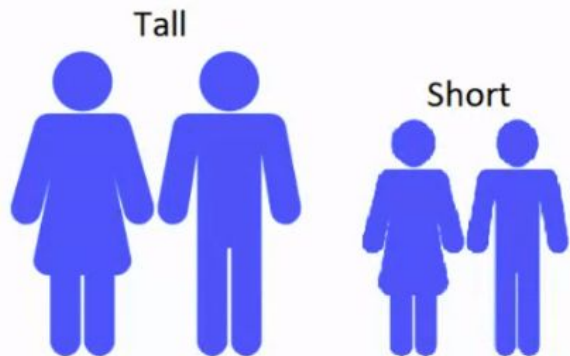
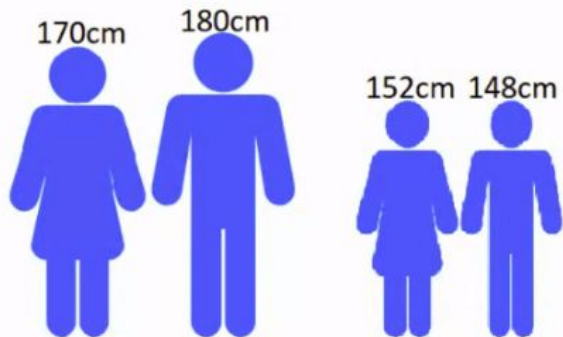


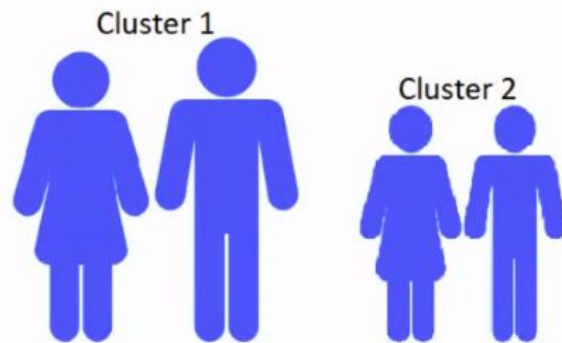
Machine Learning - Unsupervised



Classification



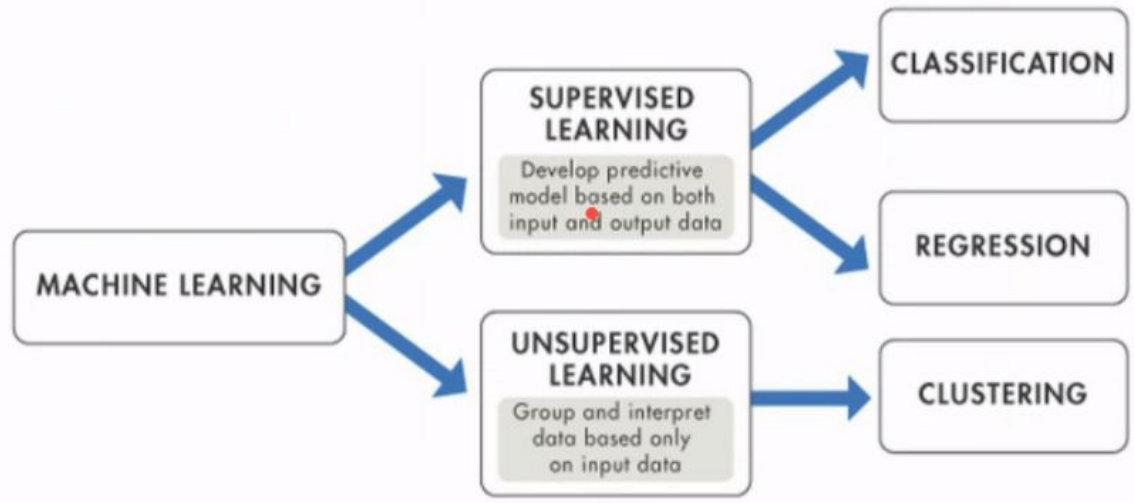
Regression



Clustering



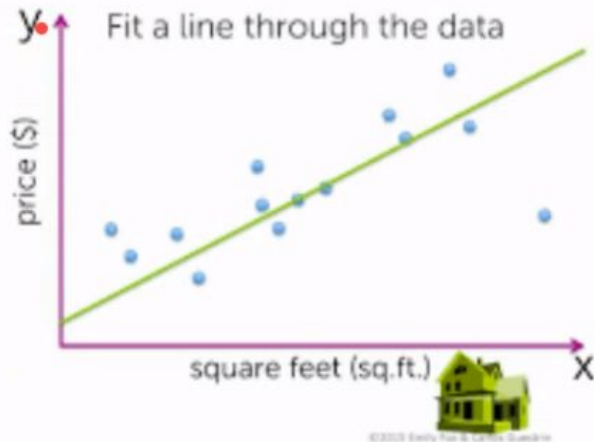
Data etiquetada o no



Tipos

Regresión

Use a **linear** regression model mojix



15

Clasificación



Supervised Learning



CAT

UnSupervised Learning





Dataset

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa

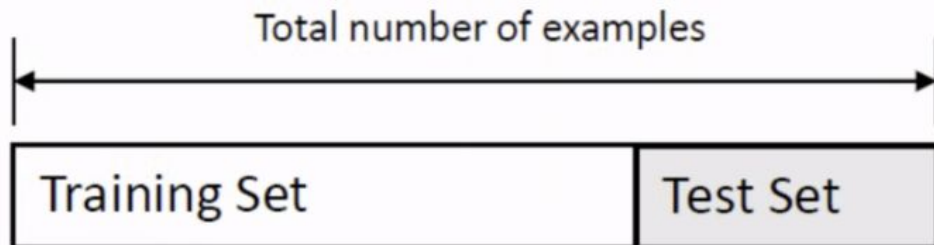
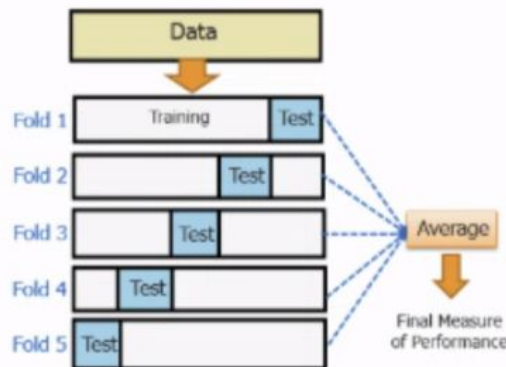
Training

