



Cómo entrenar a tu modelo de IA

~desde cero~



SNGULAR

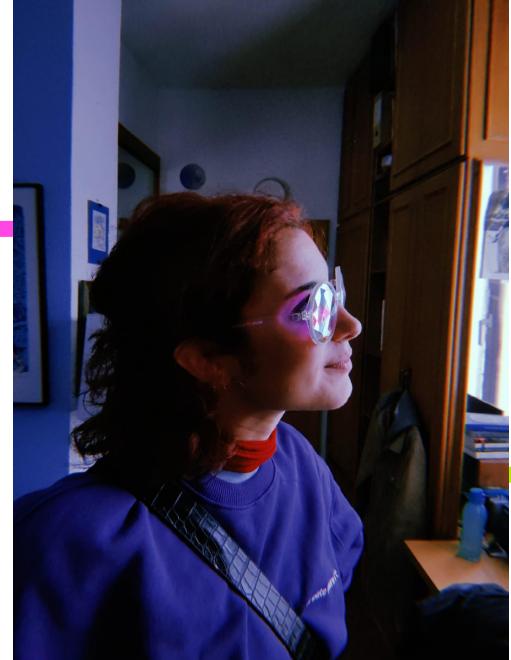
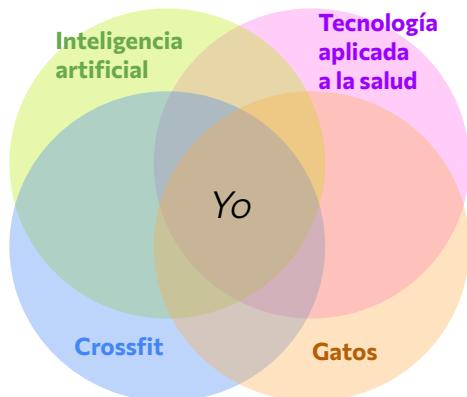
Teresa Lobo Alonso
ML Engineer @ Sngular

Hola!

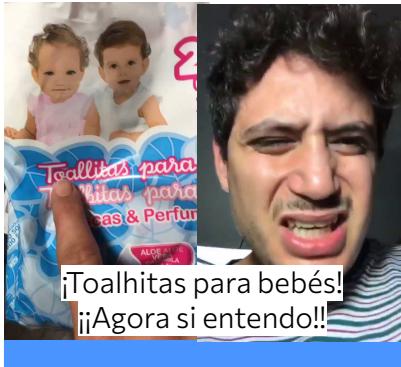
Soy Teresa (Lobo Alonso) :)

 ML Engineer @ Sngular

 M.Sc Biomedical Engineer
B.Sc. Physics



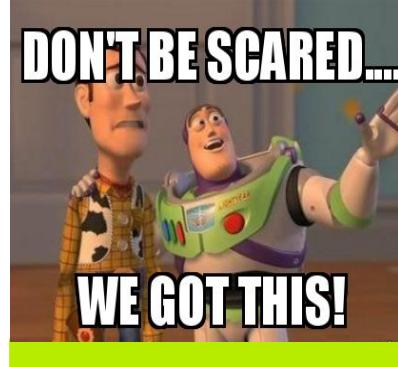
Qué podéis esperar de este taller



Entender conceptos básicos de DL



Conocimientos prácticos



Perder el miedo



Pasar un buen rato

Qué no esperar de este taller



Matemáticas

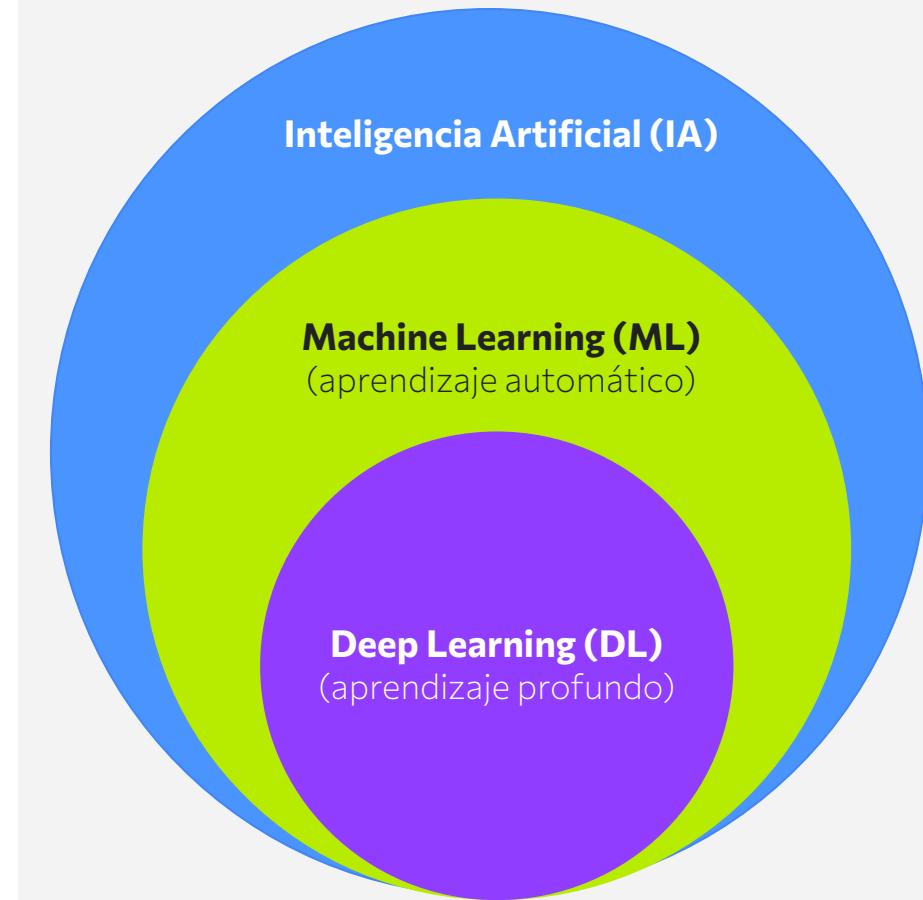
Ser un experto en IA

66 Esto es un taller de IA (o ML) centrado en DL



¿IA? ¿ML? ¿DL?, ¿de qué me hablas?

La **inteligencia artificial (IA)** es un **campo de la informática** que se enfoca en **crear sistemas con capacidades** iguales o superiores a las de los seres **humanos**.



¿IA? ¿ML? ¿DL?, ¿de qué me hablas?

La **inteligencia artificial (IA)** es un **campo de la informática** que se enfoca en **crear sistemas con capacidades** iguales o superiores a las de los seres **humanos**.

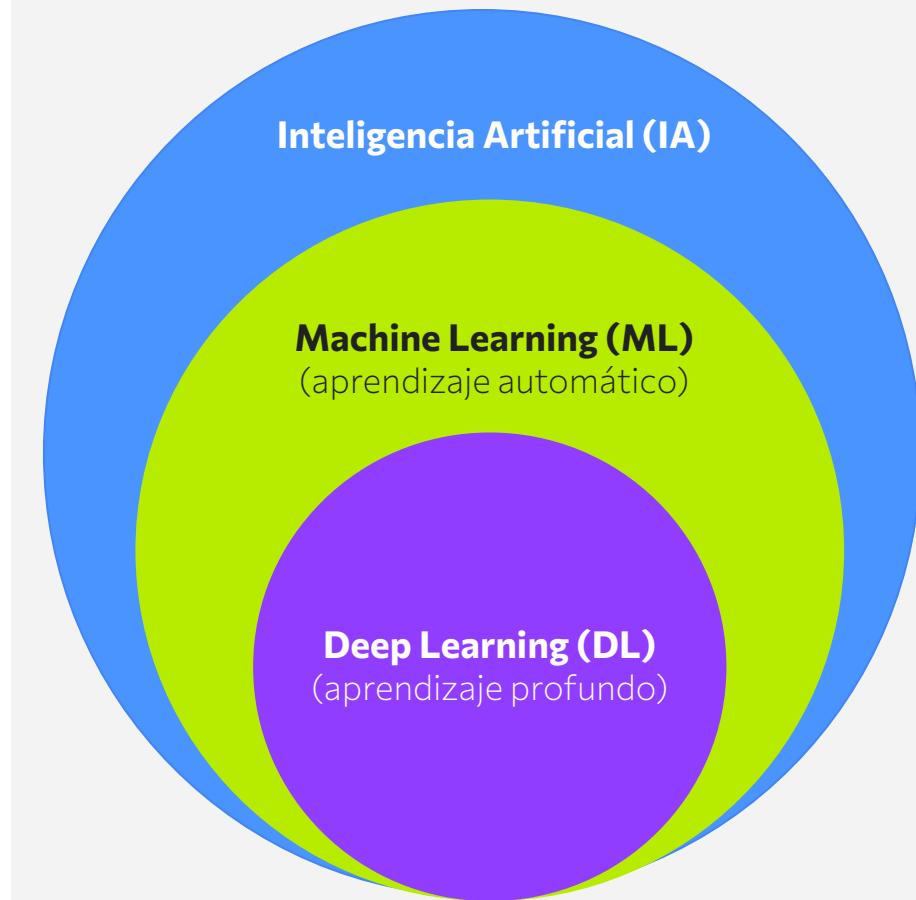
Aprendizaje automático o Machine Learning (ML):

Clases:

Supervisado, no supervisado, semi-supervisado y de refuerzo.

Ejemplos de Algoritmos:

SVM, KNN, árboles de decisión, redes neuronales...



¿IA? ¿ML? ¿DL?, ¿de qué me hablas?

La **inteligencia artificial (IA)** es un **campo de la informática** que se enfoca en **crear sistemas con capacidades** iguales o superiores a las de los seres **humanos**.

