Machine Learning

K Vecinos Próximos (KNN)

Christian Oliva Moya
Luis Fernando Lago Fernández

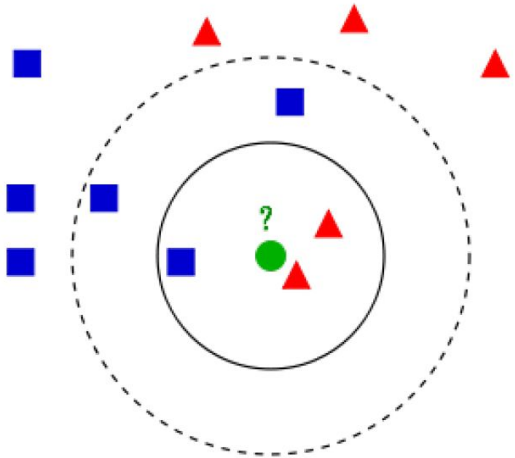
Introducción - K Vecinos Próximos (KNN)

K Vecinos Próximos (K-Nearest Neighbors, KNN) es un algoritmo de ML:

- Supervisado
- De clasificación
- No paramétrico

Además, es realmente intuitivo

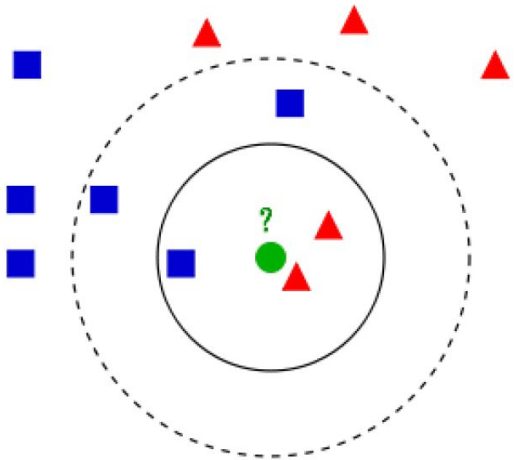
Utiliza la **proximidad a los vecinos** para hacer una predicción



Introducción - K Vecinos Próximos (KNN)

- Para clasificar un nuevo dato x_i hay que observar los K puntos más cercanos y contar cuántos son de la clase a predecir.

$$P(C_j|x_i) = \frac{K_{Cj}}{K}$$



En el ejemplo, el punto verde se clasificaría como ROJO

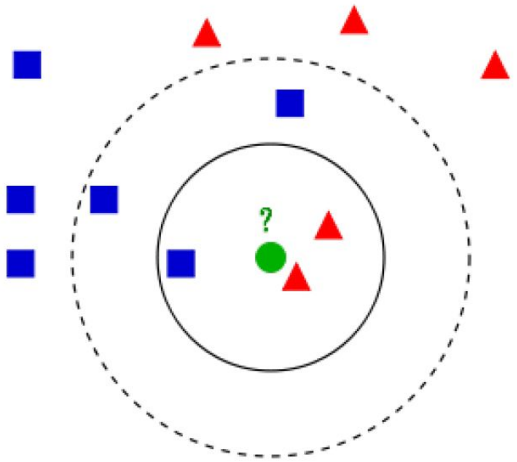
si K = 3, pero se clasificaría como AZUL si K = 5

Introducción - K Vecinos Próximos (KNN)

- Métrica de similitud típica: Distancia Euclídea

$$d(p, q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

- Donde *p* y *q* son dos puntos en un espacio de dimensión *n*



Introducción - K Vecinos Próximos (KNN)

- Consideraciones:
  - ¿Qué pasa si los atributos tienen diferente rango?
  - ¿Qué pasa si los atributos no son numéricos?
  - Es obligatorio definir el hiperparámetro K.

Observación

La mayoría de los algoritmos de ML necesitan una fase de pre-procesado que prepare las características para que sean útiles para discriminar entre clases