Validation

Test

m x n

Cross Validation



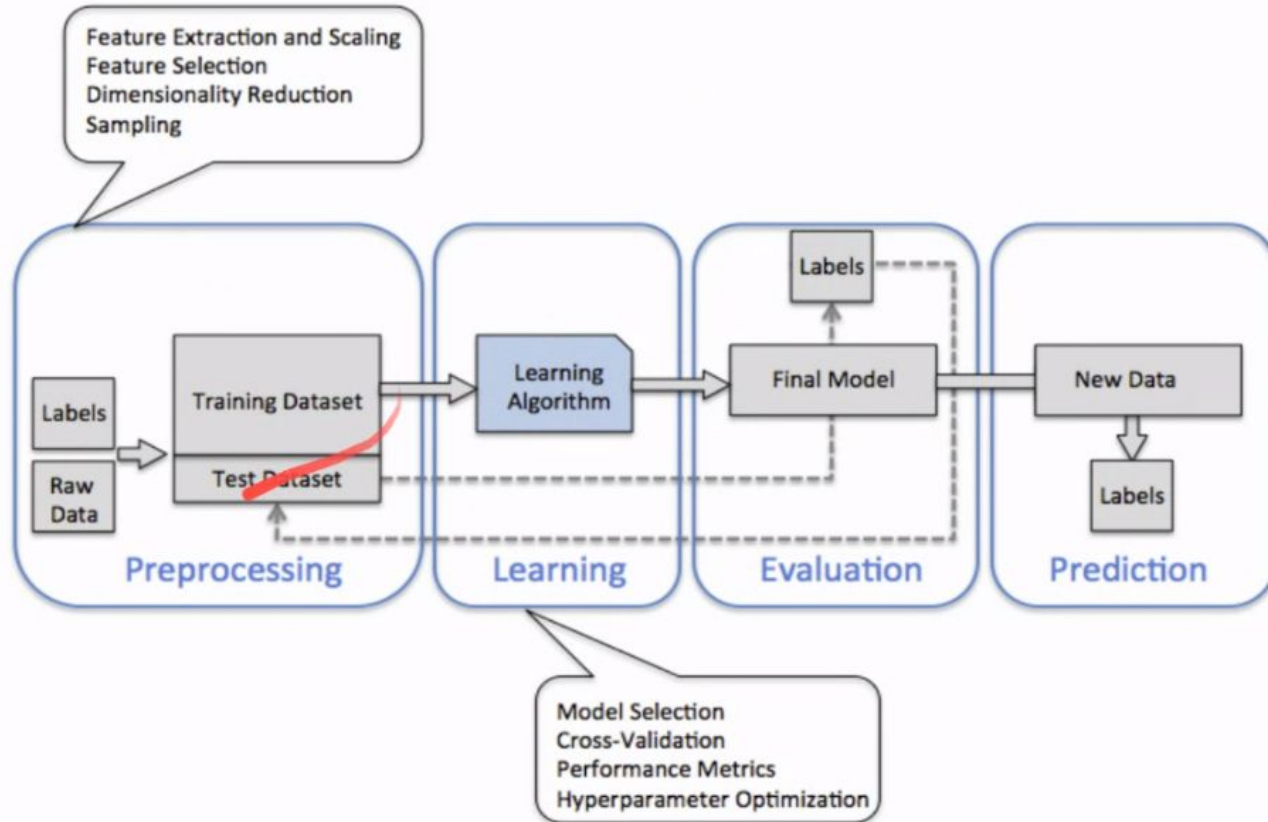
Usual data distributions

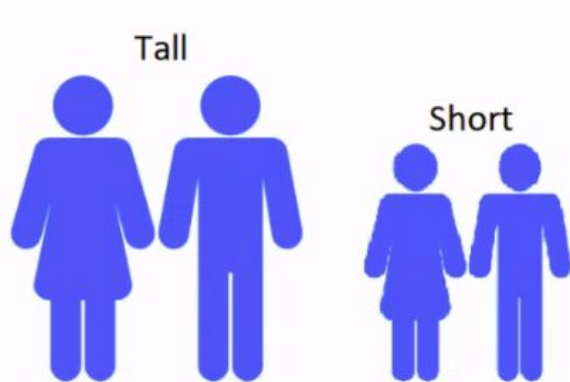
Regression: 80%-20%, 70%-30%

Classification: 90%-10%

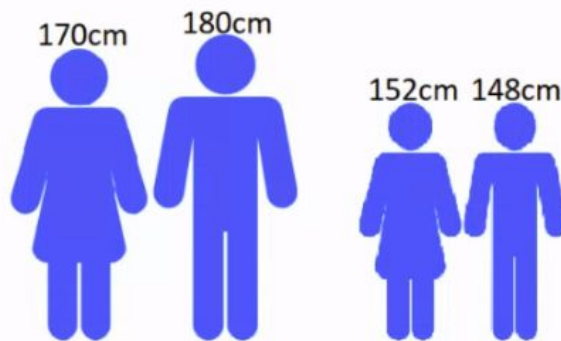


El proceso de Machine Learning

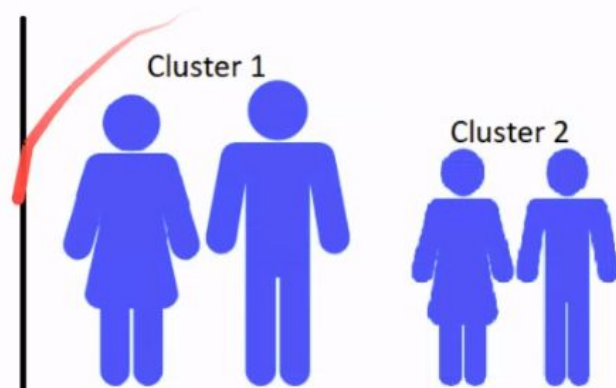




Classification



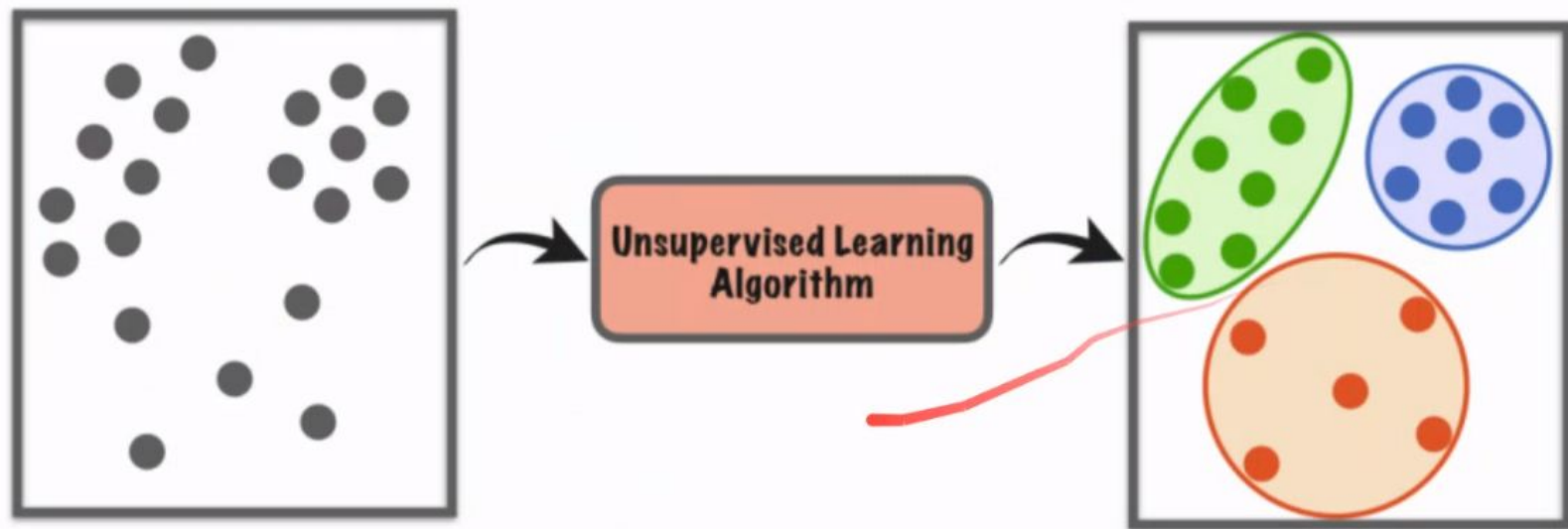
Regression



Clustering

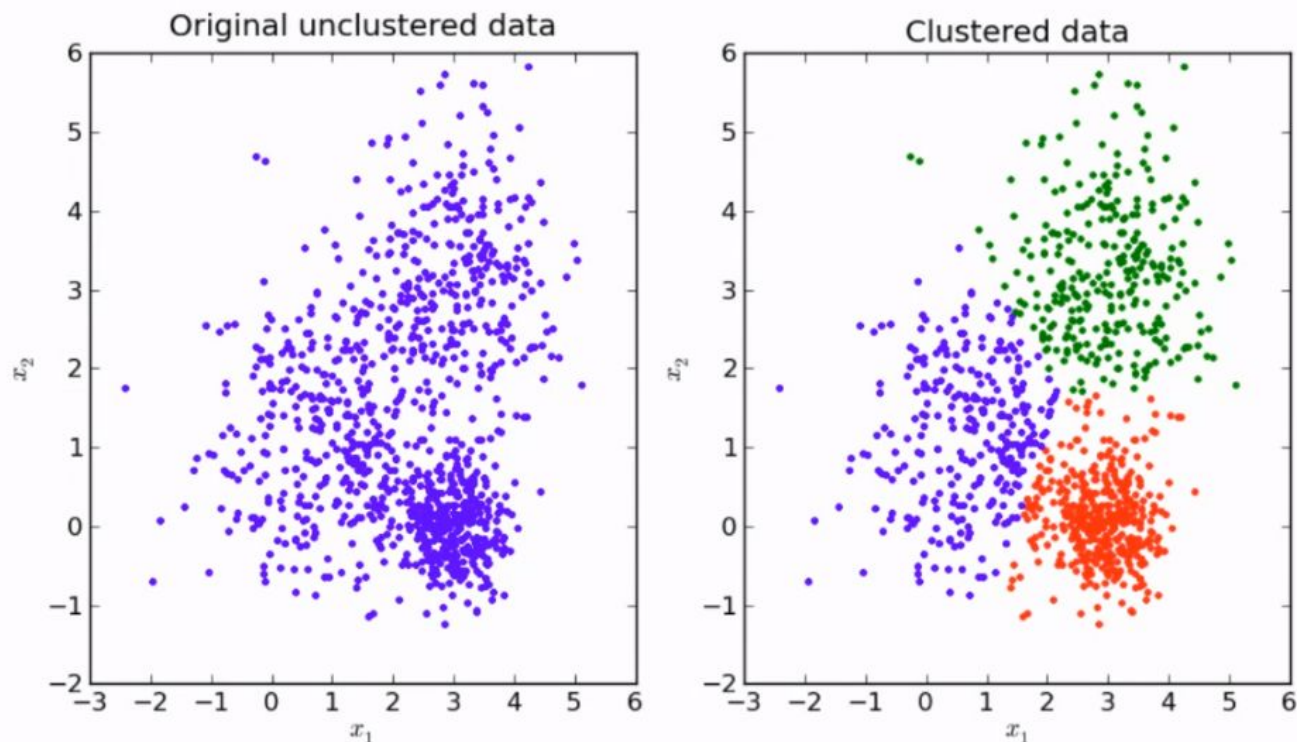


Aprendizaje No Supervisado





Clustering



<https://scikit-learn.org/stable/modules/clustering.html>



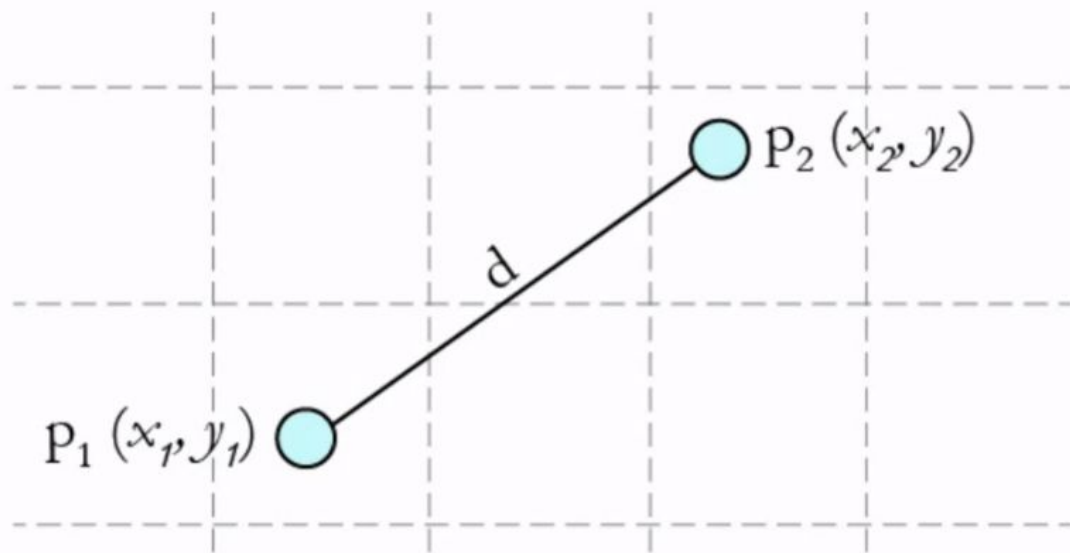
K-Means Algorithm

Iterative, hard, flat clustering algorithm based on Euclidean distance

- Specify k , the number of clusters to be generated
- Choose k points at random as cluster centers
- Assign each instance to its closest cluster center using Euclidean distance