Aprendizaje automático o Machine Learning (ML):

Clases:

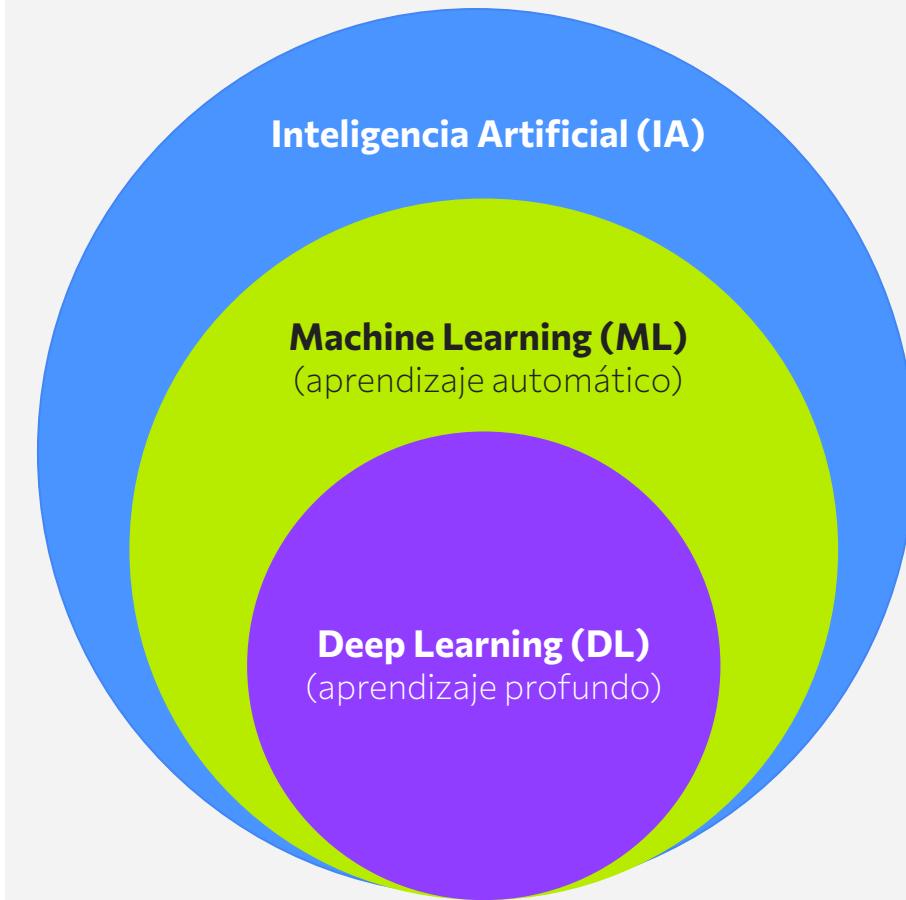
Supervisado, no supervisado, semi-supervisado y de refuerzo.

Ejemplos de Algoritmos:

SVM, KNN, árboles de decisión, **redes neuronales...**

Aprendizaje profundo o Deep Learning (DL):

Subconjunto de algoritmos de ML basados en redes neuronales.



Deep Learning y ANN

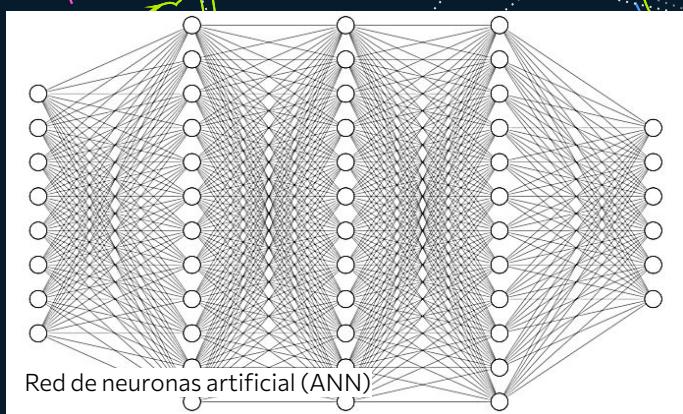
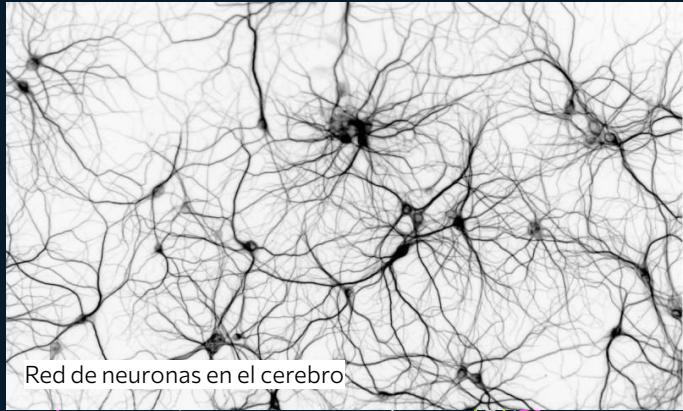
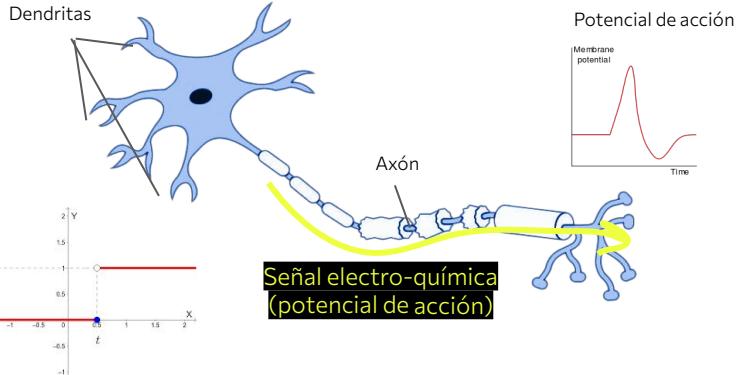
Artificial Neural Networks
=
Redes Neuronales Artificiales

Subconjunto de Machine Learning basado en un algoritmo llamado Red Neuronal.

Qué son las Redes Neuronales?

Algoritmo simplificado que emula el modo en que el cerebro humano procesa la información.

Neurona biológica



Deep Learning y ANN

Componentes de una ANN:

Neurona: unidad básica de procesamiento dentro de una ANN

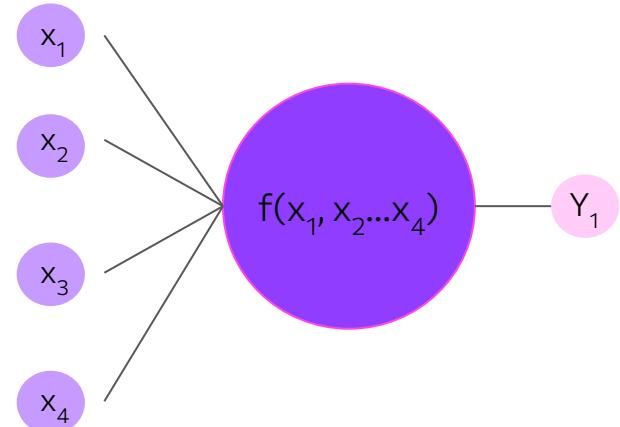
Pesos: fuerzas de conexión entre neuronas, variables.

Sesgo: qué tan predispuesta está la neurona a transmitir información independiente de los pesos.

Función de activación: función que transforma la suma ponderada en el output. Análoga a la tasa de potencial de acción disparando en el cerebro

Organización de las neuronas y entrenamiento.

Neurona artificial o nodo



Output = suma ponderada del input

$$Y_1 = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4$$

Deep Learning y ANN

Componentes de una ANN:

Neurona: unidad básica de procesamiento dentro de una ANN

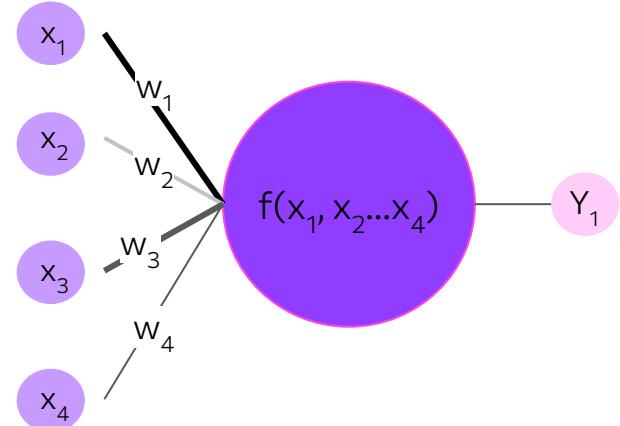
Pesos: fuerzas de conexión entre neuronas, variables.

Sesgo: qué tan predispuesta está la neurona a transmitir información independiente de los pesos.

Función de activación: función que transforma la suma ponderada en el output. Análoga a la tasa de potencial de acción disparando en el cerebro

Organización de las neuronas y entrenamiento.

Neurona artificial o nodo



Output = suma ponderada del input

$$Y_1 = w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3 + w_4 \cdot x_4$$

Deep Learning y ANN

Componentes de una ANN:

Neurona: unidad básica de procesamiento dentro de una ANN

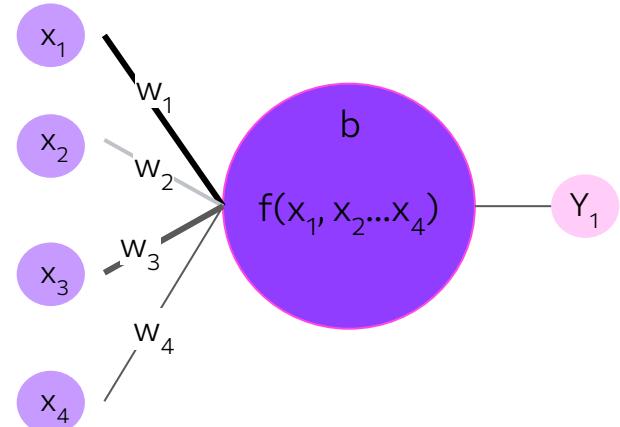
Pesos: fuerzas de conexión entre neuronas, variables.

Sesgo: qué tan predispuesta está la neurona a transmitir información independiente de los pesos.

Función de activación: función que transforma la suma ponderada en el output. Análoga a la tasa de potencial de acción disparando en el cerebro

Organización de las neuronas y entrenamiento.