Scikit-Learn - Introducción

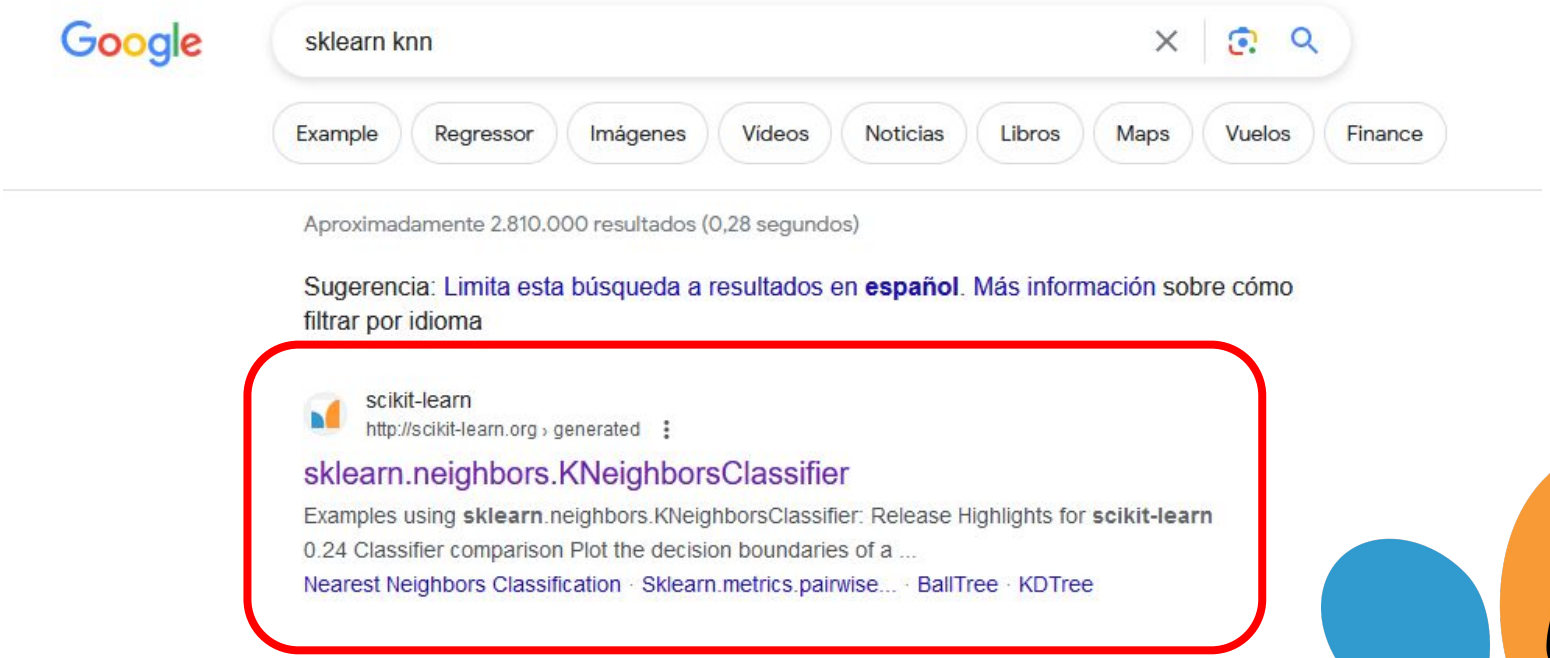
- **Sklearn** es una librería de ML con una amplia variedad de herramientas y algoritmos.

  Es realmente popular ya que:

  - Tiene una interfaz sencilla
  - Es directamente compatible con Numpy
  - Tiene una amplia documentación, incluyendo tutoriales y ejemplos
  - Tiene datasets de introducción realmente interesantes

Scikit-Learn - Introducción

- Si estás buscando un algoritmo de Machine Learning, busca en google:



Scikit-Learn - Introducción

- ¿Cómo es la documentación?

```
from sklearn.neighbors import KNeighborsClassifier
```

¿Cómo importar la clase?

sklearn.neighbors.KNeighborsClassifier

class `sklearn.neighbors.`**`KNeighborsClassifier`**(*n_neighbors=5, *, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None*) [source]

Classifier implementing the k-nearest neighbors vote.

Read more in the User Guide.

Parameters:

n_neighbors : *int, default=5*
Number of neighbors to use by default for `kneighbors` queries.

weights : *{'uniform', 'distance'}, callable or None, default='uniform'*
Weight function used in prediction. Possible values:

- 'uniform' : uniform weights. All points in each neighborhood are weighted equally.
- 'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
- [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

Scikit-Learn - Introducción

- ¿Cómo es la documentación?

Cabecera del constructor

sklearn.neighbors.KNeighborsClassifier

class `sklearn.neighbors.`**`KNeighborsClassifier`**(*n_neighbors=5, *, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None*) [source]

Classifier implementing the k-nearest neighbors vote.

Read more in the User Guide.

Parameters:

n_neighbors : *int, default=5*
Number of neighbors to use by default for `kneighbors` queries.

weights : *{'uniform', 'distance'}, callable or None, default='uniform'*
Weight function used in prediction. Possible values:

- 'uniform' : uniform weights. All points in each neighborhood are weighted equally.
- 'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
- [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

Scikit-Learn - Introducción

- ¿Cómo es la documentación?

sklearn.neighbors.KNeighborsClassifier

class `sklearn.neighbors.`**`KNeighborsClassifier`**(*n_neighbors=5, *, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None*) [source]

Classifier implementing the k-nearest neighbors vote.