- Calculate the centroid (mean) for each cluster, use it as a new cluster center
- Reassign all instances to the closest cluster center
- Iterate until the cluster centers don't change anymore



Distancia

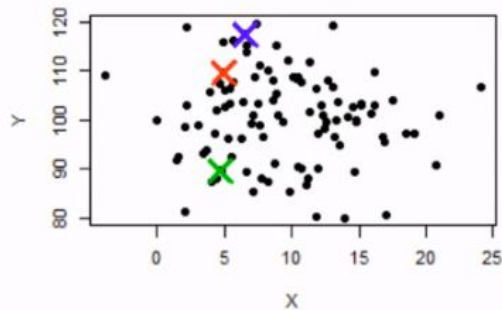


$$\text{Euclidean distance } (d) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

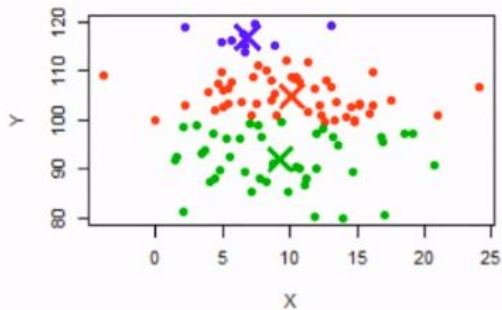


K-means

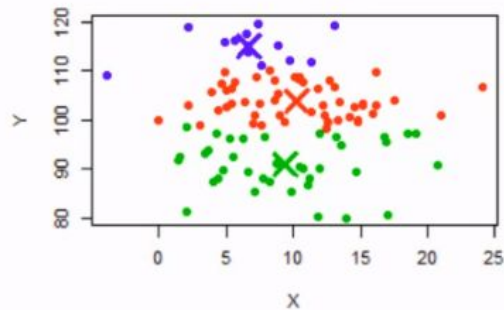
Iteration 1



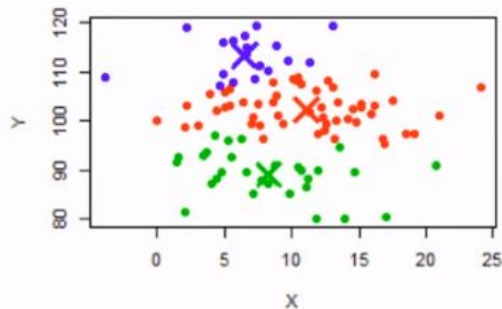
Iteration 2



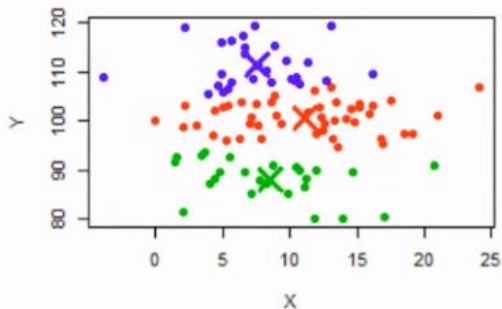
Iteration 3



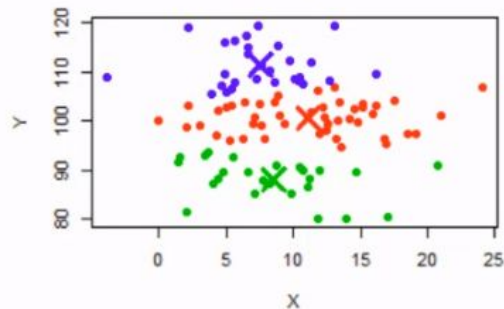
Iteration 6



Iteration 9



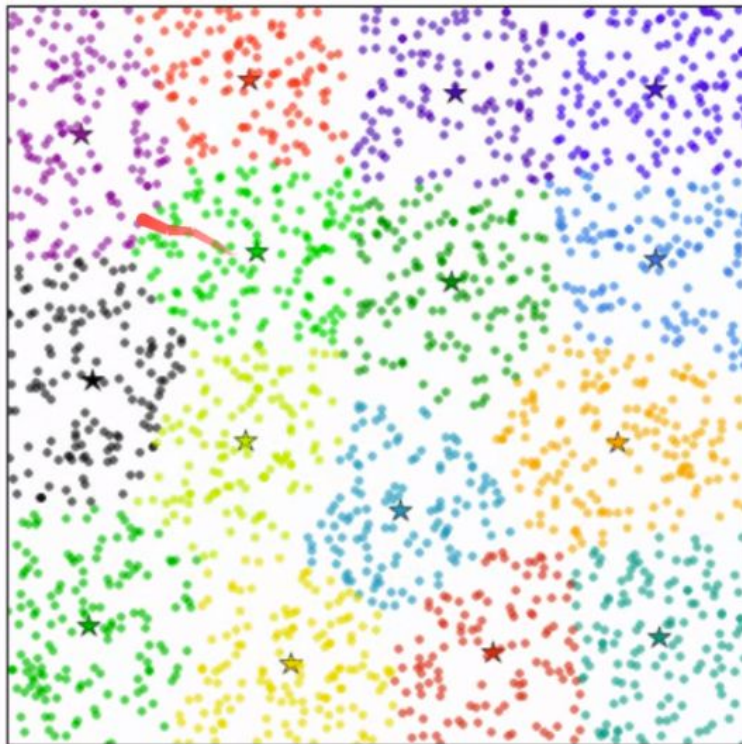
Converged!





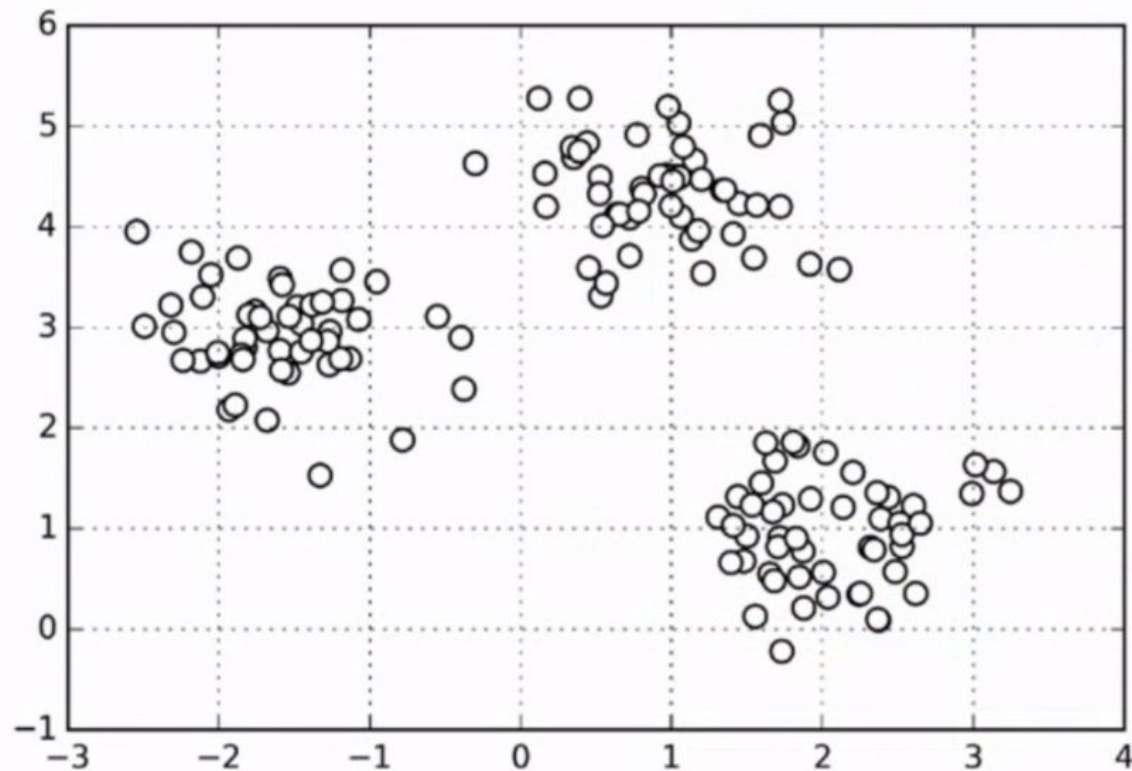
K-means

$N = 2000$, $K = 15$ Iteration 18





K means



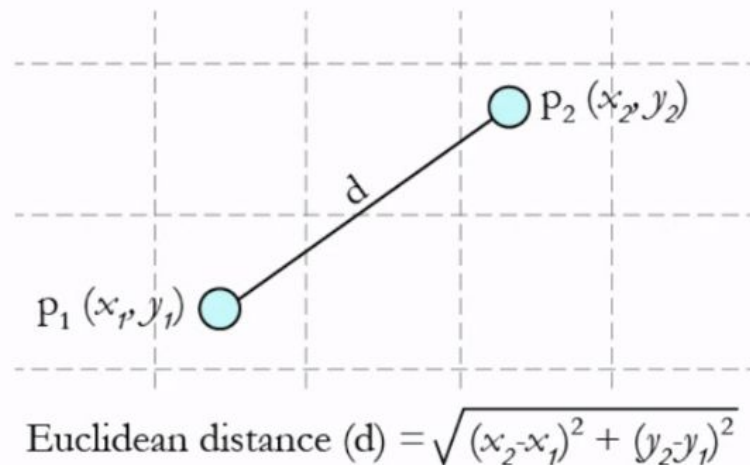
- Definir K
- No tenemos etiquetas (Ground Truth) para comparar