Neurona artificial o nodo



Output = suma ponderada del input

$$Y_1 = w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3 + w_4 \cdot x_4 + b$$

Deep Learning y ANN

Componentes de una ANN:

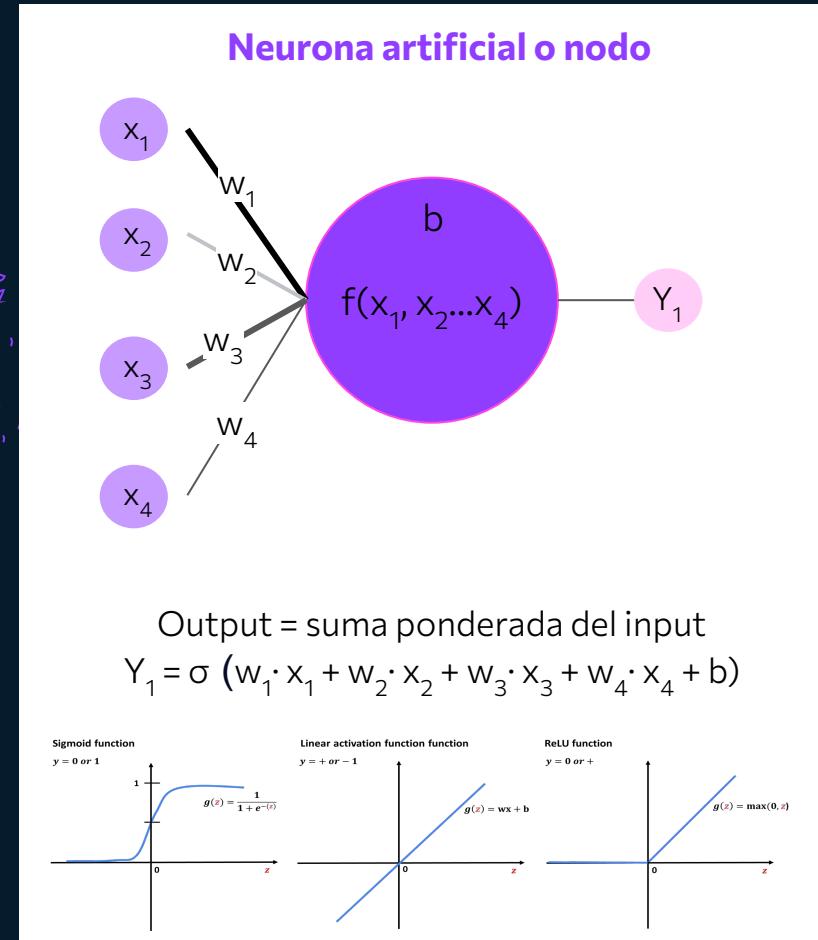
Neurona: unidad básica de procesamiento dentro de una ANN

Pesos: fuerzas de conexión entre neuronas, variables.

Sesgo: qué tan predispuesta está la neurona a transmitir información independiente de los pesos.

Función de activación: función que transforma la suma ponderada en el output. Análoga a la tasa de potencial de acción disparando en el cerebro

Organización de las neuronas y entrenamiento.



Deep Learning y ANN

Componentes de una ANN:

Neurona: unidad básica de procesamiento dentro de una ANN

Pesos: fuerzas de conexión entre neuronas, variables.

Sesgo: qué tan predispuesta está la neurona a transmitir información independiente de los pesos.

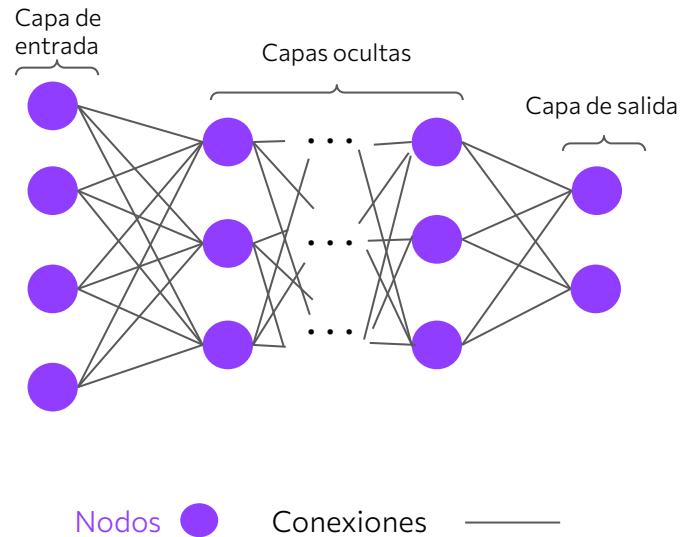
Función de activación: función que transforma la suma ponderada en el output. Análoga a la tasa de potencial de acción disparando en el cerebro

Organización de las neuronas y entrenamiento.

Arquitectura

Selección de los pesos y sesgos más adecuados

Red Neuronal Artificial Profunda



¿Cómo obtenemos nuestro modelo?

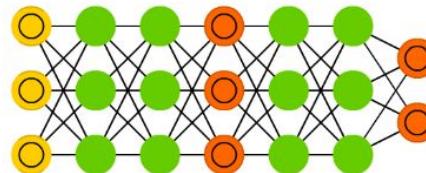
Paso número 1: elegir la arquitectura

- Número de capas y neuronas por capa.
- Cómo conectamos las neuronas.
- Tipo de función de activación por capa.

Paso número 2: Configuración de hiper-parámetros y entrenamiento.

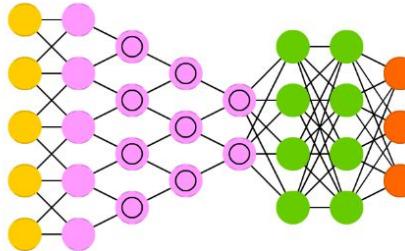
- Seleccionar el algoritmo de entrenamiento.
- Entrenar la red neuronal.

Generative Adversarial Network (GAN)

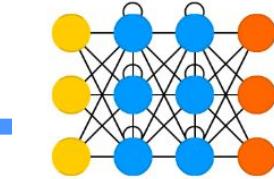


Long / Short Term Memory (LSTM)

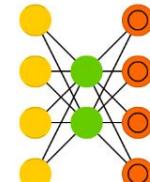
Deep Convolutional Network (DCN)



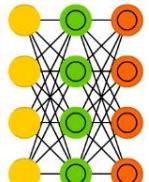
Recurrent Neural Network (RNN)



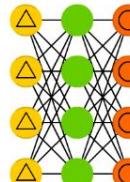
Auto Encoder (AE)



Variational AE (VAE)



Denoising AE (DAE)





LoboaTeresa/Workshop-COF-23

Práctica #1

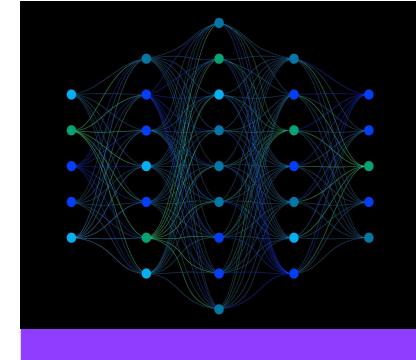
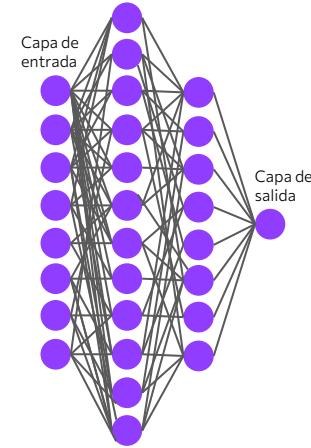
*Construcción de una red neuronal
para la predicción de diabetes*

SNGULAR

Entrenamiento de un modelo de predicción de diabetes



Age	BS Fast	BS pp	Plasma R	Plasma F	Hb1Ac	Type
22	6.8	8.8	11.2	7.2	62	Type 1
24	19	6.3	7.9	3.9	40	Normal
41	6.3	4.2	12.2	7.8	57	Type 2
44	6.8	8.2	11.6	7.4	69	Type 2
22	31	6.3	7.9	3.9	40	Normal
60	6.7	8.7	11.6	7.4	69	Type1
67	6.8	4.8	13.1	9.1	58	Type2
23	28	7.7	11.0	6.1	36	Normal
35	6.7	8.7	11.6	7.4	69	Type1
55	5.2	6.8	10.9	4.2	33	Normal
26	6.8	8.2	11.6	7.4	69	Type2
34	5.8	4.2	11.4	8.4	53	Type2
27	30	7.7	11.0	6.1	36	Normal
26	46	5.6	10.2	5.4	32	Normal



Herramientas de trabajo

[Keras](#), python y google colab.

Dataset

[Pima Indians Diabetes Database](#). 8 parámetros + diagnosis (labels/ground truth).

Arquitectura

4 Capas completamente conectadas (fully connected).

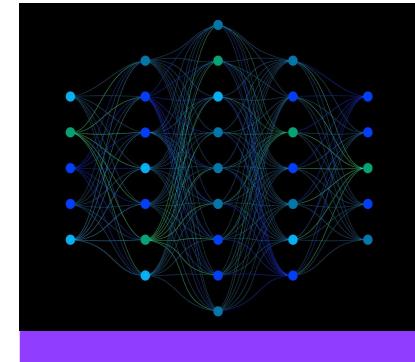
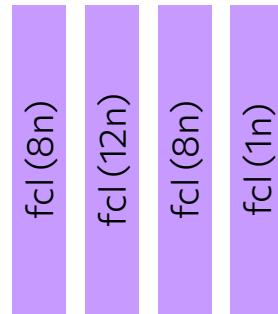
Algoritmo de entrenamiento

Épocas, optimizador, función de pérdida, batch size.

Entrenamiento de un modelo de predicción de diabetes



Age	BS Fast	BS pp	Plasma R	Plasma F	Hb1Ac	Type
22	6.8	8.8	11.2	7.2	62	Type 1
24	19	6.3	7.9	3.9	40	Normal
41	6.3	4.2	12.2	7.8	57	Type 2
44	6.8	8.2	11.6	7.4	69	Type 2
22	31	6.3	7.9	3.9	40	Normal
60	6.7	8.7	11.6	7.4	69	Type1
67	6.8	4.8	13.1	9.1	58	Type2
23	28	7.7	11.0	6.1	36	Normal
35	6.7	8.7	11.6	7.4	69	Type1
55	5.2	6.8	10.9	4.2	33	Normal
26	6.8	8.2	11.6	7.4	69	Type2
34	5.8	4.2	11.4	8.4	53	Type2
27	30	7.7	11.0	6.1	36	Normal
26	46	5.6	10.2	5.4	32	Normal



Herramientas de trabajo

[Keras](#), python y google colab.

Dataset

[Pima Indians Diabetes Database](#). 8 parámetros + diagnosis (labels/ground truth).

Arquitectura

4 Capas completamente conectadas (fully connected).

Algoritmo de entrenamiento

Épocas, optimizador, función de pérdida, batch size.

¿Cómo obtenemos nuestro modelo?

Paso número 1: elegir la arquitectura

- Número de capas y neuronas por capa.
- Cómo conectamos las neuronas.
- Tipo de función de activación por capa.

