

SISTEMAS INTELIGENTES

GUIA DE LABORATORIOS

6/18/24

Table of Contents

1.	Lab01 Entorno: Configuración del entorno de desarrollo	2
1.1.	Instalación de Anaconda	2
1.1.1.	Introducción a Anaconda.....	2
1.1.2.	Procedimiento de instalación	2
1.2.	Entornos virtuales en Anaconda.....	4
1.2.1.	Introducción a Entornos Virtuales	4
1.2.2.	Creando un Entorno Virtual.....	4
1.3.	Configurando jupyter Notebook en el environment	5
1.4.	Caso de prueba	5
2.	Lab01 EDA (Exploratory Data Analysis)	7

1. Lab01 Entorno: Configuración del entorno de desarrollo

1.1. Instalación de Anaconda

1.1.1. Introducción a Anaconda

Anaconda es una distribución de Python para computación científica. Es compatible con Linux, macOS y Windows. Proporciona una gestión simplificada de paquetes y entornos, y maneja fácilmente los problemas de instalación cuando el sistema tiene múltiples versiones de Python y paquetes de terceros. Anaconda utiliza la herramienta/comando conda o pip para implementar la gestión de paquetes y entornos. También viene con Python y herramientas relacionadas. Anaconda es una herramienta de Python para análisis de big data a nivel empresarial. Contiene más de 720 paquetes de ciencia de datos de código abierto, incluyendo visualización de datos, aprendizaje automático y aprendizaje profundo. Puede usarse para análisis de datos, big data y campos de IA.




Anaconda proporciona las siguientes características para desarrolladores:

- Con Anaconda, no es necesario instalar Python. Solo necesitas seleccionar la versión de Python al descargar Anaconda.
- Con Anaconda, solo necesitas agregar un entorno virtual a Anaconda cuando se requieren diferentes frameworks para soportar el desarrollo. Puedes realizar el desarrollo en diferentes entornos sin preocuparte por problemas de compatibilidad. También puedes configurar entornos para proyectos especiales para facilitar la gestión.