K-Means Algorithm

Iterative, hard, flat clustering algorithm based on Euclidean distance

- Specify k , the number of clusters to be generated
- Choose k points at random as cluster centers
- Assign each instance to its closest cluster center using Euclidean distance
- Calculate the centroid (mean) for each cluster, use it as a new cluster center
- Reassign all instances to the closest cluster center
- Iterate until the cluster centers don't change anymore



COLAB

<https://colab.research.google.com/drive/199MZvaBiKI3zrn0-gyNcAS8GCU96IP26?usp=sharing>

- El problema radica el nro de clusters

IMPORTAR DATASET DE SCIKTLEARN

BUSCAR

centros=centroides = 3

classes

- Blobs =>
- classes => grupos

```
kmeans = KMeans(n_clusters=3)
```

```
kmean.fit(blobs)
```

K-means as an Optimization Problem

Based on this Euclidean distance metric, we can describe the k-means algorithm as a simple optimization problem, an iterative approach for minimizing the within-cluster sum of squared errors (SSE), which is sometimes also called cluster inertia:

$$SSE = \sum_{i=1}^n \sum_{j=1}^k w^{(i,j)} \left\| \mathbf{x}^{(i)} - \boldsymbol{\mu}^{(j)} \right\|_2^2$$

Here, $\boldsymbol{\mu}^{(j)}$ is the representative point (centroid) for cluster j , and $w^{(i,j)} = 1$ if the sample $\mathbf{x}^{(i)}$ is in cluster j ; $w^{(i,j)} = 0$ otherwise.



Hard vs Fuzzy clustering

HARD: Un elemento pertenece a un solo cluster

$$\begin{bmatrix} \mu^{(1)} \rightarrow 0 \\ \mu^{(2)} \rightarrow 1 \\ \mu^{(3)} \rightarrow 0 \end{bmatrix}$$

FUZZY: Probability membership

$$\begin{bmatrix} \mu^{(1)} \rightarrow 0.1 \\ \mu^{(2)} \rightarrow 0.85 \\ \mu^{(3)} \rightarrow 0.05 \end{bmatrix}$$

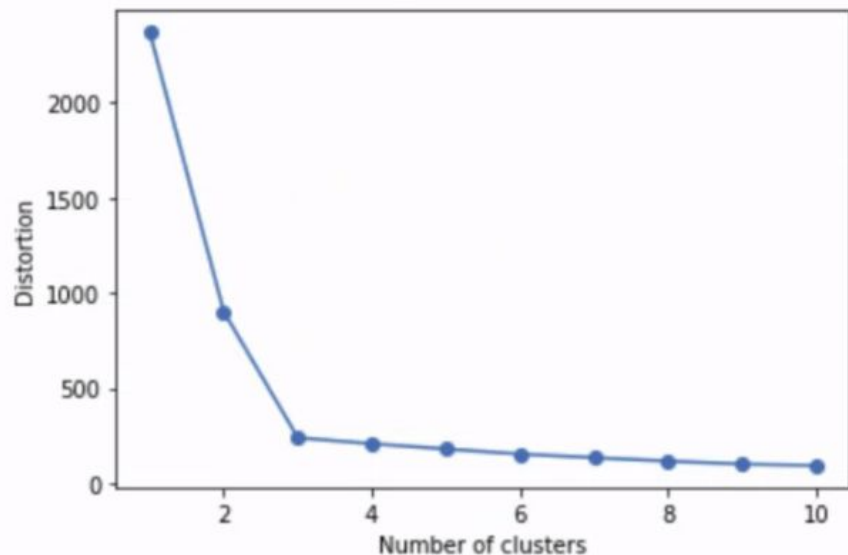
μ

Centroides

Fuzzy C-means



K óptimo → Elbow Method



- Within-cluster SSE (distortion)
- If K increases distortion will decrease, because samples will be closer.

Goal:

Identify the value of k where the distortion begins to increase



Silhouette Coefficient

Silhouette Coefficient or silhouette score is a metric used to calculate the goodness of a clustering technique. Its value ranges from -1 to 1.

1: Means clusters are well ~~apart from~~ each other and clearly distinguished.

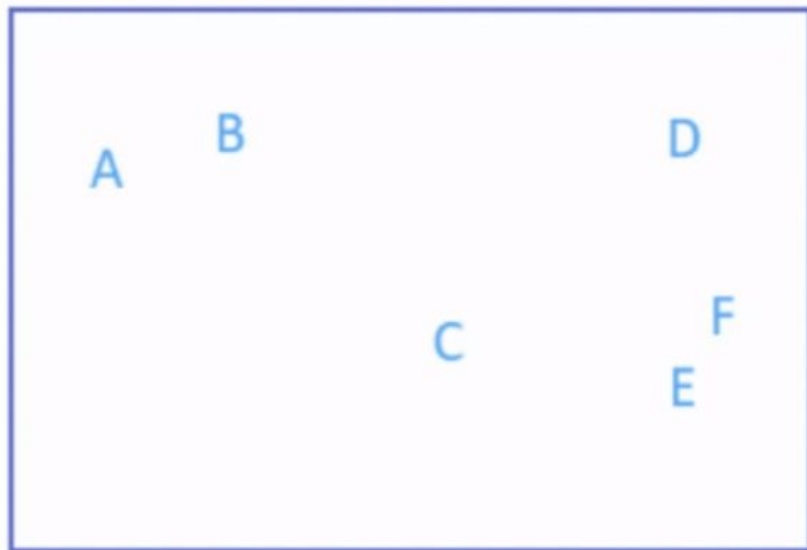
0: Means clusters are indifferent, or we can say that the distance between clusters is not significant.

-1: Means clusters are assigned in the wrong way.

<https://colab.research.google.com/drive/1wPTr22hDWloyU1Nn0bl2oxfmWwAjrARo?usp=sharing>



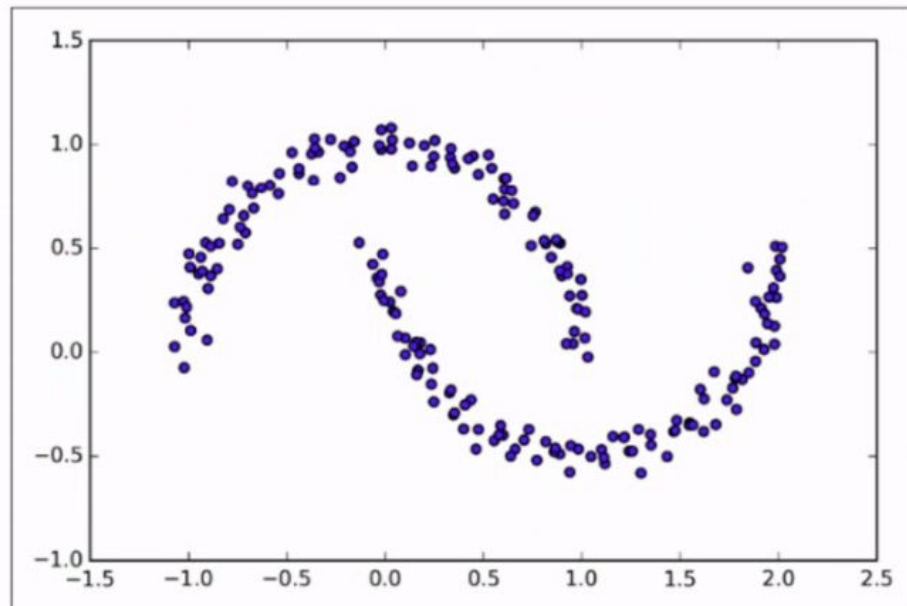
Hierarchical Clustering



Dendrogram



Density-based Spatial Clustering of Applications with Noise (DBSCAN). The notion of density in DBSCAN is defined as the number of points within a specified radius ϵ .





Unsupervised Learning

Unsupervised Learning

- Unsupervised learning: a set of statistical tools for data for which only features/inputs are available
 - We have X 's but no associated labels Y
 - Goal: discover interesting patterns/properties of the data
- e.g. for visualizing or interpreting high-dimensional data



Unsupervised Learning

Why is unsupervised learning challenging?