Paso número 2: Configuración de hiper-parámetros y entrenamiento.

- Seleccionar el algoritmo de entrenamiento.
- Entrenar la red neuronal.

- Número de épocas
- Batch size
- Optimizador
- Función de pérdida o loss function

Paso número 2:

Definición de hiper-parámetros y entrenamiento

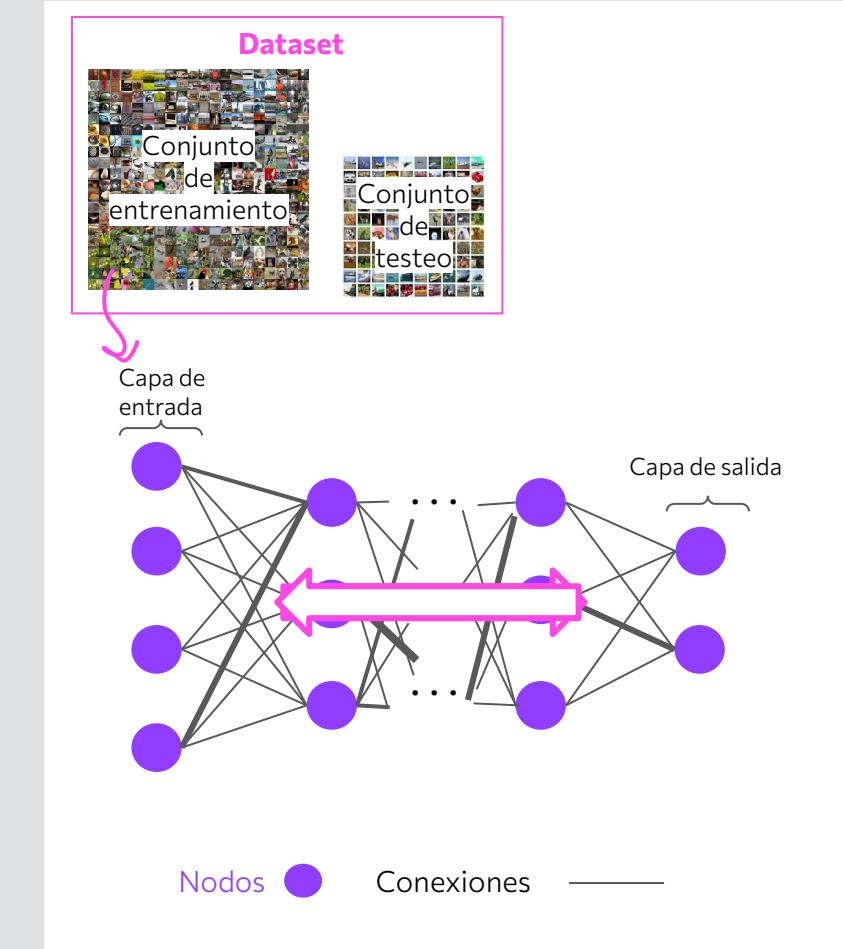
El **entrenamiento** del modelo es un proceso iterativo de **ajuste de pesos y sesgos** de todas las neuronas, para que la salida de la ANN se ajuste lo más posible a nuestro objetivo.

Épocas: número de iteraciones.

Batches: No se suele procesar todo el dataset de golpe. División en n subconjuntos o batches.

Optimizador: método para ajustar los pesos. Ej: Adam, SGD.

Función de pérdida: nos indica cuánto nos hemos equivocado con nuestras predicciones. Ej: Binary Cross-Entropy



Paso número 2:

Definición de hiper-parámetros y entrenamiento

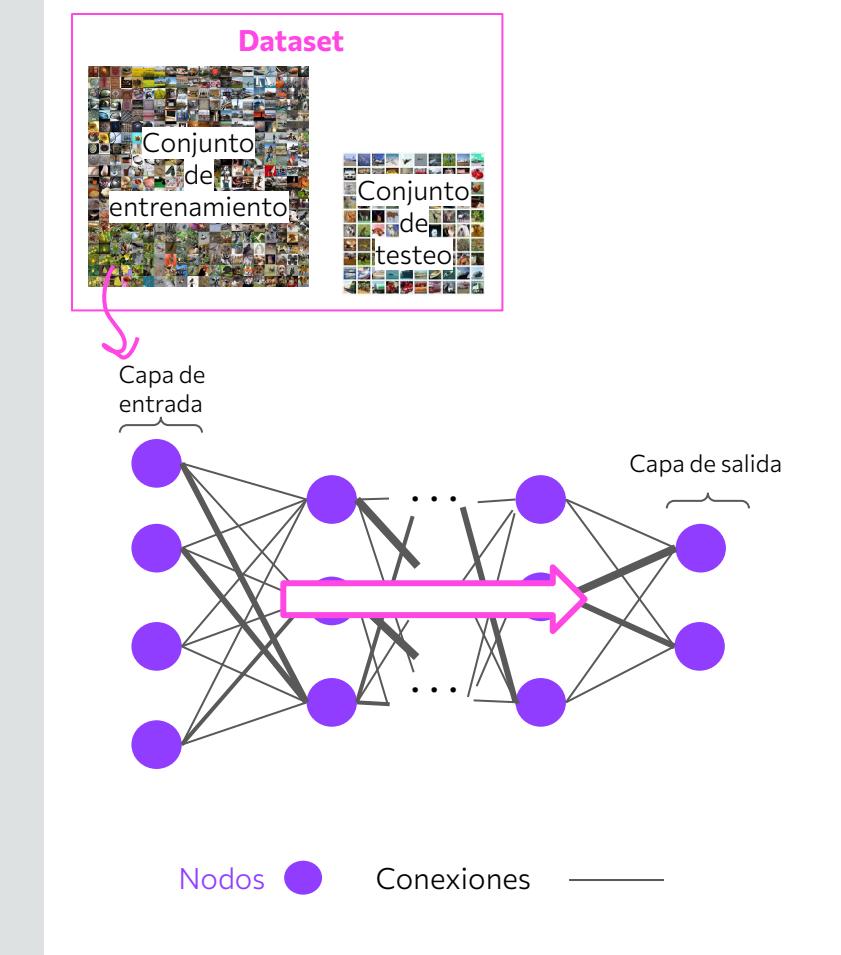
El **entrenamiento** del modelo es un **proceso iterativo** de **ajuste de pesos y sesgos** de todas las neuronas, para que la salida de la ANN se ajuste lo más posible a nuestro objetivo.

Épocas: número de iteraciones.

Batches: No se suele procesar todo el dataset de golpe. División en n subconjuntos o batches.

Optimizador: método para ajustar los pesos. Ej: Adam, SGD.

Función de pérdida: nos indica cuánto nos hemos equivocado con nuestras predicciones. Ej: Binary Cross-Entropy



Épocas: número de veces que pasamos TODO el dataset a través del modelo durante el entrenamiento.

Conjunto de entrenamiento:
1000 pacientes

TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED	NEIGHBOUR	X	Y
Other Thef	2003	5	12	16	15	99X TERM	Strathcon	493906.5	5457452
Other Thef	2003	5	7	15	20	99X TERM	Strathcon	493906.5	5457452
Other Thef	2003	4	23	16	40	99X TERM	Strathcon	493906.5	5457452
Other Thef	2003	4	20	11	15	99X TERM	Strathcon	493906.5	5457452
Other Thef	2003	4	12	17	45	99X TERM	Strathcon	493906.5	5457452
Other Thef	2003	3	26	20	45	99X TERM	Strathcon	493906.5	5457452
Break and	2003	3	10	12	0	63XX WILT	Kerrisdale	489325.6	5452818
Mischief	2003	6	28	4	13	40XX W 19	Dunbar-So	485903.1	5455884
Other Thef	2003	2	16	9	2	99X TERM	Strathcon	493906.5	5457452
Break and	2003	7	9	18	15	18XX E 3RI	Grandview	495078.2	5457221
Other Thef	2003	1	31	19	45	99X TERM	Strathcon	493906.5	5457452
Mischief	2003	9	27	1	0	40XX W 21	Dunbar-So	485853	5455684
Break and	2003	4	19	18	0	18XX E 3RI	Grandview	495093.7	5457230
Break and	2003	9	24	18	30	18XX E 3RI	Grandview	495103.8	5457221
Break and	2003	11	5	8	12	63XX WINI	Sunset	493790.5	5452631
Break and	2003	9	26	2	30	10XX ALBE	West End	491067.7	5459114
Break and	2003	10	21	10	0	18XX E 3RI	Grandview	495119.3	5457230
Other Thef	2003	1	25	12	30	99X TERM	Strathcon	493906.5	5457452
Offence Ag	2003	2	12			OFFSET TO PROTECT		0	0
Other Thef	2003	1	9	6	45	99X TERM	Strathcon	493906.5	5457452
Other Thef	2003	4	30	13	6	99X SEYMI	Central Bu	491205.2	5458520
Other Thef	2003	12	12	15	50	99X SEYMI	Central Bu	491143.3	5458446
Other Thef	2003	3	7	16	15	99X ROBSI	Central Bu	491132.2	5458889
Offence Ag	2003	4	4			OFFSET TO PROTECT		0	0

Capa de
entrada

Capa de salida

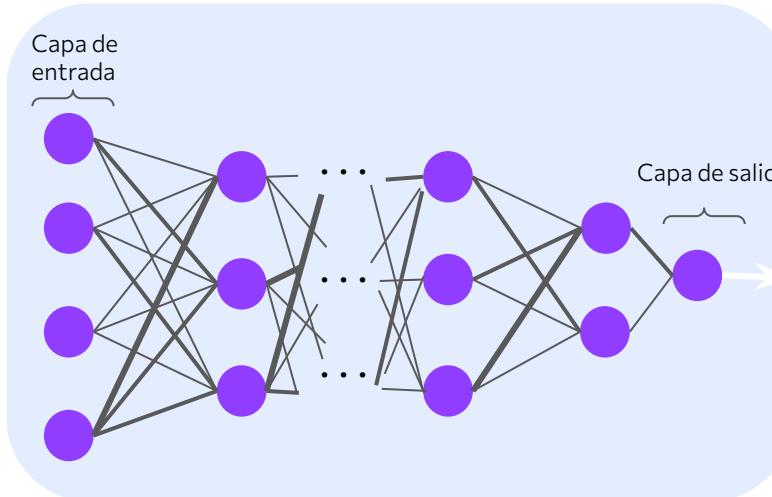
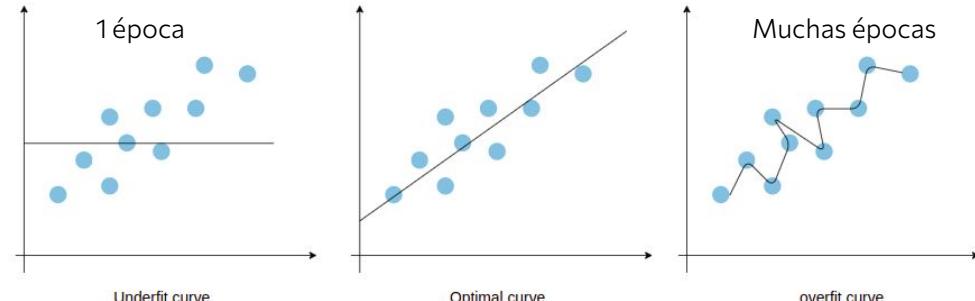
Labels o etiquetas:
"diabético" o "sano"

Función de pérdida
"diabético"

Épocas: número de veces que pasamos TODO el dataset a través del modelo durante el entrenamiento.