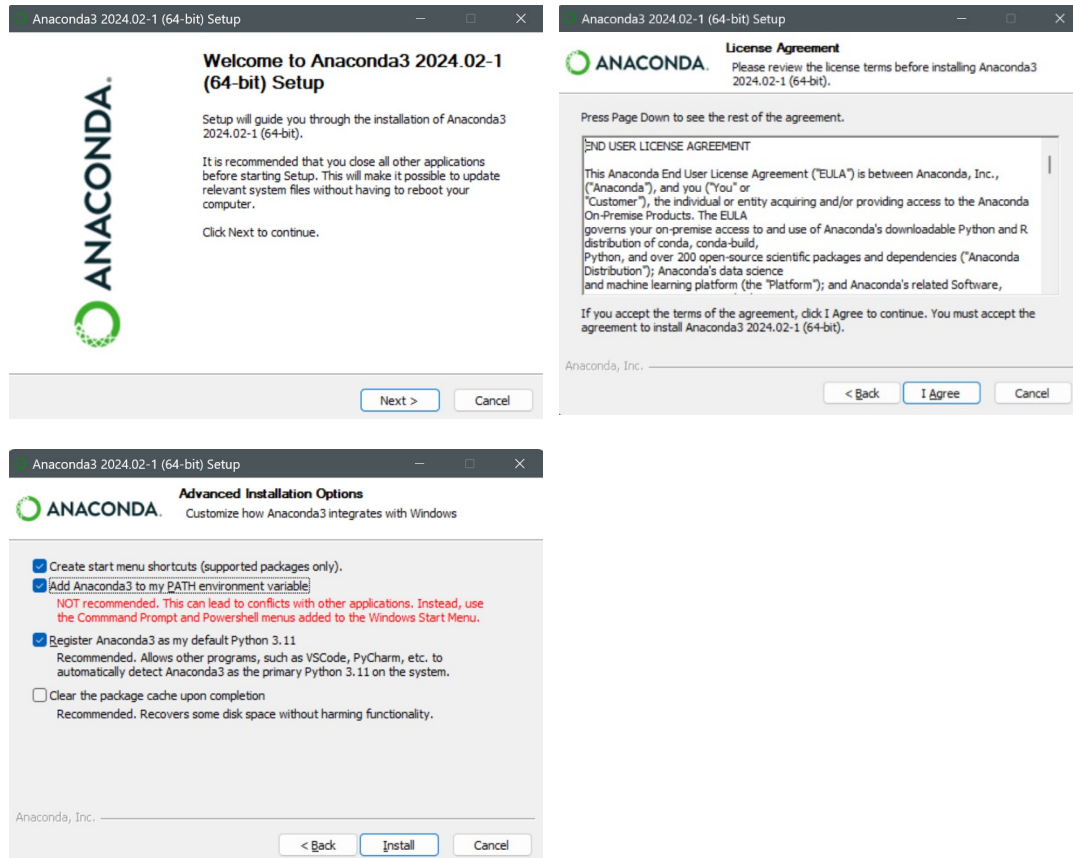
1.1.2. Procedimiento de instalación

Paso 1 Descargar Anaconda: <https://www.anaconda.com/download> selecciona la versión para Windows, macOS o Linux. Para este caso, selecciona Windows.

Anaconda Installers

 Windows Python 3.11 ↓ 64-Bit Graphical Installer (904.4M)	 Mac Python 3.11 ↓ 64-Bit (Apple silicon) Graphical Installer (697.4M) ↓ 64-Bit (Apple silicon) Command Line Installer (700 M) ↓ 64-Bit (Intel chip) Graphical Installer (728.7M) ↓ 64-Bit (Intel chip) Command Line	 Linux Python 3.11 ↓ 64-Bit (x86) Installer (997.2M) ↓ 64-Bit (AWS Graviton2 / ARM64) Installer (798.5M) ↓ 64-bit (Linux on IBM Z & LinuxONE) Installer (91.8M)
---	--	---

Paso 2 Instalar Anaconda: Haz doble clic en el archivo descargado Anaconda3-2024.02-1-Windows-x86_64. En el cuadro de diálogo que se muestra, como se ve en la Figura, haz clic en Next->I Agree->seleccionar Just Me -> Next ->Next->Seleccionar Add Anaconda 3 to my PATH environment variable->Install

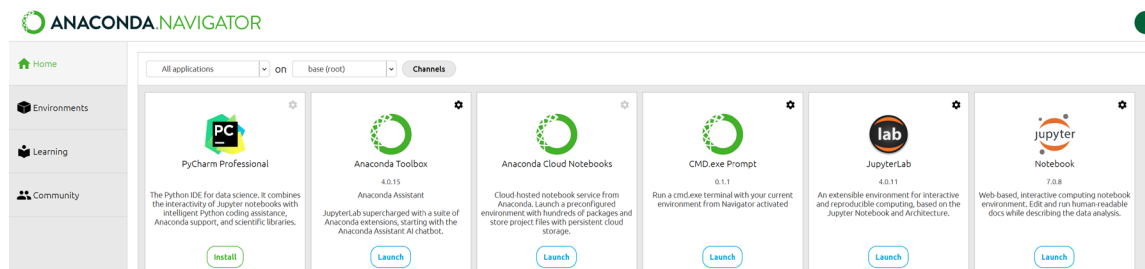


Cuando la instalación se ha completado, hacer click en Next->Next->Finish.

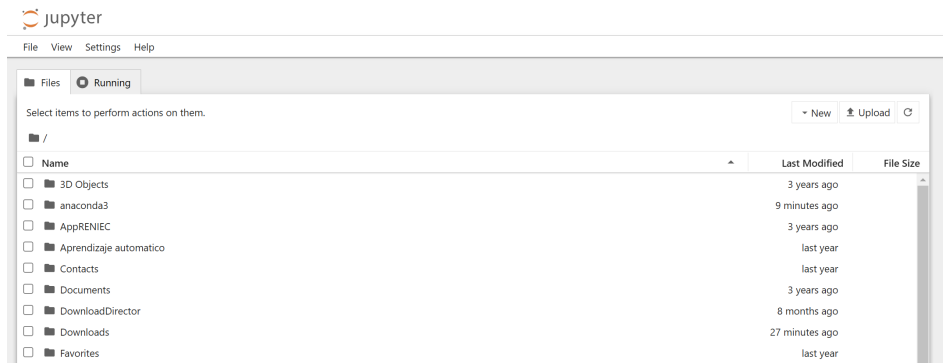
Una vez ejecutado Anaconda, de ser necesario procedemos a actualizarlo. Luego lanzamos nuevamente el navegador.