- Exploratory data analysis - goal is not as clearly defined
- Difficult to assess performance - "right answer" unknown
- Working with high-dimensional data



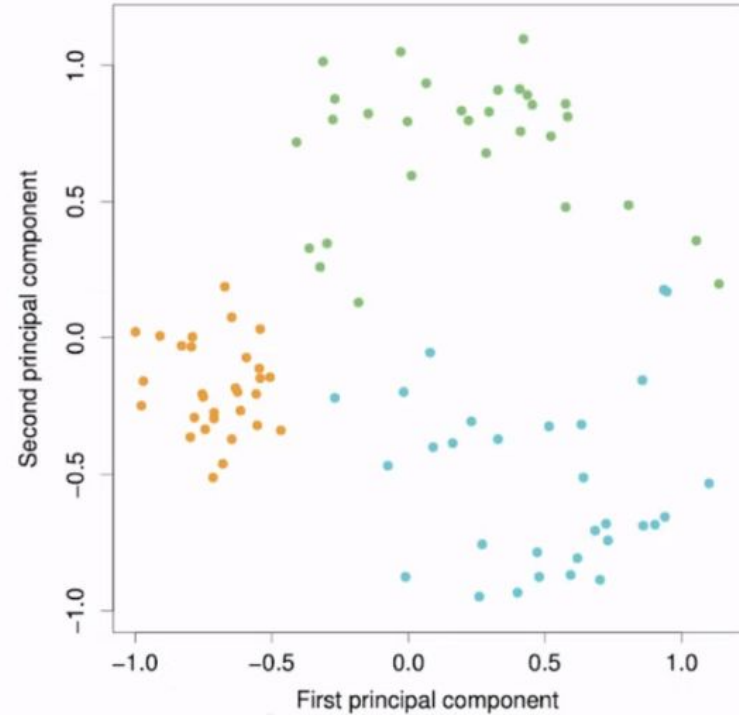
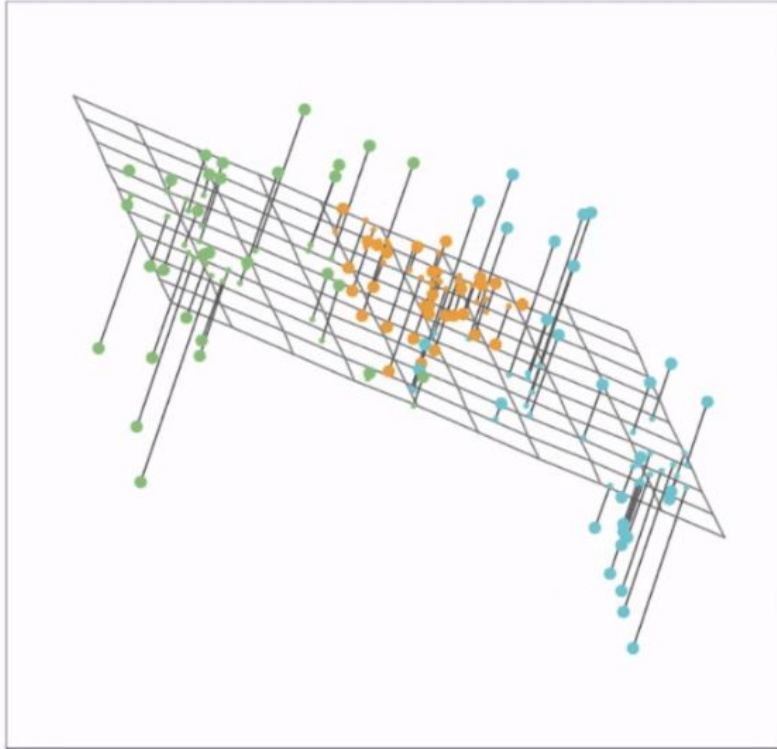
Unsupervised Learning

Two approaches:

- Cluster Analysis
 - For identifying homogeneous subgroups of samples
- Dimensionality Reduction
 - For finding a low-dimensional representation to characterize and visualize the data

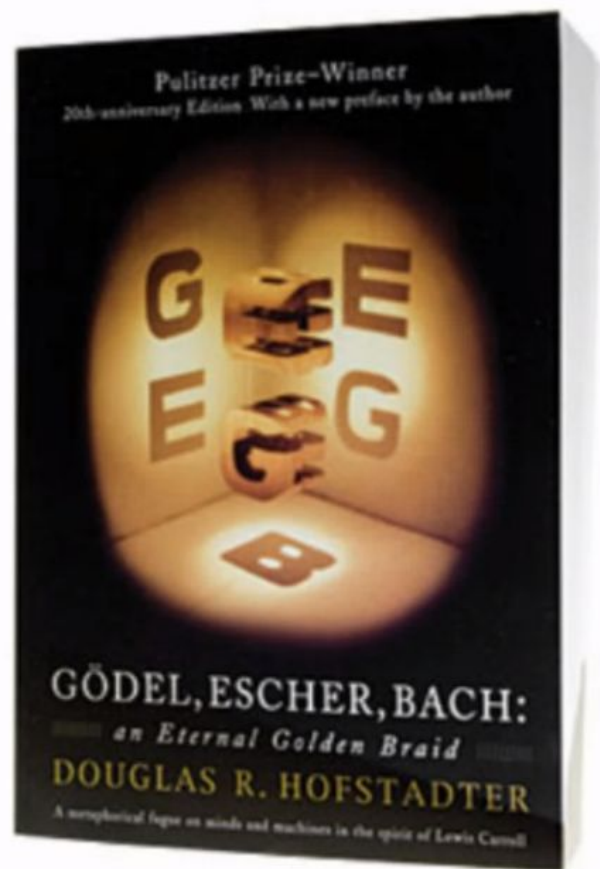


Dimensionality Reduction





Projections

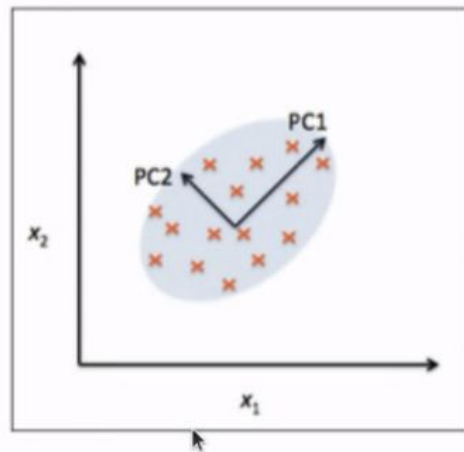




Principal Component Analysis

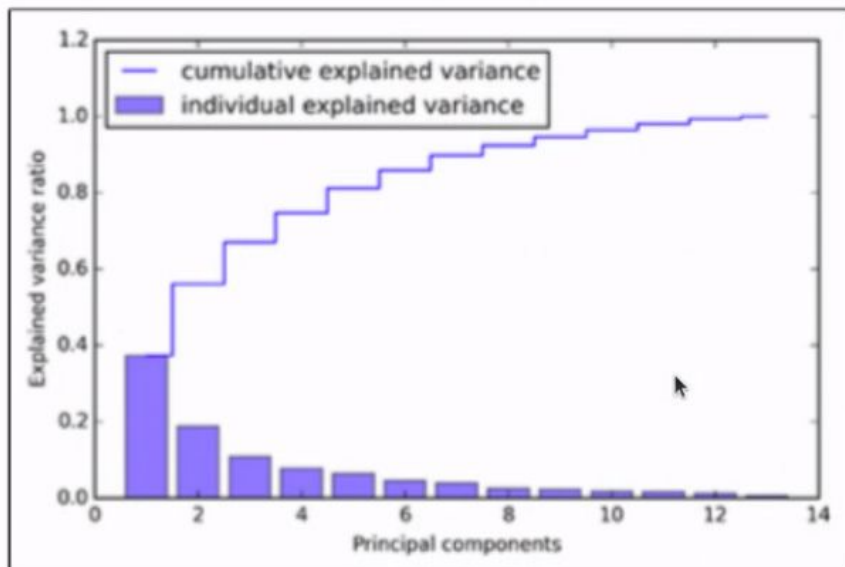
Before looking at the PCA algorithm for dimensionality reduction in more detail, let's summarize the approach in a few simple steps:

1. Standardize the d -dimensional dataset.
2. Construct the covariance matrix.
3. Decompose the covariance matrix into its eigenvectors and eigenvalues.
4. Select k eigenvectors that correspond to the k largest eigenvalues, where k is the dimensionality of the new feature subspace ($k \leq d$).
5. Construct a projection matrix W from the "top" k eigenvectors.
6. Transform the d -dimensional input dataset X using the projection matrix W to obtain the new k -dimensional feature subspace.