Conjunto de entrenamiento:
1000 pacientes

TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED	NEIGHBOUR	X	Y
Other Thef	2003	5	12	16	15	99X TERM	Strathcona	493906.5	5457452
Other Thef	2003	5	7	15	20	99X TERM	Strathcona	493906.5	5457452
Other Thef	2003	4	23	16	40	99X TERM	Strathcona	493906.5	5457452
Other Thef	2003	4	20	11	15	99X TERM	Strathcona	493906.5	5457452
Other Thef	2003	4	12	17	45	99X TERM	Strathcona	493906.5	5457452
Other Thef	2003	3	26	20	45	99X TERM	Strathcona	493906.5	5457452
Break and	2003	3	10	12	0	63XX WILT	Kerrisdale	489325.6	5452818
Mischief	2003	6	28	4	13	40XX W 19	Dunbar-So	485903.1	5455884
Other Thef	2003	2	16	9	2	99X TERM	Strathcona	493906.5	5457452
Break and	2003	7	9	18	15	18XX E 3RI	Grandview	495078.2	5457221
Other Thef	2003	1	31	19	45	99X TERM	Strathcona	493906.5	5457452
Mischief	2003	9	27	1	0	40XX W 21	Dunbar-So	485853	5455684
Break and	2003	4	19	18	0	18XX E 3RI	Grandview	495093.7	5457230
Break and	2003	9	24	18	30	18XX E 3RI	Grandview	495103.8	5457221
Break and	2003	11	5	8	12	63XX WIN	Sunset	493790.5	5452631
Break and	2003	9	26	2	30	10XX ALBE	West End	491067.7	5459114
Break and	2003	10	21	10	0	18XX E 3RI	Grandview	495119.3	5457230
Other Thef	2003	1	25	12	30	99X TERM	Strathcona	493906.5	5457452
Offence Ag	2003	2	12			OFFSET TO PROTECT		0	0
Other Thef	2003	1	9	6	45	99X TERM	Strathcona	493906.5	5457452
Other Thef	2003	4	30	13	6	99X SEYMI	Central Bu	491205.2	5458520
Other Thef	2003	12	12	15	50	99X SEYMI	Central Bu	491143.3	5458446
Other Thef	2003	3	7	16	15	99X ROBSI	Central Bu	491132.2	5458889
Offence Ag	2003	4	4			OFFSET TO PROTECT		0	0



Labels o etiquetas:
“diabético” o “sano”

Función de pérdida

Batch: No se suele procesar todo el dataset de golpe.

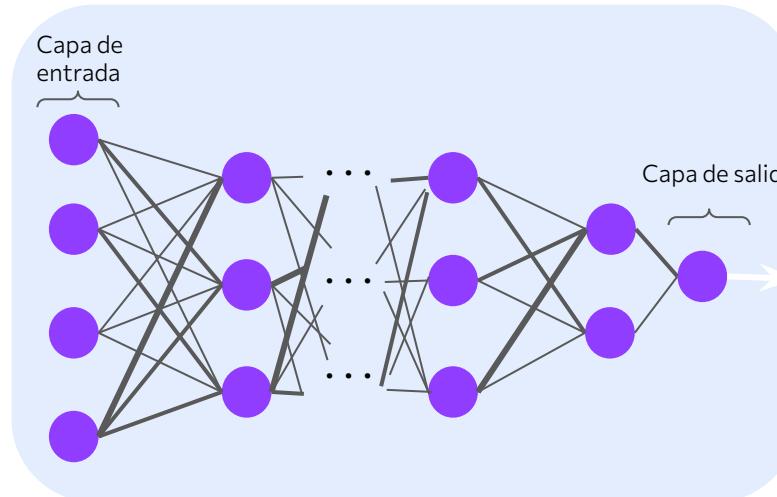
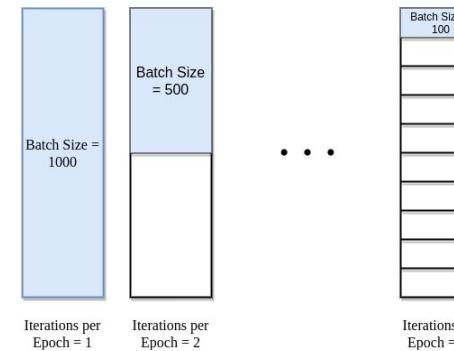
División en n subconjuntos o batches.

Conjunto de entrenamiento:

1000 pacientes

5 batches de 200 pacientes

TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED	NEIGHBOUR	Y
Other Thef	2003	5	12	16	15	99X TERM	Strathcon	493906.5 5457452
Other Thef	2003	5	7	15	20	99X TERM	Strathcon	493906.5 5457452
Other Thef	2003	4	23	16	40	99X TERM	Strathcon	493906.5 5457452
Other Thef	2003	4	20	11	15	99X TERM	Strathcon	493906.5 5457452
Other Thef	2003	4	12	17	45	99X TERM	Strathcon	493906.5 5457452
Other Thef	2003	3	26	20	45	99X TERM	Strathcon	493906.5 5457452
Break and	2003	3	10	12	0	63XX WILT	Kerrisdale	489325.6 5452818
Mischief	2003	6	28	4	13	40XX W 15 Dunbar-So	485903.1	5455884
Other Thef	2005	2	16	9	2	99X TERM	Strathcon	493906.5 5457452
Break and	2003	7	9	18	15	18XX E 3RI	Grandview	495078.2 5457221
Other Thef	2003	1	31	19	45	99X TERM	Strathcon	493906.5 5457452
Mischief	2003	9	27	1	0	40XX W 21 Dunbar-So	485853	5455684
Break and	2003	4	19	18	0	18XX E 3RI	Grandview	495093.7 5457230
Break and	2003	9	24	18	30	18XX E 3RI	Grandview	495103.8 5457221
Break and	2003	11	5	8	12	63XX WIN	Sunset	495790.5 5452631
Break and	2003	9	26	2	30	10XX ALBE	West End	491067.7 5459114
Break and	2003	10	21	10	0	18XX E 3RI	Grandview	495119.3 5457230
Other Thef	2003	1	25	12	30	99X TERM	Strathcon	493906.5 5457452
Offence Ag	2003	2	12			OFFSET TO PROTECT		0 0
Other Thef	2003	1	9	6	45	99X TERM	Strathcon	493906.5 5457452
Other Thef	2003	4	30	13	6	99X SEYMI	Central Bu	491205.2 5458520
Other Thef	2003	12	12	15	50	99X SEYMI	Central Bu	491143.3 5458446
Other Thef	2003	3	7	16	15	99X ROBSI	Central Bu	491132.2 5458889
Offence Ag	2003	4	4			OFFSET TO PROTECT		0 0



Labels o etiquetas:
"diabético" o "sano"

Función de pérdida

"diabético"

Un batch ~ dataset puede llenar toda la memoria de nuestro ordenador. Además dificulta la generalización del modelo.
Convergencia lenta.

batches pequeños: convergencia rápida pero ruidosa.

Recomendable empezar un un batch grande

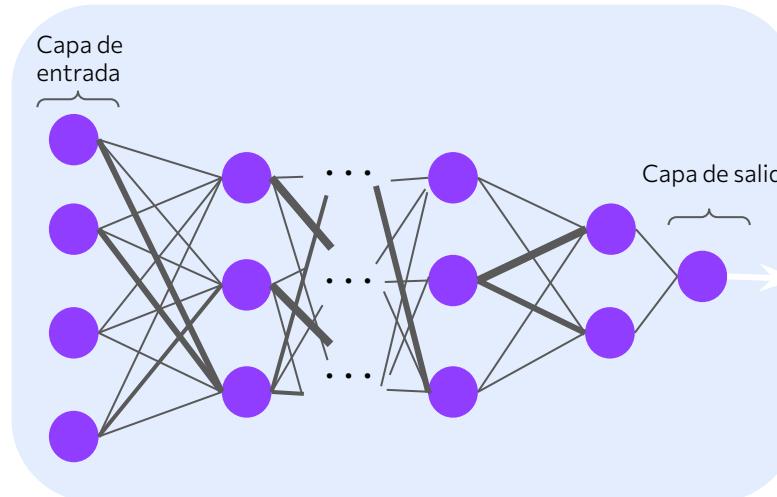
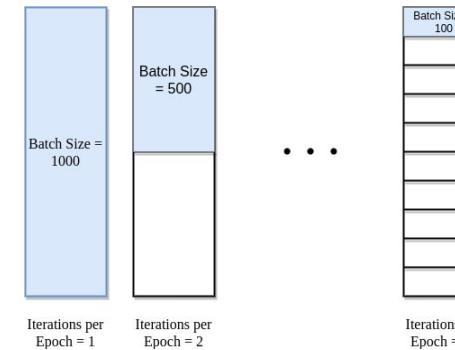
Batch: No se suele procesar todo el dataset en cada época. División en n subconjuntos o batches.