Si deseamos podemos registrarnos en Anaconda Cloud para poder acceder a Anaconda Assistant.

En la ventana Anaconda Navigator, seleccionamos jupyter Notebook, que es el entorno que vamos a utilizar.



Ahora ya tenemos acceso al entorno jupyter Notebook



1.2. Entornos virtuales en Anaconda

1.2.1. Introducción a Entornos Virtuales

Anaconda es popular debido a los entornos virtuales. Permite crear múltiples entornos independientes de Python en un host para que los desarrolladores los utilicen. Cuando las dependencias entre diferentes módulos de Python están desordenadas o existen aplicaciones de diferentes marcos de desarrollo, los entornos virtuales mantienen estas dependencias en sandboxes separadas para que los usuarios puedan cambiar entre ambas aplicaciones fácilmente y hacerlas funcionar.

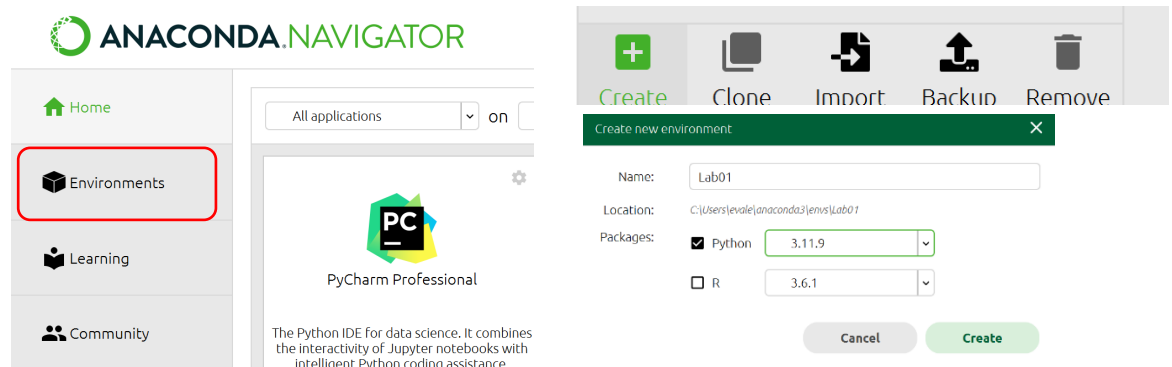
Nota: Los módulos en los entornos virtuales son independientes entre sí. Si has instalado TensorFlow en el entorno A y deseas usar TensorFlow en el entorno B, necesitas instalar TensorFlow en el entorno B. Esto también se aplica a Spyder y Jupyter Notebook.

1.2.2. Creando un Entorno Virtual

Paso 1 Iniciar Anaconda Navigator, desde el menú de inicio de Windows.

Paso 2 Crear un entorno virtual

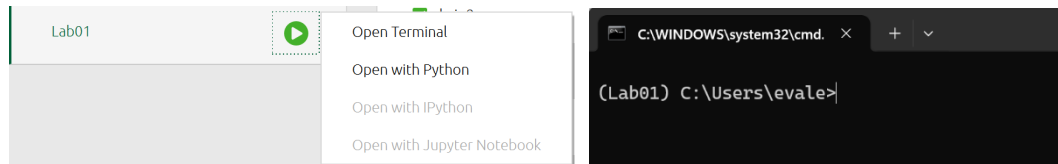
Click en Environments en el menú principal, tal como se muestra en la figura, y luego hacer click en Create



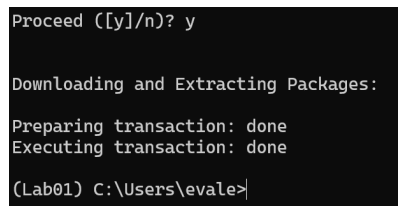
Paso 3 En la caja de dialogo que se visualiza, configurar el nombre, y seleccionar Python y la versión requerida para el entorno de desarrollo, por ejemplo 3.11 o 3.12, luego hacer click en Create.