Definición de todos los parámetros

Read more in the User Guide

Parameters:

n_neighbors : *int, default=5*
Number of neighbors to use by default for `kneighbors` queries.

weights : *{'uniform', 'distance'}, callable or None, default='uniform'*
Weight function used in prediction. Possible values:

- 'uniform' : uniform weights. All points in each neighborhood are weighted equally.
- 'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
- [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

scikit learn

Scikit-Learn - Introducción

- Siempre nos dan algún ejemplo y la definición de cada uno de los métodos

Examples

```
>>> X = [[0], [1], [2], [3]]
>>> y = [0, 0, 1, 1]
>>> from sklearn.neighbors import KNeighborsClassifier
>>> neigh = KNeighborsClassifier(n_neighbors=3)
>>> neigh.fit(X, y)
KNeighborsClassifier(...)
>>> print(neigh.predict([[1.1]]))
[0]
>>> print(neigh.predict_proba([[0.9]]))
[[0.666... 0.333...]]
```

Methods

`fit`(X, y)	Fit the k-nearest neighbors classifier from the training dataset.
`get_metadata_routing`()	Get metadata routing of this object.
`get_params`([deep])	Get parameters for this estimator.
`kneighbors`([X, n_neighbors, return_distance])	Find the K-neighbors of a point.
`kneighbors_graph`([X, n_neighbors, mode])	Compute the (weighted) graph of k-Neighbors for points in X.
`predict`(X)	Predict the class labels for the provided data.
`predict_proba`(X)	Return probability estimates for the test data X.
`score`(X, y[, sample_weight])	Return the mean accuracy on the given test data and labels.
`set_params`(**params)	Set the parameters of this estimator.
`set_score_request`(*[, sample_weight])	Request metadata passed to the `score` method.

Scikit-Learn - Introducción

- Siempre nos dan algún ejemplo y la definición de cada uno de los métodos

Examples

```
>>> X = [[0], [1], [2], [3]]
>>> y = [0, 0, 1, 1]
>>> from sklearn.neighbors import KNeighborsClassifier
>>> neigh = KNeighborsClassifier(n_neighbors=3)
>>> neigh.fit(X, y)
KNeighborsClassifier(...)
>>> print(neigh.predict([[1.1]]))
[0]
>>> print(neigh.predict_proba([[0.9]]))
[[0.666... 0.333...]]
```

fit entrena el modelo

Methods

fit(X, y)	Fit the k-nearest neighbors classifier from the training dataset.
get_metadata_routing()	Get metadata routing of this object.
get_params([deep])	Get parameters for this estimator.
kneighbors([X, n_neighbors, return_distance])	Find the K-neighbors of a point.
kneighbors_graph([X, n_neighbors, mode])	Compute the (weighted) graph of k-Neighbors for points in X.
predict(X)	Predict the class labels for the provided data.
predict_proba(X)	Return probability estimates for the test data X.
score(X, y[, sample_weight])	Return the mean accuracy on the given test data and labels.
set_params(**params)	Set the parameters of this estimator.
set_score_request(*[, sample_weight])	Request metadata passed to the `score` method.

Scikit-Learn - Introducción

- Siempre nos dan algún ejemplo y la definición de cada uno de los métodos

Examples

```
>>> X = [[0], [1], [2], [3]]
>>> y = [0, 0, 1, 1]
>>> from sklearn.neighbors import KNeighborsClassifier
>>> neigh = KNeighborsClassifier(n_neighbors=3)
>>> neigh.fit(X, y)
KNeighborsClassifier(...)
>>> print(neigh.predict([[1.1]]))
[0]
>>> print(neigh.predict_proba([[0.9]]))
[[0.666... 0.333...]]
```

predict devuelve la opción más probable
predict_proba devuelve las probabilidades

Methods

fit(X, y)	Fit the k-nearest neighbors classifier from the training dataset.
get_metadata_routing()	Get metadata routing of this object.
get_params([deep])	Get parameters for this estimator.
kneighbors([X, n_neighbors, return_distance])	Find the K-neighbors of a point.
kneighbors_graph([X, n_neighbors, mode])	Compute the (weighted) graph of k-Neighbors for points in X
predict(X)	Predict the class labels for the provided data.
predict_proba(X)	Return probability estimates for the test data X.
score(X, y[, sample_weight])	Return the mean accuracy on the given test data and labels.
set_params(**params)	Set the parameters of this estimator.
set_score_request(*[, sample_weight])	Request metadata passed to the score method.

Scikit-Learn - K Vecinos Próximos (KNN)

Recordatorio: K Vecinos Próximos (K-Nearest Neighbors, KNN) es un algoritmo de ML:

- Supervisado
- De clasificación
- **No paramétrico**

¿Cómo es que tiene un fit?