Conjunto de entrenamiento:

1000 pacientes

5 batches de 200 pacientes

TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED	NEIGHBOUR	Y
Other Thef	2003	5	12	16	15	99X TERM	Strathcon	493906.5
Other Thef	2003	5	7	15	20	99X TERM	Strathcon	493906.5
Other Thef	2003	4	23	16	40	99X TERM	Strathcon	493906.5
Other Thef	2003	4	20	11	15	99X TERM	Strathcon	493906.5
Other Thef	2003	4	12	17	45	99X TERM	Strathcon	493906.5
Other Thef	2003	3	26	20	45	99X TERM	Strathcon	493906.5
Break and	2003	3	10	12	0	63XX WLT	Kerrisdale	489325.6
Mischief	2003	6	28	4	13	40XX W 15 Dunbar-So	485903.1	5455884
Other Thef	2005	2	16	9	2	99X TERM	Strathcon	493906.5
Break and	2003	7	9	18	15	18XX E 3RI	Grandview	495078.2
Other Thef	2003	1	31	19	45	99X TERM	Strathcon	493906.5
Mischief	2003	9	27	1	0	40XX W 21 Dunbar-So	485853	5455684
Break and	2003	4	19	18	0	18XX E 3RI	Grandview	495093.7
Break and	2003	9	24	18	30	18XX E 3RI	Grandview	495103.8
Break and	2003	11	5	8	12	63XX WIN	Sunset	493790.5
Break and	2003	9	26	2	30	10XX ALBE	West End	491067.7
Break and	2003	10	21	10	0	18XX E 3RI	Grandview	495119.3
Other Thef	2003	1	25	12	30	99X TERM	Strathcon	493906.5
Offence Ag	2003	2	12			OFFSET TO PROTECT		0
Other Thef	2003	1	9	6	45	99X TERM	Strathcon	493906.5
Other Thef	2003	4	30	13	6	99X SEYMI	Central Bu	491205.2
Other Thef	2003	12	12	15	50	99X SEYMI	Central Bu	491143.3
Other Thef	2003	3	7	16	15	99X ROBSI	Central Bu	491132.2
Offence Ag	2003	4	4			OFFSET TO PROTECT		0



Labels o etiquetas:
"diabético" o "sano"

Función de pérdida

"diabético"

Un batch ~ dataset puede llenar toda la memoria de nuestro ordenador. Además dificulta la generalización del modelo.
Convergencia lenta.

batches pequeños: convergencia rápida pero ruidosa.

Recomendable empezar un un batch grande

Paso número 2:

Definición de hiper-parámetros y entrenamiento

El **entrenamiento** del modelo es un **proceso iterativo** de **ajuste de pesos y sesgos** de todas las neuronas, para que la salida de la ANN se ajuste lo más posible a nuestro objetivo.

Épocas: número de iteraciones.

Batches: No se suele procesar todo el dataset en cada época. División en n subconjuntos o batches.

Optimizador: método para ajustar los pesos. Ej: Adam, SGD.

