# Tema 04. Diseño de sistemas de Aprendizaje Automático II

**Autor: Ismael Sagredo Olivenza**

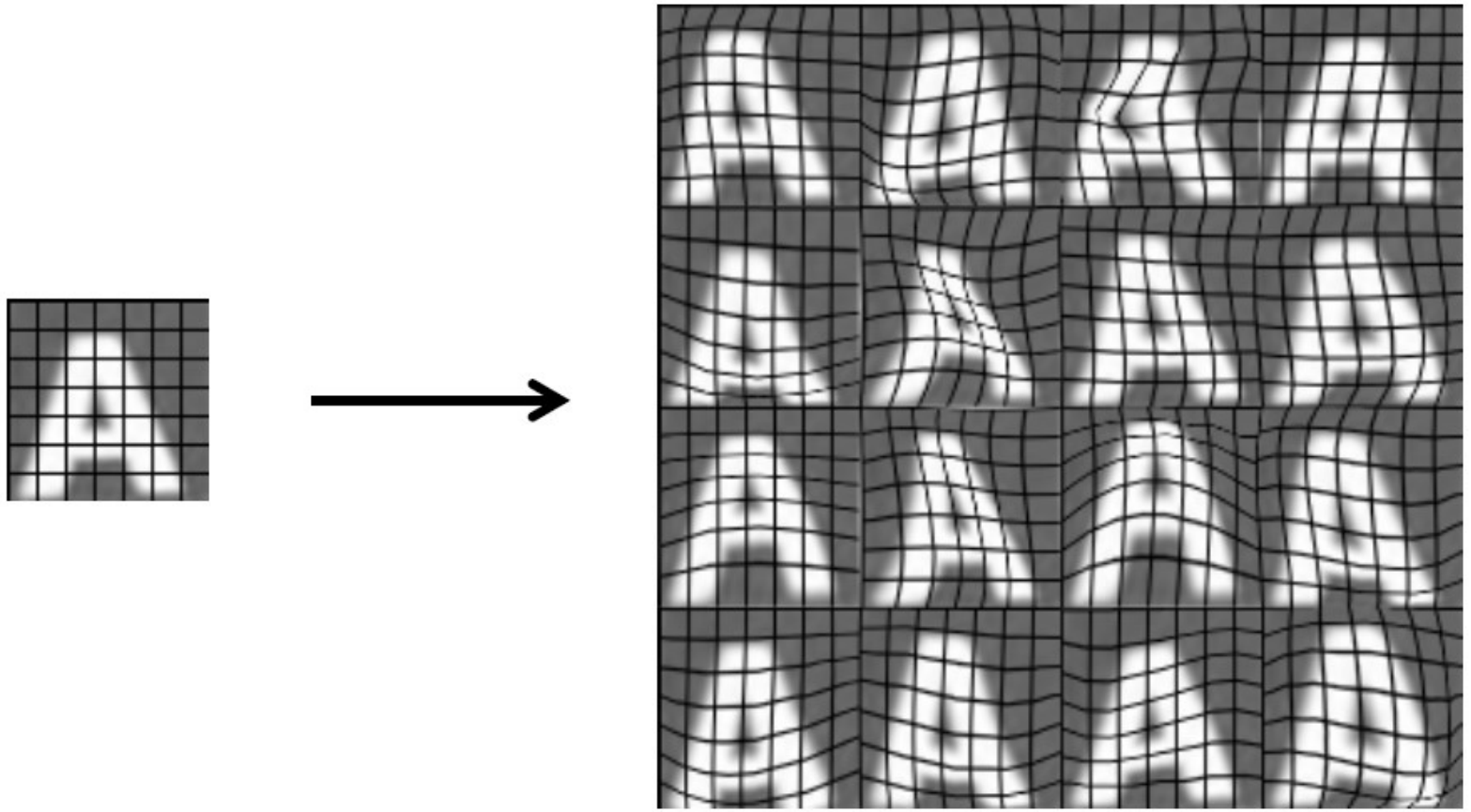
# 4.5 Adding more data

Como ya se ha comentado, añadir más datos de entrenamiento ayuda a reducir la varianza y, por tanto, el error de prueba.

Pero, ¿qué ocurre si no tengo más datos?

Con algunos tipos de datos podemos realizar algunas técnicas que nos ayudan a generar más datos de forma artificial: **Data aumentation**

# 4.5.1 Data Aumentation using distortions



# 4.5.1 Data Aumentation using distortions

Con sonidos, voz, etc., podemos introducir distintos tipos de ruido:

- Ruido de fondo: Maquinaria, Multitud, mala conexión de móvil...