Methods

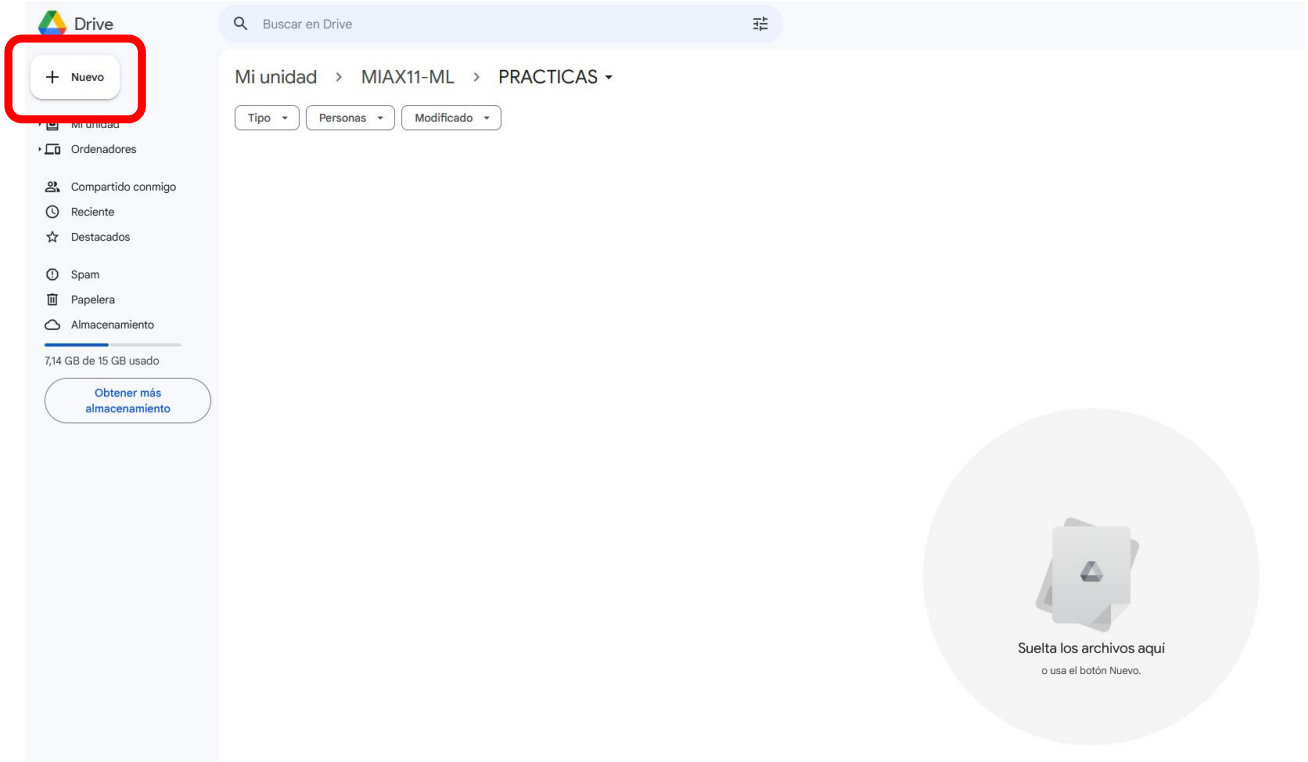
`fit`(X, y)	Fit the k-nearest neighbors classifier from the training dataset.
`get_metadata_routing`()	Get metadata routing of this object.
`get_params`([deep])	Get parameters for this estimator.
`kneighbors`([X, n_neighbors, return_distance])	Find the K-neighbors of a point.
`kneighbors_graph`([X, n_neighbors, mode])	Compute the (weighted) graph of k-Neighbo
`predict`(X)	Predict the class labels for the provided d
`predict_proba`(X)	Return probability estimates for the tes
`score`(X, y[, sample_weight])	Return the mean accu give
`set_params`(**params)	Set the parameters
`set_score_request`(*[, sample_weight])	Request metadata p

Google Colaboratory

- Plataforma en línea de Google para ejecutar código Python
- Diseñada para ser una herramienta colaborativa que permite compartir proyectos de ML en un entorno basado en el navegador.
- Tienes acceso a potentes recursos en la nube: GPUs y TPUs para Deep Learning
- No requiere instalación más allá de instalar la herramienta en Google Drive
- Se basa en los cuadernos Jupyter