Función de pérdida: nos indica cuánto nos hemos equivocado con nuestras predicciones. Ej: Binary Cross-Entropy

```
# Build the model
model = Sequential()
model.add(Dense(12, input_dim=8,
               init='uniform', activation='relu'))
model.add(Dense(8, init='uniform',
               activation='relu'))
model.add(Dense(1, init='uniform',
               activation='sigmoid'))
```

```
# Compile the model
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```

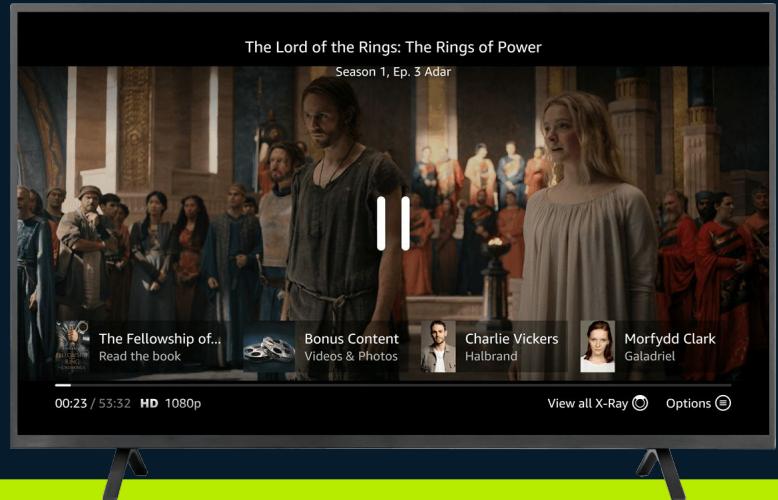
```
# Train the model
model.fit(X, Y, nb_epoch=150,
          batch_size=10)
```

Snippet de la práctica #1

Práctica #2

*Entrenamiento de una red
neuronal para la clasificación de
caras de actores.*

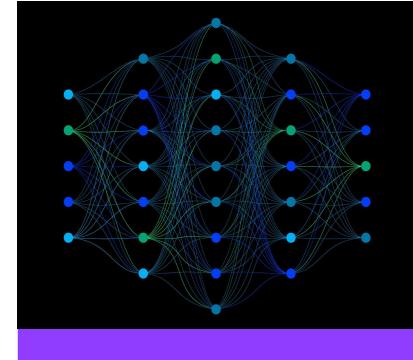
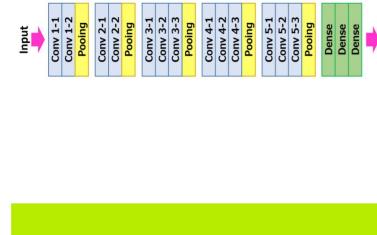
SNGULAR



Entrenamiento de un modelo de predicción de diabetes



Age	BS Fast	BS pp	Plasma R	Plasma F	Hb1Ac	Type
22	6.8	8.8	11.2	7.2	62	Type 1
24	19	6.3	7.9	3.9	40	Normal
41	6.3	4.2	12.2	7.8	57	Type 2
44	6.8	8.2	11.6	7.4	69	Type 2
22	31	6.3	7.9	3.9	40	Normal
60	6.7	8.7	11.6	7.4	69	Type 1
67	6.8	4.8	13.1	9.1	58	Type 2
23	28	7.7	11.0	6.1	36	Normal
35	6.7	8.7	11.6	7.4	69	Type 1
55	5.2	6.8	10.9	4.2	33	Normal
26	6.8	8.2	11.6	7.4	69	Type 2
34	5.8	4.2	11.4	8.4	53	Type 2
27	30	7.7	11.0	6.1	36	Normal
26	46	5.6	10.2	5.4	32	Normal



Herramientas de trabajo

[Keras](#), python y google colab.

Dataset

[Celebrity Face Image Dataset](#).

17 actores y actrices, 100 imágenes de cada uno.

Arquitectura

[VGG16](#)

Algoritmo de entrenamiento

Épocas, optimizador, función de pérdida, batch size.

Core layers

- Input object
- Dense layer
- Activation layer
- Embedding layer

Convolution layers

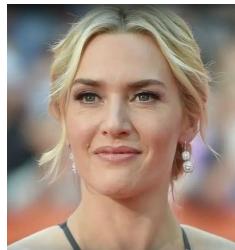
- Conv1D layer
- Conv2D layer
- Conv3D layer
- SeparableConv1D layer
- SeparableConv2D layer
- DepthwiseConv2D layer
- Conv1DTranspose layer
- Conv2DTranspose layer
- Conv3DTranspose layer

Pooling layers

- MaxPooling1D layer
- MaxPooling2D layer
- MaxPooling3D layer
- AveragePooling1D layer
- AveragePooling2D layer
- AveragePooling3D layer
- GlobalMaxPooling1D layer

Reshaping layers

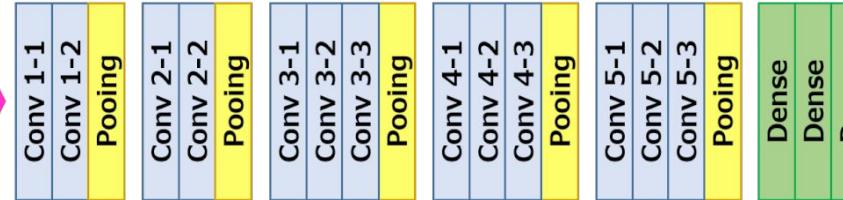
- Reshape layer
- Flatten layer



Construcción de la red neuronal

arquitectura: VGG16

Input



Output

"kate
winslet"

Antes (1D):

(3,
4,
6,
2.4,
4,
0.3,
5.5
32)



Ahora (2D):

0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0
0	0	4	60	157	236	255	255	177	95	61	32	0	0	29	
0	10	16	119	238	255	244	245	243	250	249	255	222	103	0	
0	14	170	255	255	244	254	253	245	255	249	253	251	124	1	
2	98	255	228	255	251	254	211	141	116	122	215	251	238	49	
13	217	243	255	155	33	226	52	2	0	13	332	255	255	36	
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	
61	1245	245	255	212	35	11	9	3	0	116	236	243	255	137	
0	87	252	250	248	215	60	0	1	21	121	252	255	248	144	
0	13	113	255	255	245	255	187	181	248	252	242	208	36	0	
1	0	5	172	251	255	241	255	247	255	241	163	17	0	7	
0	0	0	4	58	251	255	246	254	253	255	250	120	11	0	
0	0	4	97	255	255	255	248	252	255	244	255	182	10	4	
0	22	206	252	246	241	241	100	24	113	255	245	255	194	9	
0	111	255	242	255	255	158	24	0	0	6	39	205	222	230	
0	218	251	250	137	7	11	0	0	2	62	255	250	125	3	
0	173	255	255	255	101	9	20	0	3	13	182	251	245	61	
0	0	107	251	241	255	230	98	55	19	118	217	248	253	42	
0	18	146	250	255	247	255	255	249	255	240	255	129	0	5	
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	
0	0	6	1	0	52	133	233	255	252	147	37	0	4	1	
0	0	5	5	0	0	0	0	0	14	1	0	6	6	0	

Core layers

- Input object
- Dense layer
- Activation layer
- Embedding layer

Convolution layers

- Conv1D layer
- Conv2D layer
- Conv3D layer
- SeparableConv1D layer
- SeparableConv2D layer
- DepthwiseConv2D layer
- Conv1DTranspose layer
- Conv2DTranspose layer
- Conv3DTranspose layer

Pooling layers

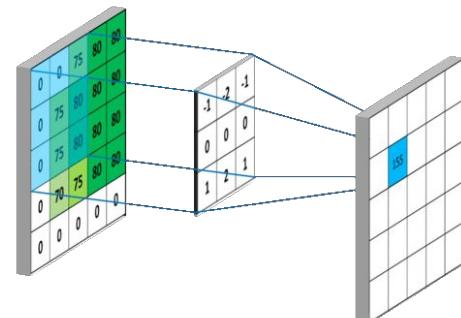
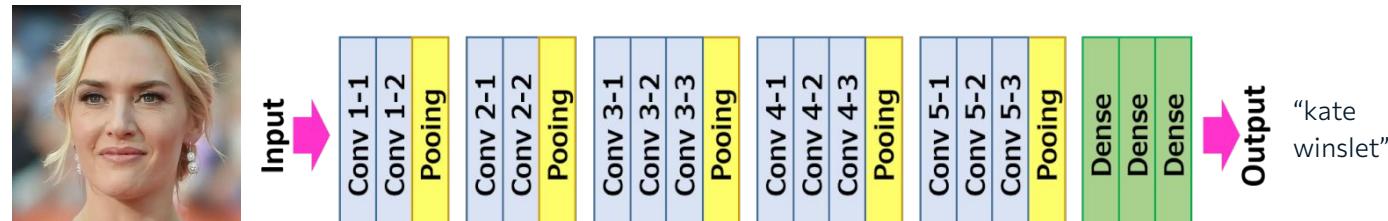
- MaxPooling1D layer
- MaxPooling2D layer
- MaxPooling3D layer
- AveragePooling1D layer
- AveragePooling2D layer
- AveragePooling3D layer
- GlobalMaxPooling1D layer

Reshaping layers

- Reshape layer
- Flatten layer

Construcción de la red neuronal

arquitectura: VGG16



Capa convolucional:

El flujo de información de una capa a la siguiente se realiza mediante la convolución de filtros sobre la imagen/mapa de características.

Core layers

- Input object
- Dense layer
- Activation layer
- Embedding layer

Convolution layers

- Conv1D layer
- Conv2D layer
- Conv3D layer
- SeparableConv1D layer
- SeparableConv2D layer
- DepthwiseConv2D layer
- Conv1DTranspose layer
- Conv2DTranspose layer
- Conv3DTranspose layer

Pooling layers

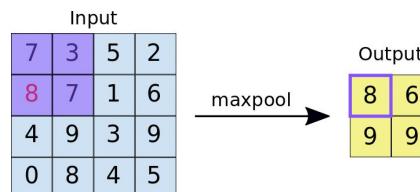
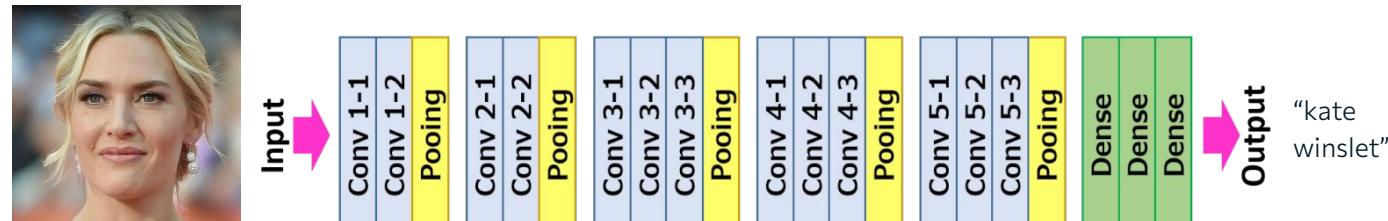
- MaxPooling1D layer
- MaxPooling2D layer
