicciones, ya que la información en este caso fluye “*hacia adelante*”.

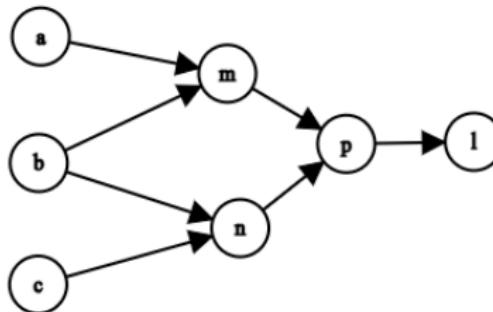
Notar que en los cálculos intervienen los valores h_j , lo que significa que muchas veces será necesario guardar el valor de las activaciones (o al menos cierta información de ellas) para el cálculo posterior de los gradientes.

$$m = a + b$$

$$n = \max(b, c)$$

$$p = m \times n$$

$$l = p^2$$



Contenidos

1 Presentación de la materia

- Presentación
- Esquema de clases y criterio de aprobación
- Objetivos

2 Repaso de Machine Learning

- Definición IA, ML, DL
- Terminología básica
- Generalización y sobreajuste
- Conjuntos de validación y test
- Ejemplo

3 Introducción a Redes Neuronales

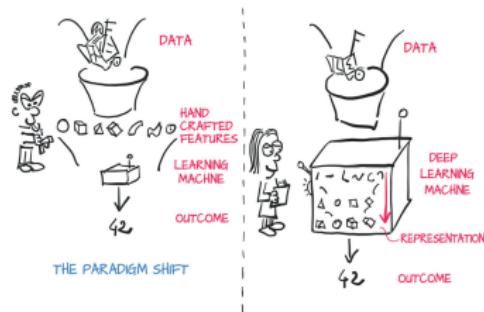
- Motivación
- Formulación actual
- Descenso por gradiente estocástico
- Algoritmo de backpropagation
- **Por qué todo esto funciona?!**

4 Natural Language Processings

Aprendiendo características

Todo esto pinta chamuyo, por qué funciona?!

- Las redes neuronales han superado el estado del arte en muchos dominios (imágenes, audio, texto). Qué tienen estos dominios en común? Son datos no estructurados⁴
- Los resultados asombrosos ocurrieron *antes* de que se tuviera una explicación de por qué funcionan tan maravillosamente.⁵
- La clave detrás del éxito de las redes neuronales es lo que se denomina **representation learning**: cada capa se entrena en calcular características útiles que sirven, a su vez, para que la capa subsiguiente aprenda características más complejas, y así sucesivamente, hasta la última capa que calcula la predicción.

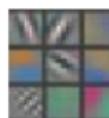


⁴No obstante también han demostrado buenos resultados en algunos problemas que implicaban datos tabulares.

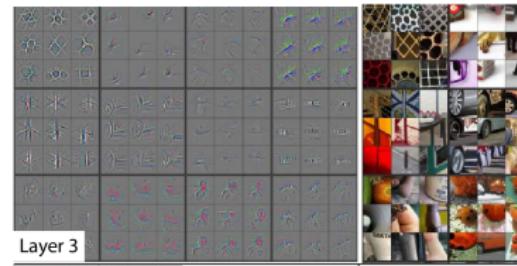
⁵Por eso se plantea que el Machine Learning sigue un paradigma empírista en contraste del paradigma racionalista que es más dominante en las ciencias de la computación (ver [Amnon H Eden])

Aprendiendo características (cont.)

- Una de las primeras visualizaciones que arrojaron luz sobre este fenómeno fue el trabajo de [Zeiler and Fergus] mostrando las secciones de imágenes que más ponderaba la red AlexNet.
- Se observa cómo las primeras capas aprenden a reconocer características básicas (diagonales, horizontales y verticales de ciertos colores), las capas intermedias características más complejas (espirales, curvas y diferentes texturas), hasta las últimas capas que aprenden a extraer conceptos semánticos de alto nivel (flores, perros, personas, vehículos, etc).

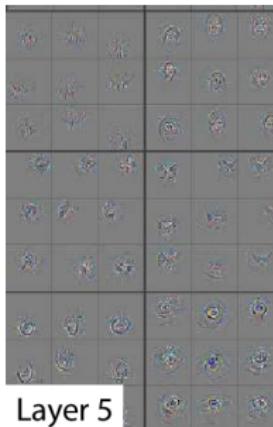
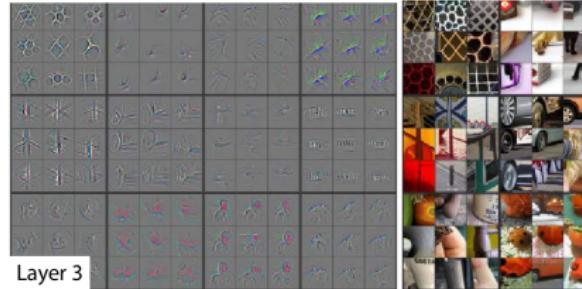


Layer 1



Aprendiendo características (cont.)

- Muchas de las características aprendidas por el modelo son parecidas a las que se solían extraer manualmente por algoritmos anteriores de análisis de imágenes.
- El trabajo nos muestra no solamente una forma de “visualizar qué aprenden los filtros” en las CNN, sino también prueba el hecho de que las representaciones aprendidas son patrones deseables, componibles y de creciente complejidad.



Layer 5



Natural Language Processing

Como las computadores codifican el significado del texto?

- **Semántica Denotacional**

semantica : Simbolo → Significado

Como se implementa esto en una computadora?

- Conjunto de sinónimos.
- Relaciones “es un” entre ellos (“ontología”). Ej: perro “es un” animal.

- **Deep Learning Approach** Recordemos que una red neuronal es una función:

$$\mathbb{R}^{1000} \rightarrow \mathbb{R}^{100} \rightarrow \dots \rightarrow \mathbb{R}^{\text{clases}}$$

- Si queremos meter texto en una red neuronal necesitamos representarlo como un vector \mathbb{R}^d
- Para las imágenes, la unidad básica son los píxeles. Cada uno representa que tanto brillo tiene una posición de la cuadrícula de la imagen.
- Para el texto la unidad básica son las palabras. Su número respectivo podría ser la posición en el diccionario.
- Notar que a diferencia de lo que sucede con los píxeles, la magnitud de esos números no nos importa: no nos dice nada que dos palabras estén “cerca” en el diccionario. Son categorías.
- Las categorías se suelen representar en Machine Learning con vectores “one-hot”

Natural Language Processing

Tendremos entonces

OpenAI

jpa

Dudas?

Dudas? Síganme en <https://mariancrosetti.com>



MARIANO CROSETTI

—
NLP & Computer Vision
SWE Distributed Systems
ICPC Coach & LATAM Champion



LEARN



WORK



READ



CHILL