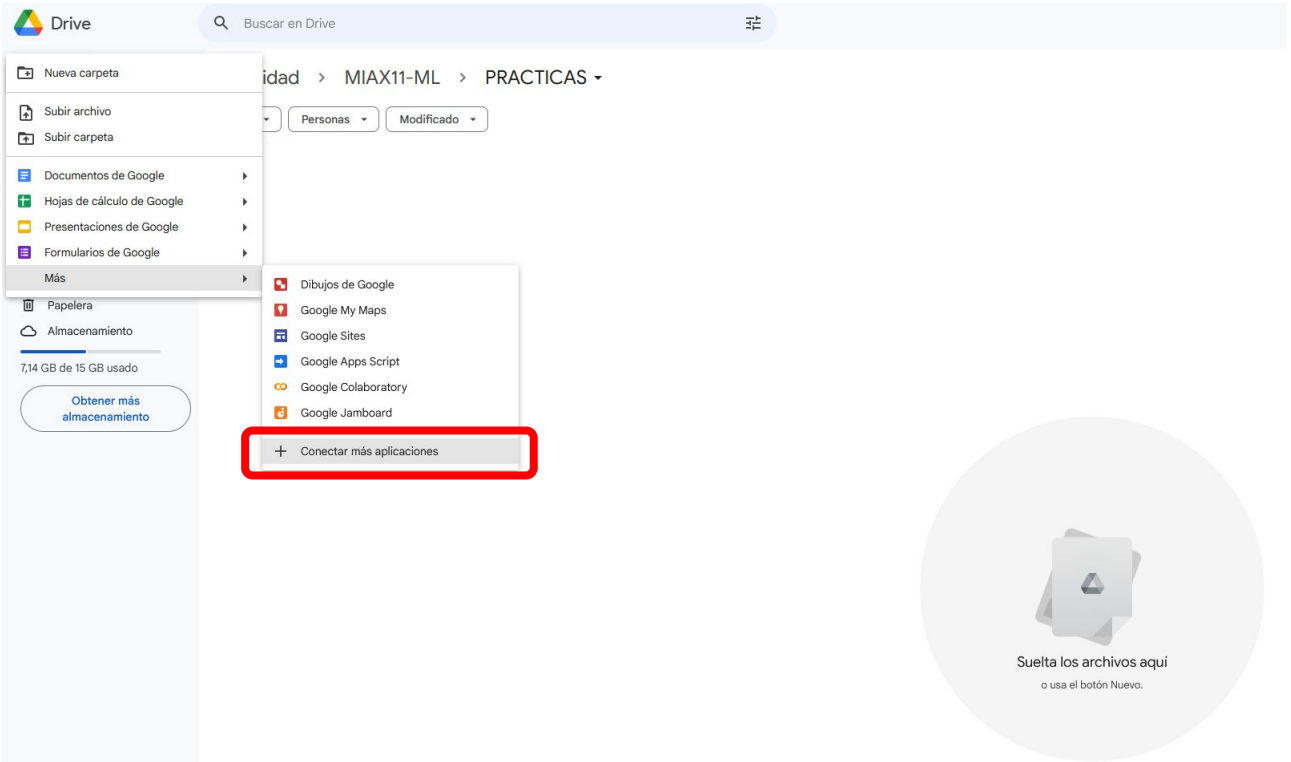
Google Colaboratory

- ¿Cómo instalarlo?



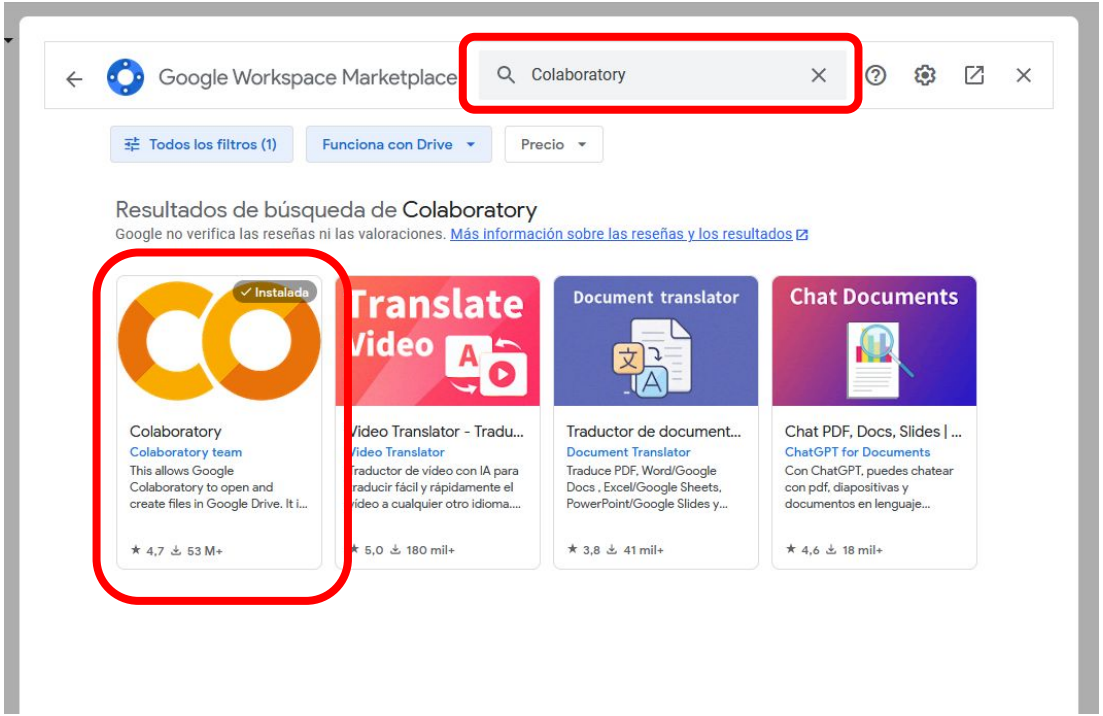
Google Colaboratory

- ¿Cómo instalarlo?