Con imágenes:

- Diferentes tipos de iluminación
- Transformaciones de perspectiva para poder reconocer una imagen transformada

**NOTA**: No suele ser útil añadir ruido aleatorio o sin sentido a los datos.

# Artificial data synthesis for photo OCR

Synthesis: using artificial data inputs to create a new training example.

# 4.5.2 Ejemplo de data aumentation usando keras:

```
import matplotlib.pyplot as plt
import numpy as np
import tensorflow as tf
import tensorflow_datasets as tfds
from tensorflow.keras import layers
# https://www.tensorflow.org/tutorials/images/data_augmentation?hl=es-419
(train_ds, val_ds, test_ds), metadata = tfds.load('tf_flowers',
split=['train[:80%]', 'train[80%:90%]', 'train[90%:]'],with_info=True,as_supervised=True)
num_classes = metadata.features['label'].num_classes
IMG_SIZE = 180

resize_and_rescale = tf.keras.Sequential([
  layers.Resizing(IMG_SIZE, IMG_SIZE),
  layers.Rescaling(1./255)
])
result = resize_and_rescale(image)
_ = plt.imshow(result)
data_augmentation = tf.keras.Sequential([
  layers.RandomFlip("horizontal_and_vertical"),
  layers.RandomRotation(0.2),
])
```

Asegurarse de que disponemos de un clasificador con un sesgo bajo antes de realizar el esfuerzo de buscar más datos. (Trazad curvas de aprendizaje). Por ejemplo, aumenta el número de características/número de unidades ocultas de la red neuronal hasta que conseguir un clasificador con un sesgo bajo.

Una vez que estemos seguros de que necesitamos más datos, podemos abordar este problema de múltiples maneras:

- Síntesis artificial de datos
- Recopilarlos uno mismo
- Crowdsourcing: Amazon Mechanical Turk https://www.mturk.com/

# 4.6 Transfer Learning

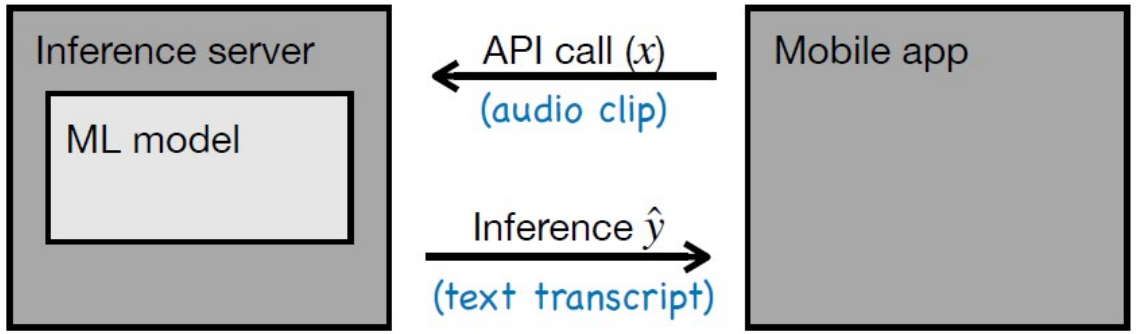
**El Transfer Learning**: "conjunto de métodos que permiten transferir conocimientos adquiridos gracias a la resolución de problemas para resolver otros problemas.”

Las redes de neuronas pueden encadenarse mediante capas. Esta es la base del Deep Learning. Podemos tener un sistema que sea capaz de realizar una tarea, pre-entrenarlo y usar ese sistema ya entrenado con otros datos, añadiendo nuevas capas.

Esas nuevas capas son las que vamos a entrenar, dejando el resto de parámetros con lso valores establecidos en el pre-entrenamiento.

# 4.7 Arquitectura de un sistema de ML

## Deployment




Software engineering may be needed for:

- Ensure reliable and efficient predictions
- Scaling
- Logging
- System monitoring
- Model updates

MLOps
machine learning
operations

# 4.8 ROC curve

Una curva ROC (curva de característica operativa del receptor) es un gráfico que muestra el rendimiento de un modelo de clasificación. Esta curva representa dos parámetros:

- Tasa de verdaderos positivos
- Tasa de falsos positivos

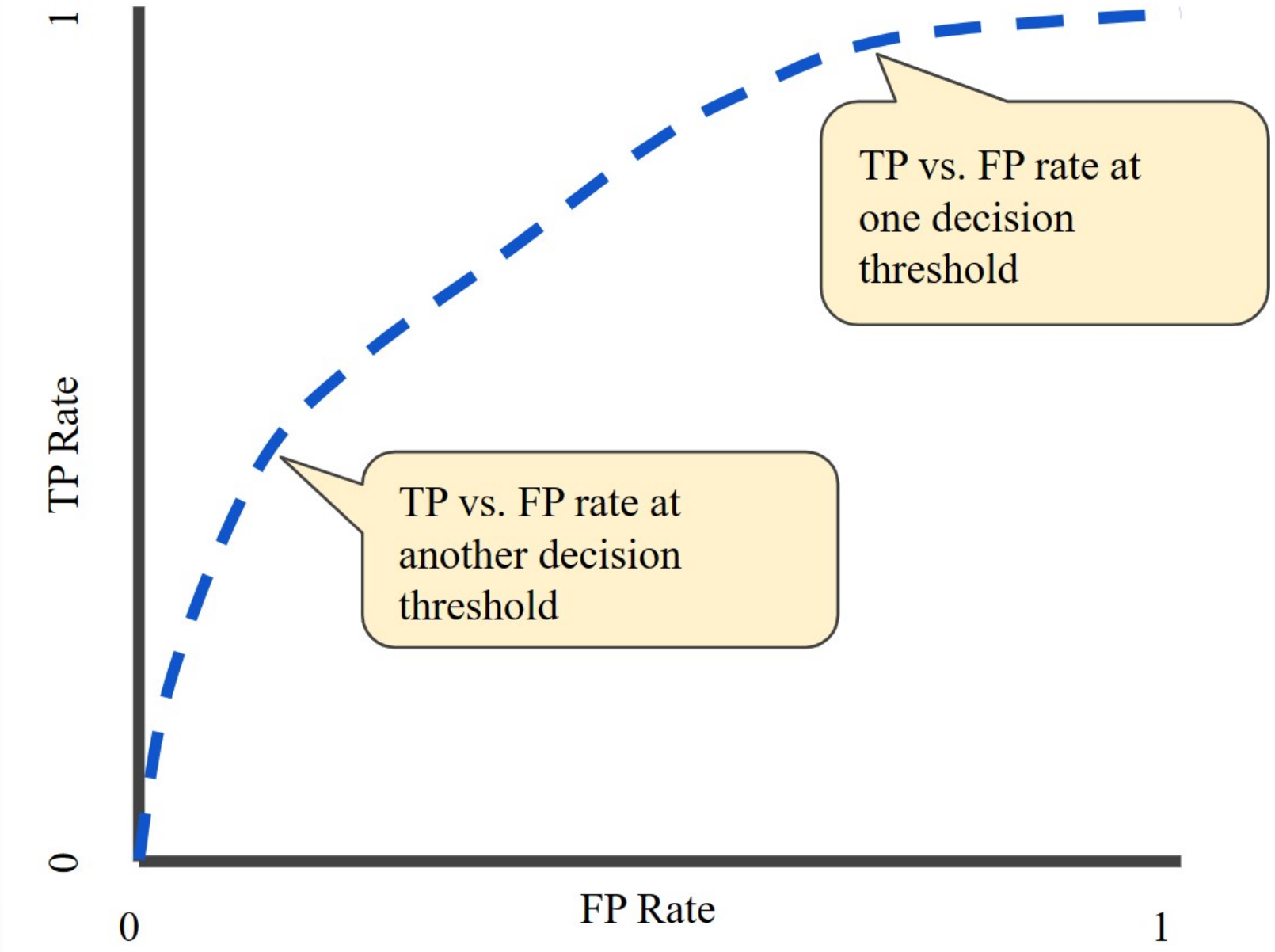
Tasa de falsos positivos (FPR) se define de la siguiente manera:

$$FPR = \frac{FP}{FP + TN}$$

Los clasificadores discretos, como los Árbol de decisión o los sistemas de reglas, dan como resultados a valores numéricos una etiqueta binaria. Estos sólo proporcionan un único punto en el espacio ROC.

Para otros clasificadores, como una Red neuronal artificial o regresión logística, la salida son valores de probabilidad que representan hasta qué punto una instancia pertenece a una de las dos clases.

Para estos métodos se debe fijar un valor **umbral** que determinará un punto en el espacio ROC. Si fijamos el umbral a 0.8, la probabilidad >= a 0.8 será positiva.