Cumulative explained variance



PCA

https://colab.research.google.com/drive/12VL_fYaT04EGNKfHsQP-CrSvIOKwCPV4?usp=sharing

<https://www.kaggle.com/uciml/pima-indians-diabetes-database>

[INTERNAL] ML_Bootcamp_2021 ☆ 📷 🔄

File Edit View Insert Format Slide Arrange Tools Add-ons Help Last edit was 2 hours ago



Present

Share

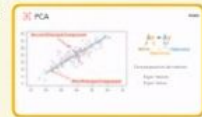


+ ↶ ↷ 🖨️ 📌 🔍 🖱️ 📄 🗑️ 🧰 📏 Background Layout Theme Transition

78



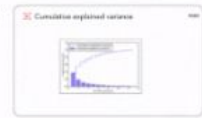
79



80



81

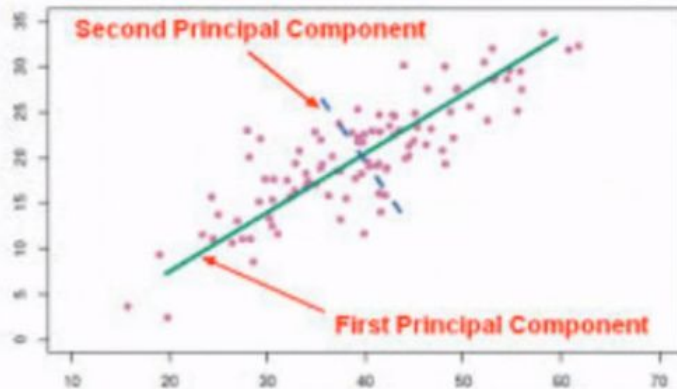


82



PCA

mojix



$$A v = \lambda v$$

Matrix Eigenvalue
Eigenvector

Descomposición de matrices

- Eigen Vectors
- Eigen Values

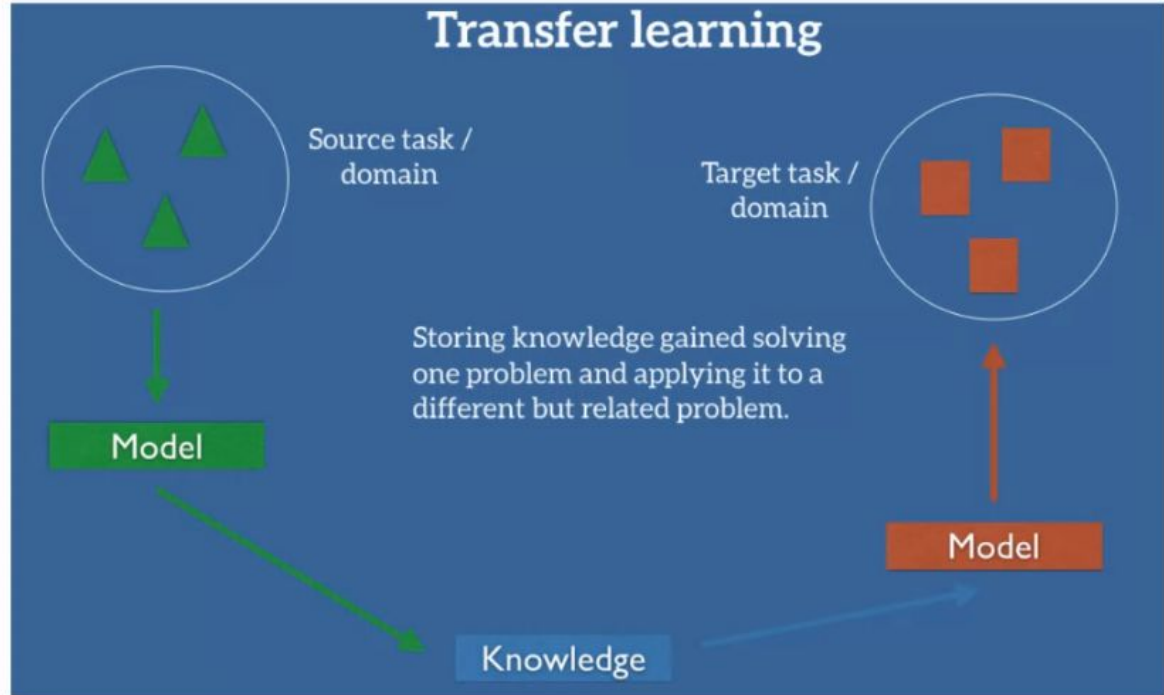
Explore

Show all

archive (2).zip

Transfer Learning

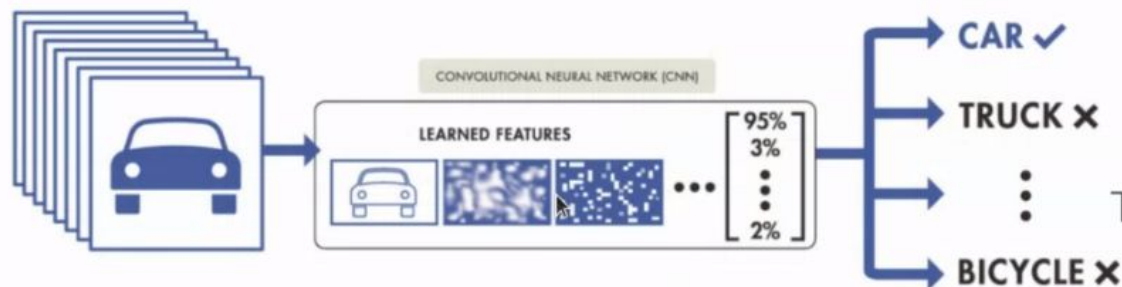
Transfer Learning is the process of applying an existing trained ML model to a new, but related, problem.





Transfer Learning

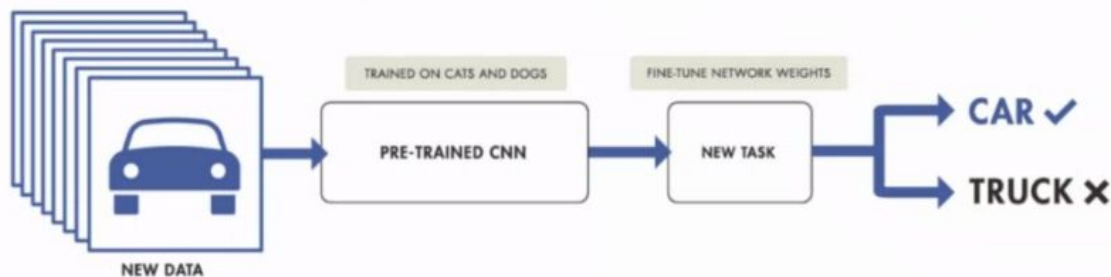
TRAINING FROM SCRATCH



Training Options:

- Use the original part of the network as feature extractor.
- Fine tuning the whole network.

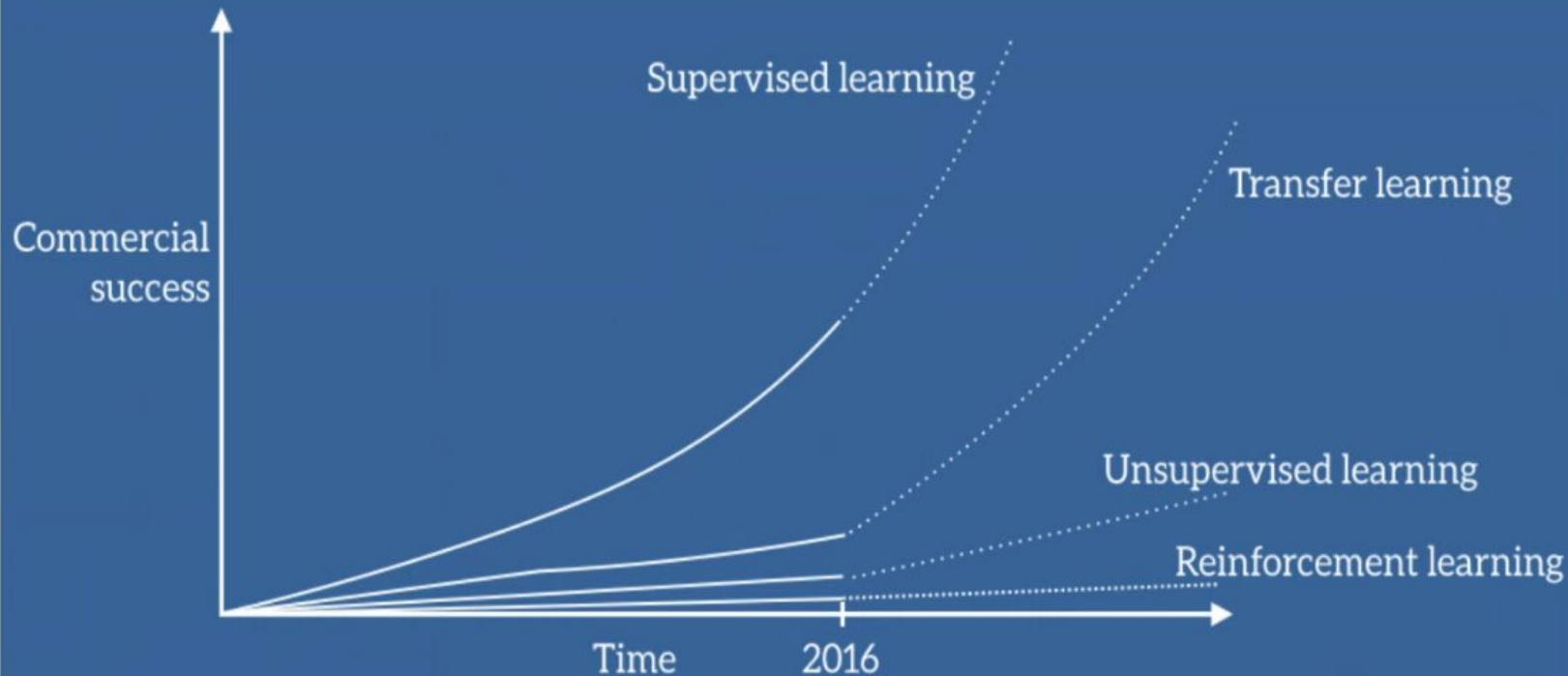
TRANSFER LEARNING





Why Transfer Learning?