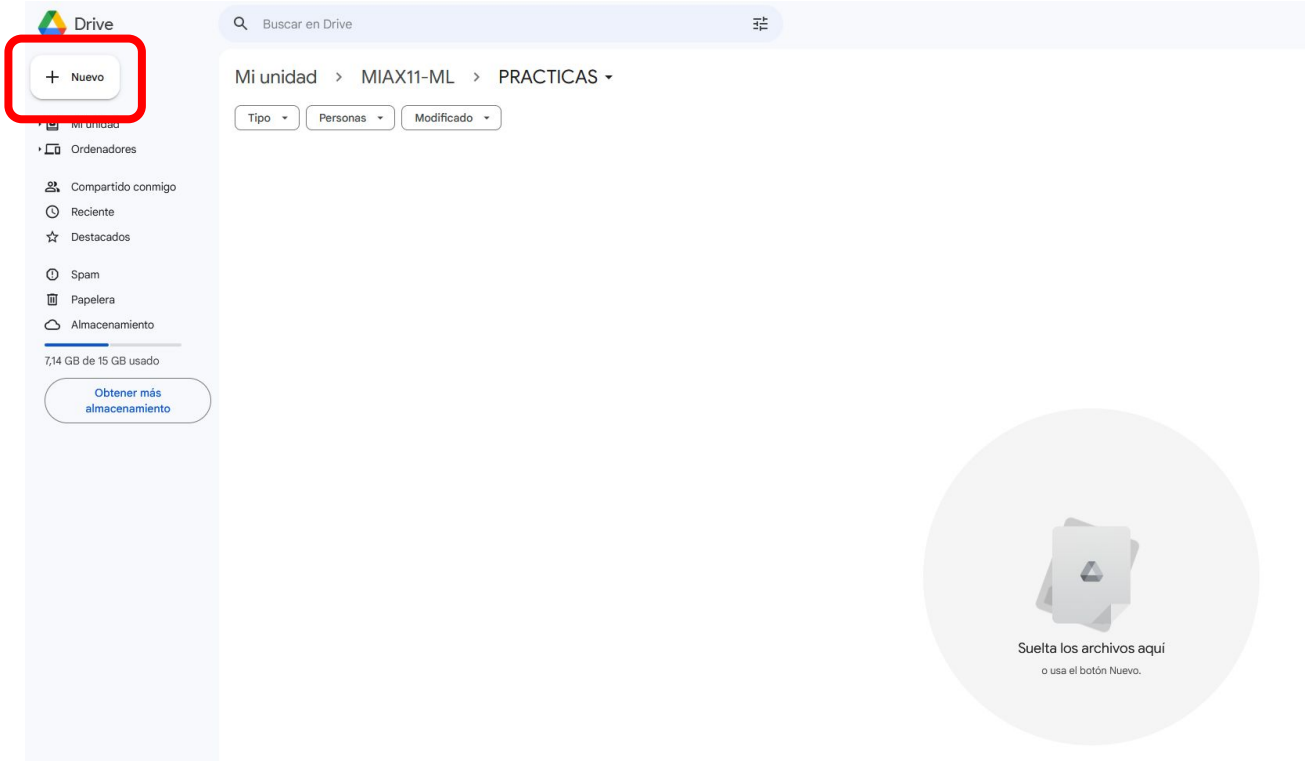
Google Colaboratory

- ¿Cómo instalarlo?