- MaxPooling3D layer
- AveragePooling1D layer
- AveragePooling2D layer
- AveragePooling3D layer
- GlobalMaxPooling1D layer

Reshaping layers

- Reshape layer
- Flatten layer

Construcción de la red neuronal

arquitectura: VGG16

**Capa de pooling:**

Operación que por lo general se aplica entre dos capas de convolución para reducir el tamaño de las imágenes/mapas de características.

Core layers

- Input object
- Dense layer
- Activation layer
- Embedding layer

Convolution layers

- Conv1D layer
- Conv2D layer
- Conv3D layer
- SeparableConv1D layer
- SeparableConv2D layer
- DepthwiseConv2D layer
- Conv1DTranspose layer
- Conv2DTranspose layer
- Conv3DTranspose layer

Pooling layers

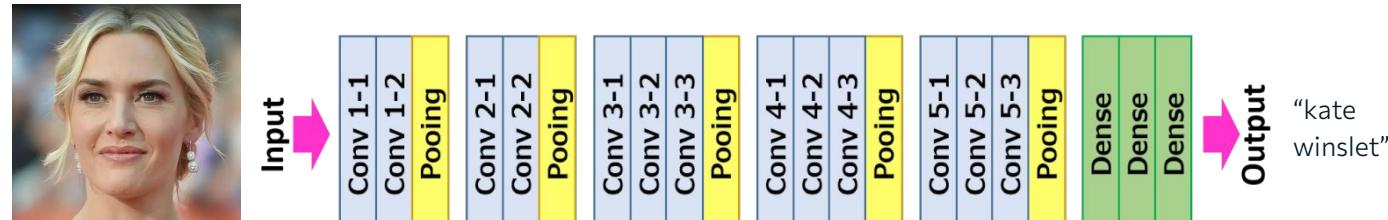
- MaxPooling1D layer
- MaxPooling2D layer
- MaxPooling3D layer
- AveragePooling1D layer
- AveragePooling2D layer
- AveragePooling3D layer
- GlobalMaxPooling1D layer

Reshaping layers

- Reshape layer
- Flatten layer

Construcción de la red neuronal

arquitectura: VGG16



1	1	0
4	2	1
0	2	1

Pooled Feature Map

Flattening

1
1
0
4
2
1
0
2
1

Flattening y capa densa:

Aplana las imágenes 2D en un vector 1D para poder conectar imágenes con capas de neuronas completamente conectadas o densas.



Cómo entrenar a tu modelo de IA

desde otro ~
~ modelo

SNGULAR

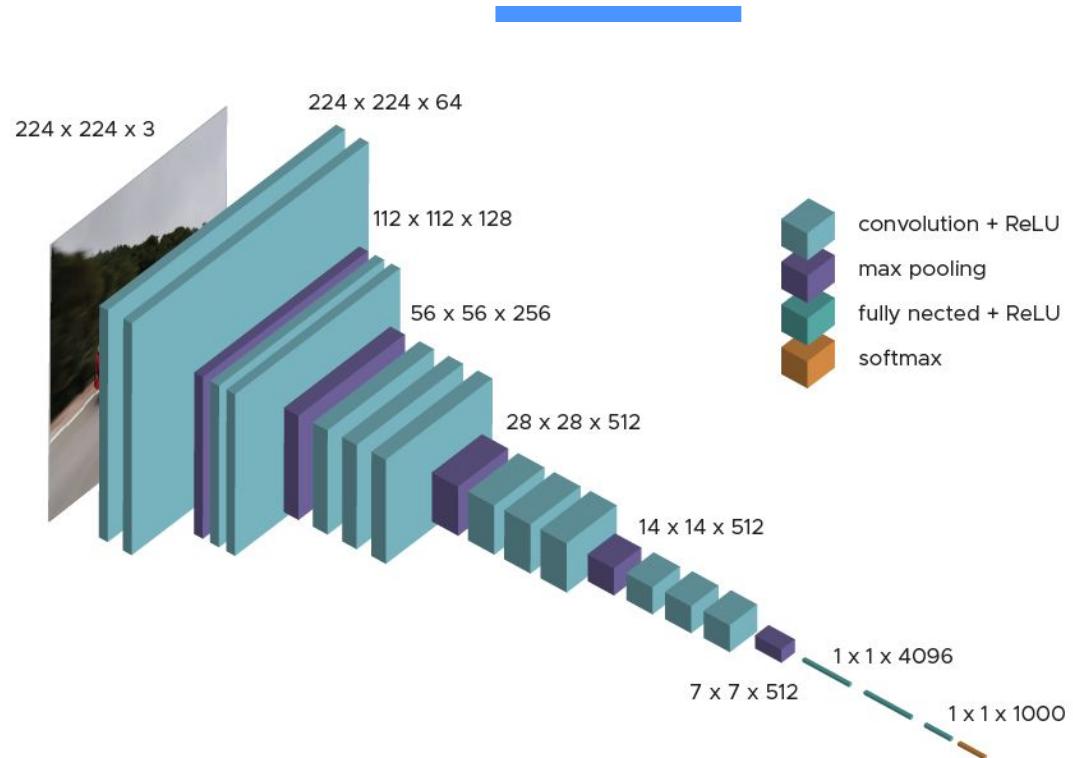


Teresa Lobo Alonso
ML Engineer @ Sngular



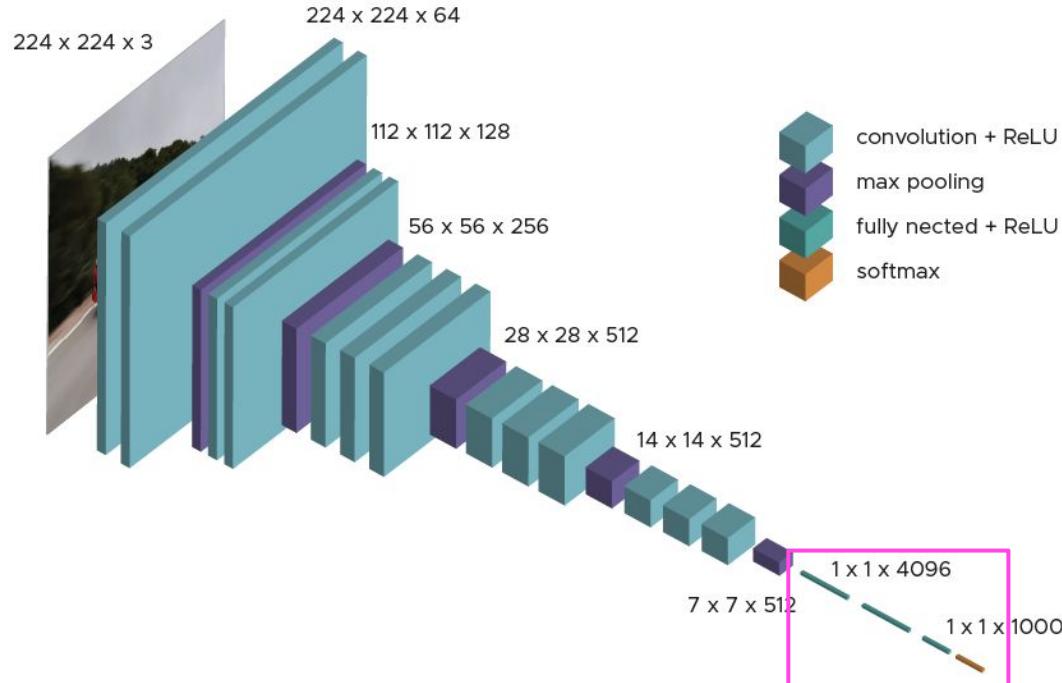
Transfer Learning

¿qué es y por qué es lo mejor que nos podía pasar?



Clases

- Coche
- Tiburón
- Persona
- Tulipán
- Pastor Alemán
- ... (1000 clases)



Clases

- Angelina Jolie
- Will Smith
- Scarlett Johansson
- Leonardo Dicaprio
- Jennifer Lawrence
- ... (10 clases)

Librerías de DL para Python

Tensorflow



Librería de código abierto para Machine Learning.

Desarrollado por Google.

[Introducción a Tensorflow](#)

Pytorch



Framework de de Machine Learning basado en la librería Torch. Código abierto.

Desarrollada por Meta AI.

[Tutorial con pytorch](#).

Keras



Librería de python de código abierto para Deep Learning.

Constituye una capa superior para usar Tensorflow.

Incluye varias redes preentrenadas como VGG, ResNet, Xception, MobileNet...

[Guías de Keras.](#) [Ejemplos prácticos con Keras.](#)

Caffe



Librería de python de código abierto para Deep Learning. Especializada en soluciones de Computer Vision. Código abierto.

Desarrollada por Meta.

[Tutorial](#)

Scikit-Learn



Librería de Python de código abierto para Machine Learning.

Incluye varios algoritmos de clasificación, regresión, random forest, Gradient Boosting, SVM, K-means.

[Introducción a Scikit-Learn.](#) [Ejemplos/tutoriales.](#)

Theano



Librería de Python y un compilador de optimización para manipular y evaluar expresiones matemáticas.

Librerías de DL para Python

OpenMMLab



Librería fundacional de DL.

Pose estimation, OCR, Detección de objetos, LLM,
IA Generativa, tracking, reconocimiento de
acciones...

Ultralytics YOLOv8



YOLOv8 es un grupo de modelos de redes
neuronales convolucionales, creados y entrenados
utilizando el framework PyTorch.

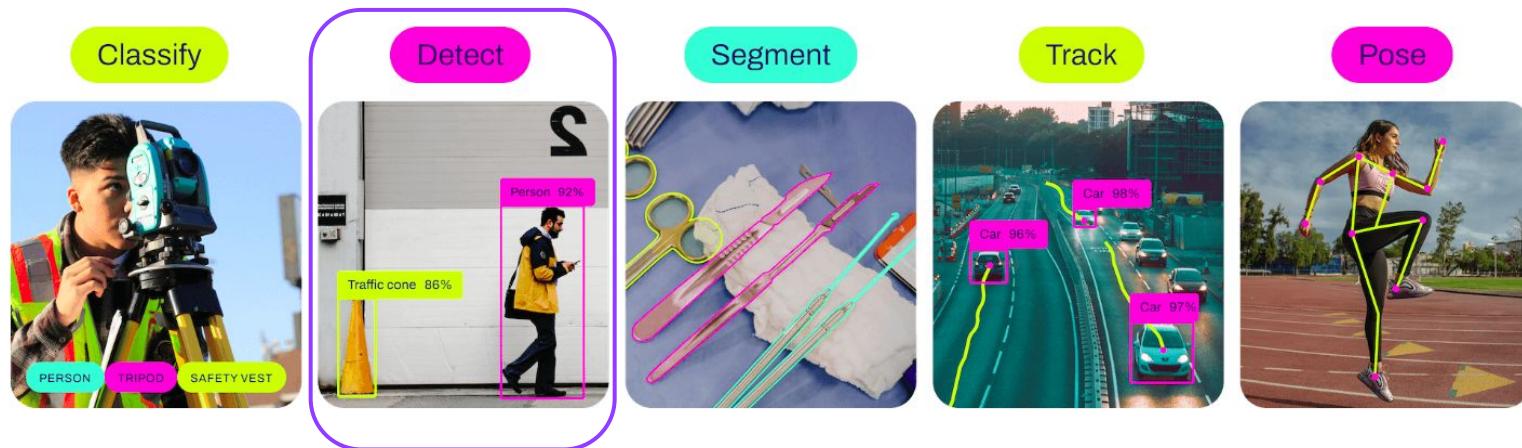
La librería Ultralytics proporciona una única API de
Python para trabajar con todos ellos.

Centrado en Computer Vision

YOLO: Algoritmo de detección de objetos

You Only Look Once

Ultralytics es una librería para hacer uso de un conjunto de modelos llamados YOLOv8 que soportan múltiples tareas.

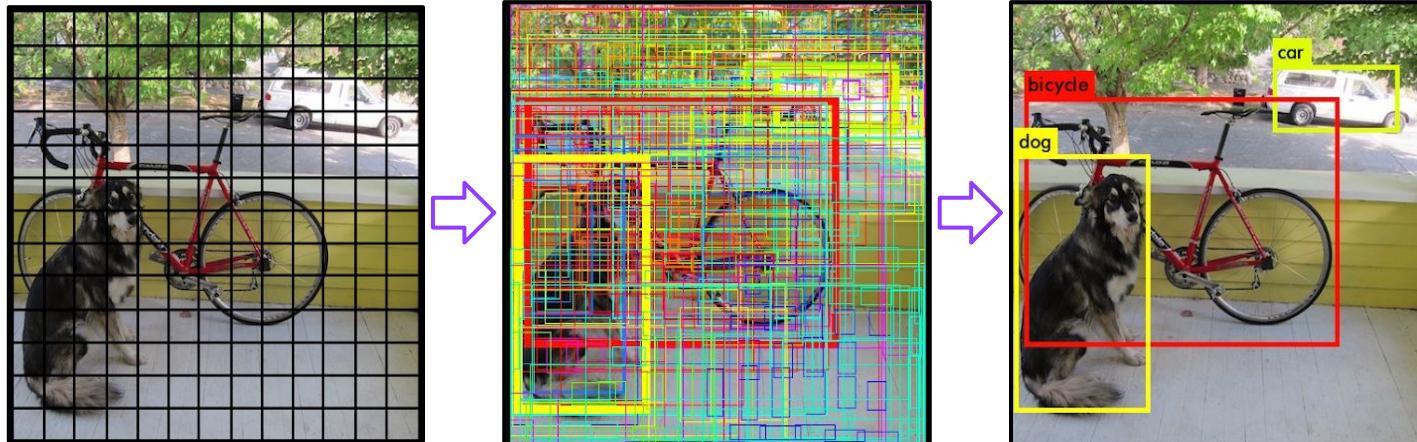


En la práctica número 3 vamos a hacer uso de un modelo pre-entrenado para la detección y vamos a hacer transfer learning para detectar unos objetos muy particulares...

YOLO: Algoritmo de detección de objetos

You Only Look Once

Ultralytics es una librería para hacer uso de un conjunto de modelos llamados YOLOv8 que soportan múltiples tareas.



Práctica #3

*Entrena un modelo para jugar
al Mario party.*

Detección de objetos con YOLOv8

SNGULAR





MARIO PARTY

Entrenaremos un modelo para detectar goombas y bombas

¡¡Este modelo nos hará ganar al Mario Party!!

Goomba 0.98



Bomb 0.89



Cosas a tener en cuenta

Elegid sabiamente vuestros hiper-parámetros. Revisad la [documentación](#) de Ultralytics! ;)

Tamaño del modelo pre-entrenado:

yolov8n.pt, yolov8s.pt, yolov8m.pt, yolov8l.pt

Data augmentation:

Aumentar la variabilidad del dataset.

Número de épocas (iteraciones de entrenamiento):

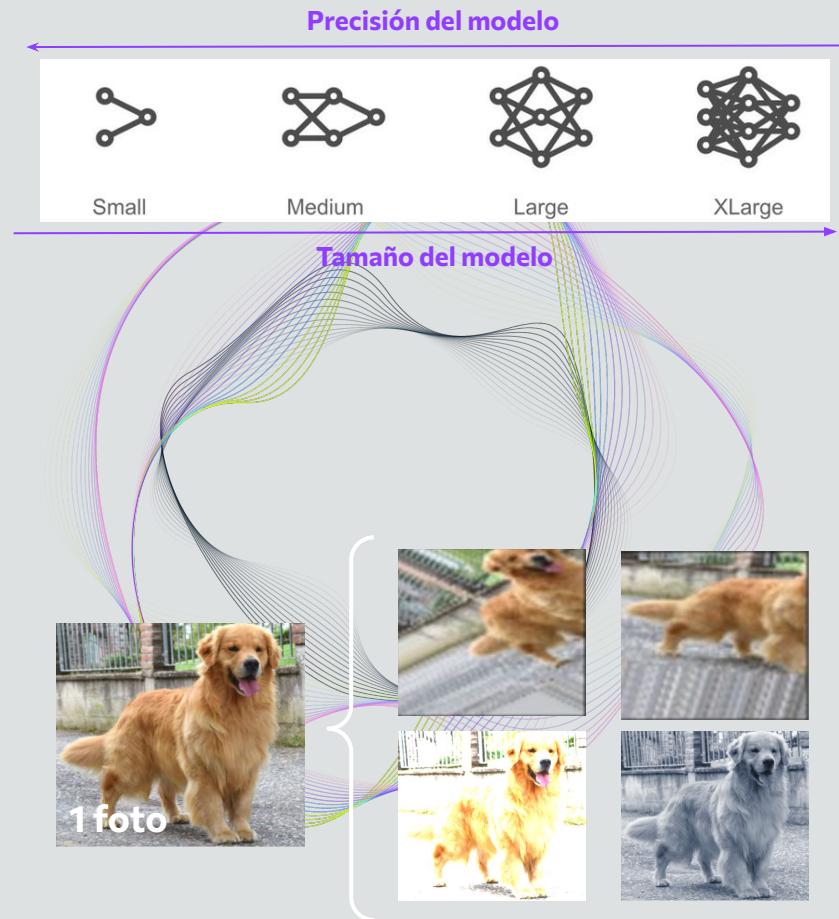
10, 50, 100, 200?

Tamaño del batch:

5, 32, 64?

Tamaño de las imágenes de entrada:

64, 128, 320, 256?





LoboaTeresa



Teresa Lobo

¡Muchas gracias!

*Espero que os animéis a crear
vuestros propios proyectos de DL.*

Mucha suerte en el Hackatón.

SNGULAR



Dame tu opinión :)

Teresa Lobo Alonso
ML Engineer