1
TP Rate
0
TP vs. FP rate at one decision threshold
TP vs. FP rate at another decision threshold
0
FP Rate
1

# 4.8.1 ¿Cómo dibujar la curva ROC?

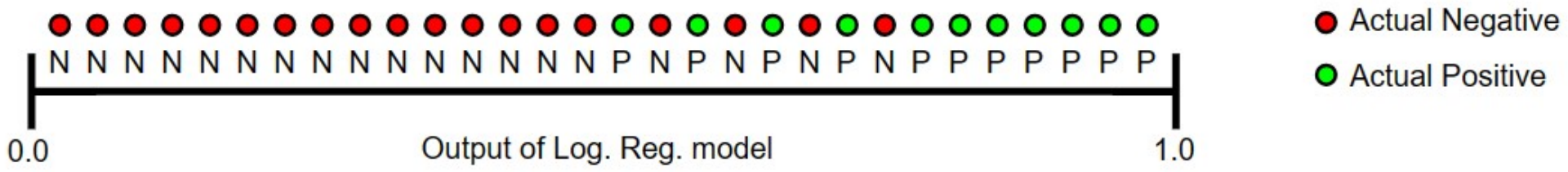
Para cada umbral que decidamos, podemos construir una matriz de confusión, y calcular el punto en el espacio ROC correspondiente.

Según vamos variando el umbral (por ejemplo, en pasos de 0.1) tendríamos una nueva matriz de confusión y un nuevo punto en el espacio ROC. Dibujar la curva ROC consiste en poner juntos todos los puntos correspondientes a todos los umbrales o puntos de corte.

# 4.8.2 AUC: Área bajo la curva ROC

El AUC proporciona una medida agregada del rendimiento en todos los umbrales de clasificación posibles. Es el indicador más utilizado para determinar la calidad del clasificador.

Se puede interpretar como la probabilidad de que un clasificador ordenará o puntuará una instancia positiva elegida aleatoriamente más alta que una negativa.



# 4.8.3 Ejemplo de cómo se calcula la curva ROC

```
import numpy as np
from sklearn import metrics
y = np.array([1, 1, 2, 2])
scores = np.array([0.1, 0.4, 0.35, 0.8])
fpr, tpr, thresholds = metrics.roc_curve(y, scores, pos_label=2)
fpr #array([0. , 0. , 0.5, 0.5, 1. ])
tpr #array([0. , 0.5, 0.5, 1. , 1. ])
thresholds #array([ inf, 0.8 , 0.4 , 0.35, 0.1
```

Entrada: salidas reales (0,1) del conjunto de test y las predicciones de probabilidades obtenidas del modelo para la clase 1.
Devuelve: la tasa de falsos positivos y la tasa de verdaderos positivos para cada umbral así como la lista de valores utilizados como umbral.

# 4.8.4 Ejemplo de Como se calcula UAC

```
>>> from sklearn.linear_model import LogisticRegression
>>> from sklearn.metrics import roc_auc_score
>>> X, y = load_breast_cancer(return_X_y=True)
>>> clf = LogisticRegression(solver="liblinear", random_state=0).fit(X, y)
>>> roc_auc_score(y, clf.predict_proba(X)[:, 1]) # 0.99...
# multi_class='ovr' for more than two clases.
roc_auc_score(y, clf.decision_function(X)) # 0.99...
```

Devuelve el valor de AUC ,comprendido entre 0.5 (aleatorio) y 1.0 (perfecto). Modelo con predicciones 100% incorrectas => AUC de 0.0; predicciones 100% correctas => AUC de 1.0.

**NOTA**: si un clasificador es muy malo (< 0.5) se puede construir un clasificador inverso. No tiene sentido valores por debajo de 0.5

# 4.9 Cuestiones éticas

Todos lo metodos de machine learning utilizan sesgos (bias) para aprender. Hay que tener mucho cuidado con los sesgos que introduzcamos artificialmente. Ejemplos:

- Datos de entrenamiento sesgados por género, etnia, etc:
  - Sistemas de reconocimiento facial empareja individuos de piel oscura con criminales.
  - Aprobación sesgada de prestamos bancarios.
  - Refuerzo de estereotipos negativos

# 4.9.1 Malos usos de la IA

- Ataques adversarios.
- Deepfakes
- Generate fake content: fake news, etc.
- Cometer fraudes
- Atentar contra la dignidad, integridad de las personas (por ejemplo: fake nudes)

# 4.9.1 Responsabilidad de los errores de la IA

Si un vehículo autónomo atropella a un peatón. ¿De quién es la responsabilidad?

- ¿Del creador del algoritmo?
- ¿Del que lo puso en producción?