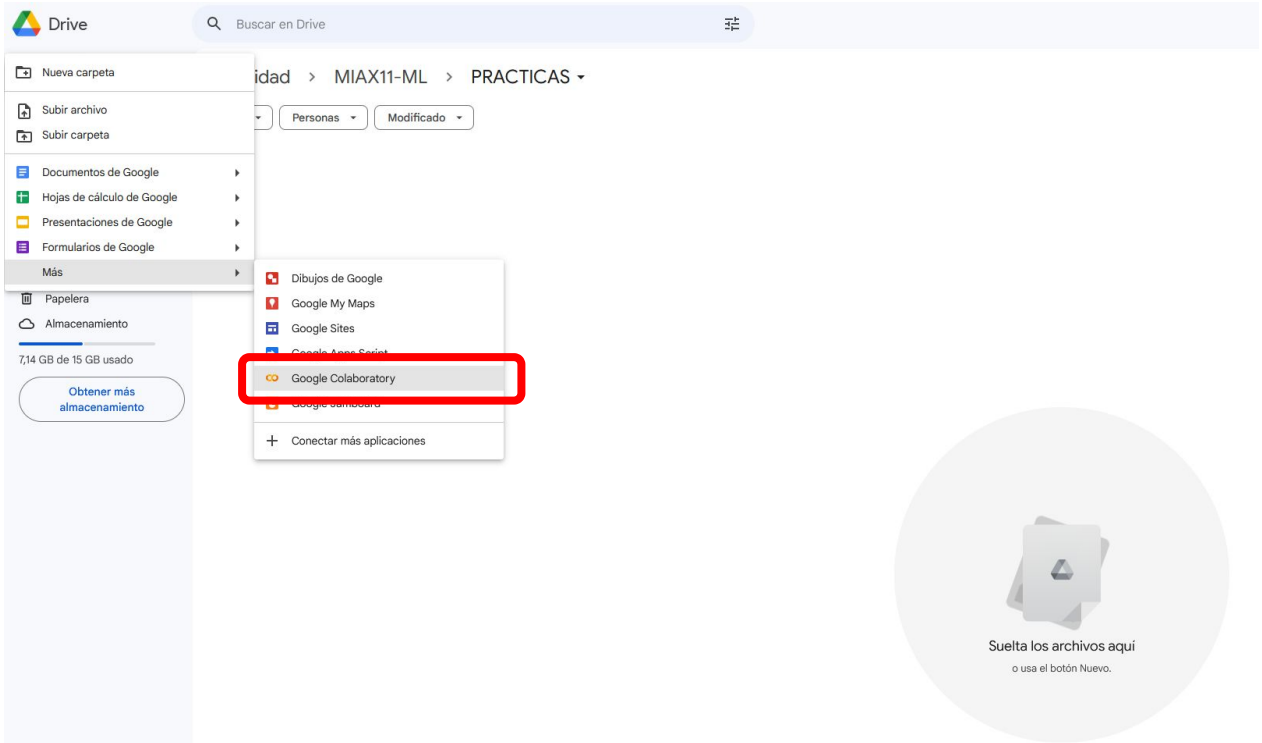
Google Colaboratory

- ¿Cómo crear un nuevo notebook?



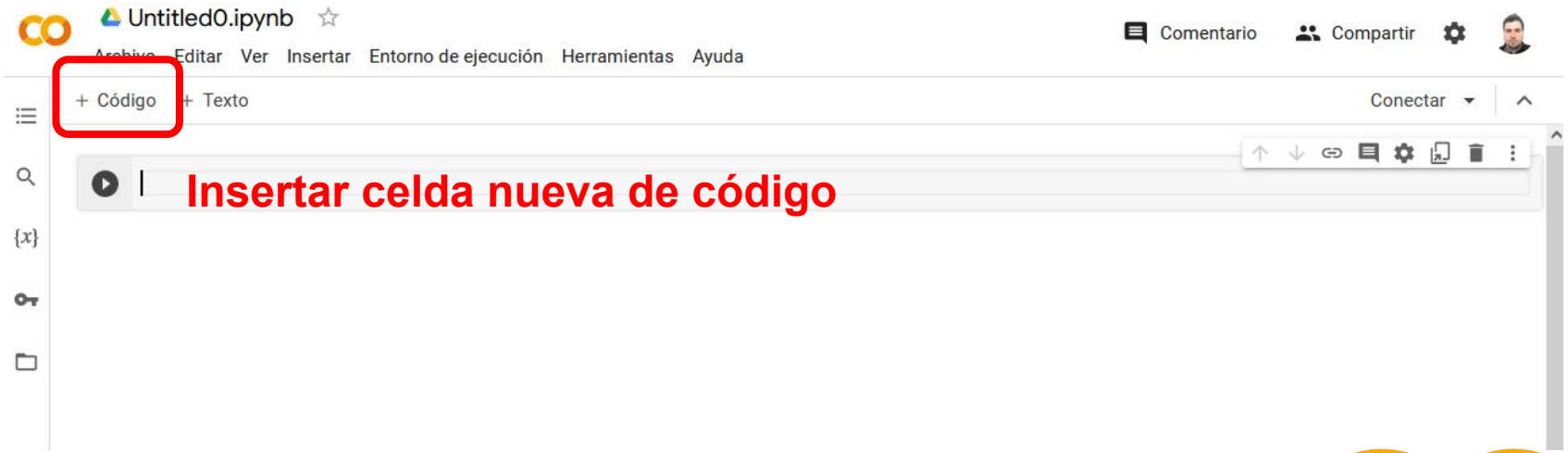
Google Colaboratory

- ¿Cómo crear un nuevo notebook?



Google Colaboratory

- ¿Cómo es Google Colaboratory?