Drivers of ML success in industry



El Reto

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language
[jacobdevlin@google.com, mingw@cs.cmu.edu, kentonl@google.com, toutanova@google.com]

Abstract

We introduce a new language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers. Unlike most language representation models (Peters et al., 2018; Radford et al., 2019), BERT is designed to pre-train deep bidirectional representations from unlabeled text by predicting missing words in one direction and predicting masked tokens in the other.

BERT is conceptually simple and remarkably powerful. It shows state-of-the-art results on eleven natural language processing tasks, including reading the TEAC, even on 30,000 770 word abstracts, representing 100,000 words in 10,000 1000 word paragraphs, and 10,000 1000 word paragraphs in 10,000 1000 word paragraphs.

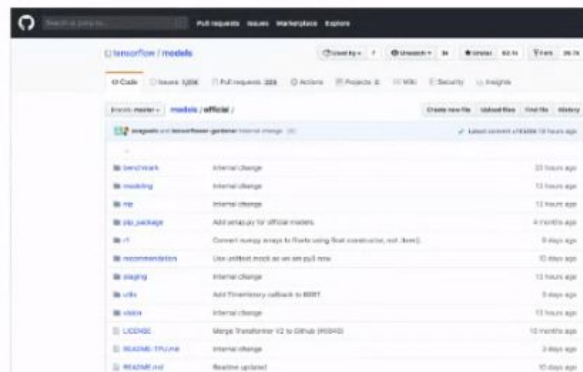
1 Introduction

Language model pre-training has been shown to be effective for improving many natural language processing tasks (Chen and Li, 2019; Peters et al., 2018; Radford et al., 2019; Brown and Balle, 2019). These include sentence-level tasks such as natural language inference (Shenkar et al., 2017; Williams et al., 2018) and paraphrasing (Chen and Beaulieu, 2018), which are in turn used for identifying between sentences for matching them. In addition, as well as token-level tasks such as word-level representations and question answering, where models are required to produce fine-grained output as the token level (Gong, 2018; Wang and Hu, 2018; Ruder et al., 2018).

There are two existing strategies for applying pre-trained language representations to downstream tasks: Joint-training and Pre-training. The Joint-training approach, such as ELMo (Peters et al., 2018), uses task-specific architectures that include the pre-trained representations as additional layers. The fine-tuning approach, such as the Generative Pre-trained Transformer (GPT) (Radford et al., 2019), introduces unidirectional task-specific parameters, and is trained on the downstream tasks by simply discarding all pre-trained parameters. The two approaches share the same objective function during pre-training when they use unidirectional language models to learn general language representations.

We argue that current techniques restrict the power of the pre-trained representations, especially for the fine-tuning approaches. The use of unidirectional pre-trained language models is suboptimal, and this limits the choice of architectures that can be used during pre-training. For example, in GPT, the authors use a unidirectional architecture, where only tokens are only at and previous tokens in the self-attention layer of the Transformer (Vaswani et al., 2017). Such restrictions are not optimal for certain downstream tasks, and could be very harmful when applying fine-tuning, based approaches to token-level tasks such as question answering, where it is useful to include context from both directions.

In this paper, we improve the fine-tuning based approaches by pre-training BERT, Bidirectional Encoder Representations from Transformers. BERT introduces the previously mentioned model, unidirectional, by using a "masked language model" (MLM) pre-training objective, as stated by the Chinese task (Chen, 2018). The masked language model randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary of the masked



¿Cómo lo uso?

¿Es seguro?

¿Cuento con todos los datos?

¿Es la última versión?

TensorFlow Hub es una biblioteca de módulos de aprendizaje automático reutilizables.

TensorFlow Hub es una biblioteca para la publicación, el descubrimiento y el consumo de módulos reutilizables de modelos de aprendizaje automático. Un *módulo* es una pieza autónoma de un grafo de TensorFlow, junto con sus pesos y recursos, que se puede reutilizar para diferentes tareas en procesos de aprendizaje por transferencia. Con el aprendizaje por transferencia, se puede realizar lo siguiente:

- Entrenar un modelo con un conjunto de datos más pequeño
- Mejorar la generalización
- Acelerar el entrenamiento

[Explorar módulos en tfhub.dev](#)

```
!pip install "tensorflow_hub>=0.6.0"
!pip install "tensorflow>=2.0.0"

import tensorflow as tf
import tensorflow_hub as hub

module_url = "https://tfhub.dev/google/nlm-en-dim128/2"
embed = hub.KerasLayer(module_url)
embeddings = embed(["A long sentence.", "single-word",
                    "http://example.com"])
print(embeddings.shape)  #(3,128)
```

TensorFlow Hub

tfhub.dev[Explorar módulos](#)

Introducción a TensorFlow Hub

[Leer en el blog de TensorFlow](#)

TensorFlow Hub en la Dev Summit

[Ver el video](#)

TensorFlow Hub en GitHub

[Ver en GitHub](#)

TensorFlow Hub

#TensorFlowCommunityTraining

Una colección completa de modelos



Image



Text



Video



Audio

Modelos Listos para usar

#TensorFlowCommunityTraining

Modelos pre entrenados listos para transmitir el aprendizaje a tus propios datasets y además son deployables en varios tipos de dispositivos



TensorFlow
Extended



TensorFlow
JS

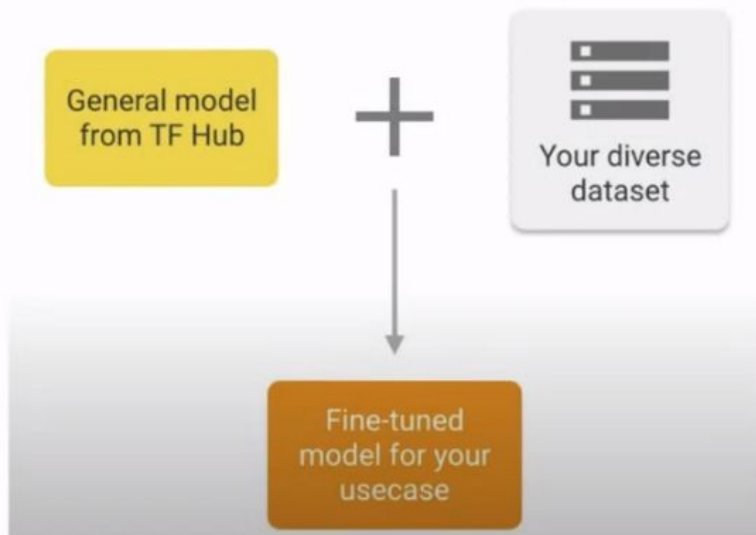


TensorFlow
Lite



Coral

Transfer Learning



Productive Machine Learning

<https://www.youtube.com/watch?v=SJ1LGUyw-Xg>

From researchers to you

Universal Sentence Encodings

Daniel Cer^a, Yinfei Yang^a, Angus Kong^a, Nan Hua^a, Nicole Limtiaco^b,
Rhomni St. John^a, Noah Constant^a, Mario Guajardo-Céspedes^a, Steve Yuan^a,
Chris Tar^a, Yun-Hsuan Sung^a, Brian Strope^a, Ray Kurzweil^c

^aGoogle Research
Mountain View, CA

^bGoogle Research
New York, NY

^cGoogle
Cambridge, MA

Abstract

models are implemented in TensorFlow (Abadi et al., 2016) and are available to download as pre-trained models here:

<https://tfhub.dev/google/universal-sentence-encoder/1>

The models take as input English strings and produce as output a fixed dimensional embedding representation of the string. The encoding models

nington et al., 2014). Recent work has demon-
transfer results using pre-trained
embeddings (Comneau et al., 2017).
r, we present a collection of models
sentence embeddings that demon-
transfer learning to a number of other
tasks. The sentence encoding mod-
are made publicly available. We in-
sents with varying amounts of trans-
fer data to illustrate the interac-
the impact of transfer learning and
ize. Engineering characteristics of
for transfer learning are important.
modeling trade-offs regarding mem-
ory as well as compute time on CPU/
source consumption comparisons are
ences of varying lengths.

Toolkit

able two new models for encoding
embedding vectors. One makes use
of the Transformer (Vaswani et al., 2017) architec-
ture, while the other is formulated as a Deep Aver-
aging Network (DAN) (Iyyer et al., 2015). Both
models are implemented in TensorFlow (Abadi

The pre-trained sentence encoding models
reported in the paper are made freely avail-
able for download.

of the Transformer (Vaswani et al., 2017) architec-
ture, while the other is formulated as a Deep Aver-
aging Network (DAN) (Iyyer et al., 2015). Both
models are implemented in TensorFlow (Abadi

Activities

Google Chrome

S

DG

TJ

archive (2).zip

Show all

Thu Oct 28 8:59 PM

INTERNAL

Clustering

Copia de

Clustering

Copia de

Copia de

PCA [MO]

Copia de


Pima India

docs.google.com/presentation/d/1AtFK-RPNjnEzL3w08tgVPZmj9vBb_oHJQsC6LQZXVos/edit#slide=id.gd274f44f61_0_1166

Colab

mo

<https://colab.research.google.com/drive/1slkBSZGVi9BIkjD4JilLIV94PBw5A0O8?usp=sharing>



TENSORFLOW HUB

<https://colab.research.google.com/drive/1slkBSZGVi9BlkjD4JiILIV94PBw5A008?usp=sharing>

Activities

Google Chrome

Thu Oct 28 9:01 PM

100%

INTE x Clust x Copia x Clust x Copia x Copia x PCA x Copia x k Pima x Tenso x Copia x Tenso x

colab.research.google.com/drive/19YFy6QQ6QDFKEhVSpeUprPvao8hnV3F1#scrollTo=3u9J8_QhxamK