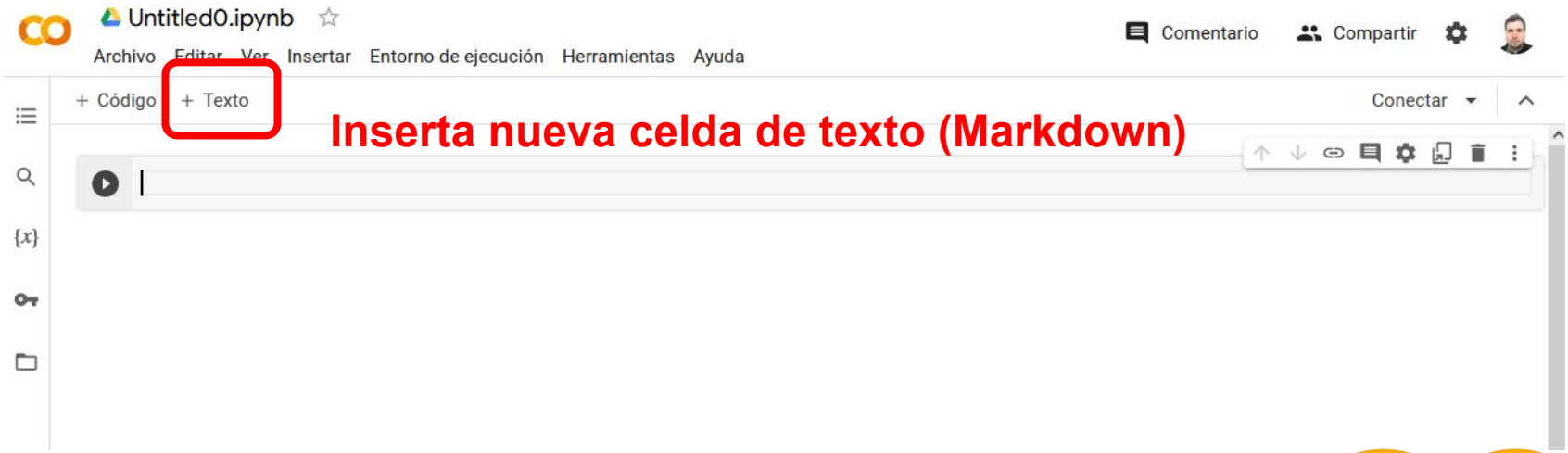


Google Colaboratory

- ¿Cómo es Google Colaboratory?

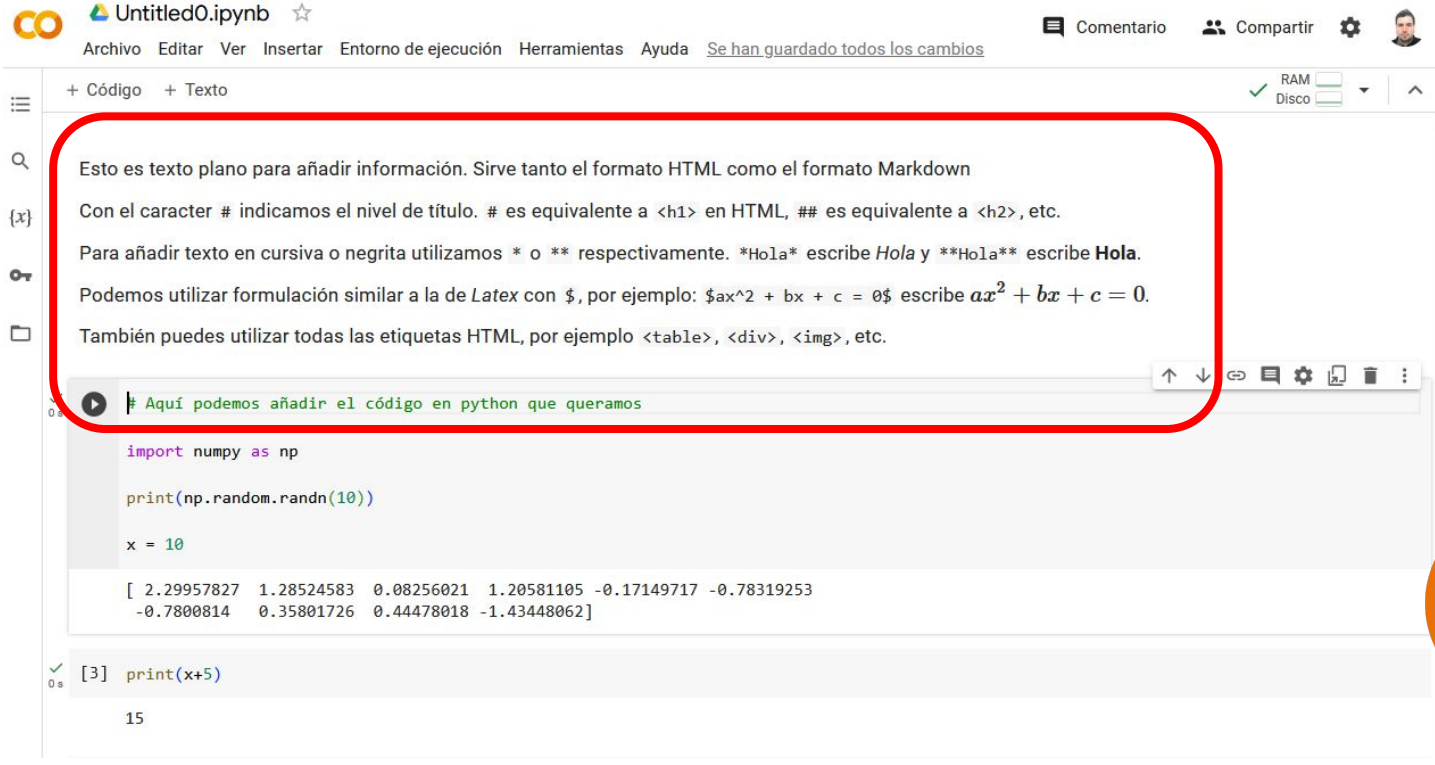
Google Colaboratory

- ¿Cómo es Google Colaboratory?



Google Colaboratory

- ¿Cómo es Google Colaboratory?



Introducción - K Vecinos Próximos (KNN)

- Vamos al notebook para probar el KNN

1. Descargad el notebook del material de clase que os damos.
2. Subid el notebook a vuestro Drive
3. Abridlo con Google Colaboratory:

El notebook es: ***2_1_knn.ipynb***