Colab de TensorFlowHub_TheBasics_[MOJIX].ipynb

Comentario Compartir

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda Se han guardado todos los cambios

+ Código + Texto

RAM Disco

Editar

<https://www.tensorflow.org/hub/hosting>

```
# Vamos a crear una variable con la url del modelo
# https://tfhub.dev/<publisher>/<model_name>/<version>

modelo_tfhub = "https://tfhub.dev/google/imagenet/mobilenet_v2_100_224/classification/4"
```

▼ Paso 3: Define la arquitectura de tu modelo

Revisa la documentación del modelo que escogiste

```
[ ] IMAGE_SHAPE = (224, 224)

modelo = Sequential([
    hub.KerasLayer(modelo_tfhub, input_shape=IMAGE_SHAPE+(3,))
])
```

▼ Paso 4: Escoge una imagen para realizar las predicciones

```
[ ] imagen_test_url = 'https://live.staticflickr.com/5449/7440792338_eb509ba9da_b.jpg'
```

0 s completado a las 21:01

Show all

Back

MobileNet V2 ImageNet (ILSVRC-2012-CLS)

Overall usage data
501.5k Downloads

Model formats

TF .JS (v1, default) .JS (v2, default) .JS (v3, default)

Warning: An updated version of this module is available at https://tfhub.dev/google/imagenet/mobilenet_v2_100_224/classification/5

TF2.0 Saved Model (v4) Usage data: 2.5k Downloads V4

Activities

Google Chrome

File Explorer

VS Code

Firefox

Spotify

Telegram

LibreOffice Writer

LibreOffice Calc

LibreOffice Impress

LibreOffice Draw

LibreOffice Base

LibreOffice Math

LibreOffice Science

LibreOffice Access

LibreOffice Database

LibreOffice Presentation

LibreOffice Spreadsheet

LibreOffice Graphics

LibreOffice Sound

LibreOffice Video

LibreOffice Animation

LibreOffice Font

LibreOffice Color

LibreOffice Style

LibreOffice Layout

LibreOffice Print

LibreOffice Help

LibreOffice About

LibreOffice License

LibreOffice Privacy

LibreOffice Security

LibreOffice Support

LibreOffice Feedback

LibreOffice News

LibreOffice Updates

LibreOffice Downloads

LibreOffice Links

LibreOffice Social

LibreOffice Legal

LibreOffice Privacy

LibreOffice Security

LibreOffice Support

LibreOffice Feedback

LibreOffice News

LibreOffice Updates

LibreOffice Downloads

LibreOffice Links

LibreOffice Social

LibreOffice Legal

Thu Oct 28 9:02 PM

es

100%

[INT] x Clusl x Copi x Clusl x Copi x Copi x PCA x Copi x Pima x Tens x Copi x Tens x Tens x

colab.research.google.com/drive/19YFy6QQ6QDFKEhVSpeUprPvao8hnV3F1#scrollTo=nuERg7MGz2OC

Search Star Camera Extensions Profile

Copia de TensorFlowHub_TheBasics_[MOJIX].ipynb

☆

Comentario Compartir Settings Profile

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda Se han guardado todos los cambios

+ Código + Texto

RAM Disco Editar

▼ Paso 3: Define la arquitectura de tu modelo

Revisa la documentación del modelo que escogiste

✓ 3 s

[3] IMAGE_SHAPE = (224, 224)

modelo = Sequential([

hub.KerasLayer(modelo_tfhub, input_shape=IMAGE_SHAPE+(3,))

])

▼ Paso 4: Escoge una imagen para realizar las predicciones

▶

imagen_test_url = 'https://live.staticflickr.com/5449/7440792338_eb509ba9da_b.jpg'

"https://boliviatravelsite.com/Images/Attractionphotos/illimani-02.jpg"

imagen_test_url

Vamos a descargar la imagen y vamos a ajustar sus dimensiones

[] imagen_test = tf.keras.utils.get_file("image"+str(np.random.randint(100))+'.jpg',imagen_test_url,)

imagen_test = Image.open(imagen_test).resize((IMAGE_SHAPE[0], IMAGE_SHAPE[1]))

✓ 3 s completado a las 21:02

archive (2).zip

Show all

colab.research.google.com/drive/19Yfy6QQ6QDFKEhVSpeUprPvao8hnV3F1#scrollTo=6r7Elzg_17Xe

Copia de TensorFlowHub_TheBasics_[MOJIX].ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda Se han guardado todos los cambios

Comentario

Compartir

Editar

+ Código + Texto

RAM Disco

[3] hub.KerasLayer(modelo_tfhub, input_shape=IMAGE_SHAPE+(3,))

Paso 4: Escoge una imagen para realizar las predicciones

[] imagen_test_url = 'https://live.staticflickr.com/5449/7440792338_eb509ba9da_b.jpg'
"https://boliviatravelsite.com/Images/Attractionphotos/illimani-02.jpg"
imagen_test_url

Vamos a descargar la imagen y vamos a ajustar sus dimensiones

imagen_test = tf.keras.utils.get_file("image"+str(np.random.randint(100))+'.jpg',imagen_test_url,)
imagen_test = Image.open(imagen_test).resize(IMAGE_SHAPE)
imagen_test

Vamos a ajustar los valores de la imagen

[] imagen_test = np.array(imagen_test)/255.0
imagen_test.shape

3 s completado a las 21:02


archive (2).zip

Show all

← → ↺

colab.research.google.com/drive/19Yfy6QQ6QDFKEhVSpeUprPvao8hnV3F1#scrollTo=6r7Elzg_17Xe

🔍 ☆ 📷 ⚙️ 👤 ⋮

 Copia de TensorFlowHub_TheBasics_[MOJIX].ipynb ☆

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda


Comentario


Compartir

⚙️ 👤

+ Código + Texto

✓ 0 s





RAM

Disco

Editar

↑ ↓ 🔗 💬 ⚙️ 📄 🗑️ ⋮

Vamos a ajustar los valores de la imagen

```
imagen_test = np.array(imagen_test)/255.0
imagen_test.shape
```

▼ Paso 5: Realizamos predicciones

Obtenemos la predicción bajo la estructura de las clases con las que ha sido entrenado el modelo

https://tfhub.dev/google/imagenet/mobilenet_v2_130_224/classification/4

✓ 0 s completado a las 21:03

📄 archive (2).zip

Show all



+ Código + Texto

✓ RAM
Disco

✎ Editar



▼ Paso 5: Realizamos predicciones

Obtenemos la predicción bajo la estructura de las clases con las que ha sido entrenado el modelo

https://tfhub.dev/google/imagenet/mobilenet_v2_130_224/classification/4

```
[7] prediccion = modelo.predict(imagen_test[np.newaxis, ...])  
prediccion.shape  
  
(1, 1001)
```

```
[8] prediccion  
  
array([[ 0.24092045, -0.31635118,  1.8853828 , ..., -1.0570533 ,  
        1.0253358 , -0.7496758 ]], dtype=float32)
```

```
clase_prediccion = np.argmax(prediccion)  
clase_prediccion
```

664

Descargamos las clases en formato de texto

```
[ ] labels_path = tf.keras.utils.get_file('ImageNetLabels.txt', 'https://storage.googleapis.com/download.tensorflow.org/data/ImageNetLabels.txt')
```

✓ 0 s completado a las 21:04



+ Código + Texto

```
1.0253358 , -0.7496758 ]], dtype=float32)
```

```
[9] clase_prediccion = np.argmax(prediccion)
     clase_prediccion

     664
```

Descargamos las clases en formato de texto

```
[10] labels_path = tf.keras.utils.get_file('ImageNetLabels.txt', 'https://storage.googleapis.com/download.tensorflow.org/data/ImageNetLabels.txt')
     imagenet_labels = np.array(open(labels_path).read().splitlines())

     Downloading data from https://storage.googleapis.com/download.tensorflow.org/data/ImageNetLabels.txt
     16384/10484 [=====] - 0s 0us/step
     24576/10484 [=====] - 0s 0us/step
```

Vamos a obtener el nombre de la clase de nuestra imagen

```
imagenet_labels[clase_prediccion]

'monastery'
```


Ruth Chirinos7:11 PM
Podemos grabar la sesión?

Nathaly Alarcon7:39 PM
<https://colab.research.google.com/drive/199MZvaBiKI3zm0-qyNcAS8GCU96IP26?usp=sharing>

Nathaly Alarcon8:04 PM
<https://colab.research.google.com/drive/1wPTTr22hDWloyU1Nn0bl2oxfmWwAjrARo?usp=sharing>

Nathaly Alarcon8:19 PM
<https://www.displayr.com/what-is-hierarchical-clustering/#:~:text=Hierarchical%20clustering%2C%20also%20known%20as,broadly%20similar%20to%20each%20other.>

Nathaly Alarcon8:22 PM
<https://realpython.com/k-means-clustering-python/>

<https://colab.research.google.com/drive/1LZ86ki2A0naIWHDnWuV2wBU1LcfV7Jlq?usp=sharing>

Nathaly Alarcon8:34 PM
https://colab.research.google.com/drive/12VL_fYaT04EGNKfHsQP-CrSvIOKwCPV4?usp=sharing
<https://www.kaggle.com/uciml/pima-indians-diabetes-database>

You8:46 PM
se puede saber cuales son esos componentes en el dataset (los nombres de las columnas)?

Aldo Gutierrez8:47 PM
osea son planos entre componentes Xs?