1.3. Configurando jupyter Notebook en el environment

Paso 1 Abrir una terminal, verificamos que esta activo el entorno



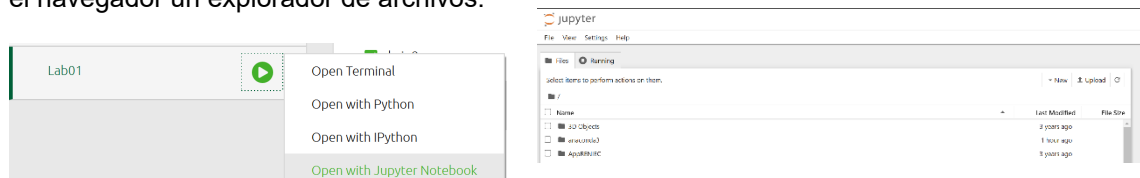
Paso 2 Instalamos jupyter, desde la terminal ejecutamos el comando: conda install jupyter, posteriormente aceptamos la instalación, presionando la tecla “Y”



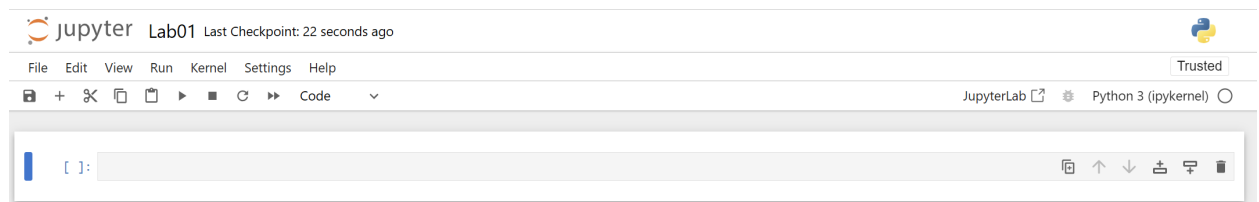
Para verificar que ya esta instalado Jupyter, podemos ver activado la opción “Open with Jupyter Notebook” desde nuestro entorno.

1.4. Caso de prueba

Paso 1 Iniciar Jupyter, utilizando el environment creado en los pasos previos, se ejecuta en el navegador un explorador de archivos.



Paso 2 Crear un proyecto Jupyter, primero debemos crear una carpeta de trabajo, para la asignatura vamos a crear la carpeta: MachineLearning/GrupoX luego hacemos click en New->Notebook seleccionamos el Kernel, y luego podemos modificar el nombre del notebook a Lab01 Entorno, ahora ya estamos listos para validar nuestra configuración.



Probamos el Notebook

```
[7]: # Instalamos las librerías necesarias
# Instala TensorFlow utilizando pip
# %pip en un Jupyter Notebook es una práctica recomendada, ya que garantiza que la instalación se realice en el entorno Python
# que está utilizando el notebook. Esto es especialmente útil cuando se trabaja con múltiples entornos de Anaconda.
%pip install tensorflow

Requirement already satisfied: tensorflow in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (2.16.1)
Requirement already satisfied: tensorflow-intel==2.16.1 in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (from tensorflow) (2.16.1)
Requirement already satisfied: absl-py==1.0.0 in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (from tensorflow-intel==2.16.1->tensorflow) (2.1.0)
Requirement already satisfied: astunparse==1.6.0 in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (from tensorflow-intel==2.16.1->tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers==23.5.26 in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (from tensorflow-intel==2.16.1->tensorflow) (24.3.25)
Requirement already satisfied: gast!=0.5.0,!0.5.1,!0.5.2,>=0.2.1 in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (from tensorflow-intel==2.16.1->tensorflow) (0.5.4)
Requirement already satisfied: google-pasta==0.1.1 in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (from tensorflow-intel==2.16.1->tensorflow) (0.2.0)
Requirement already satisfied: h5py>=3.10.0 in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (from tensorflow-intel==2.16.1->tensorflow) (3.11.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (from tensorflow-intel==2.16.1->tensorflow) (18.1.1)
Requirement already satisfied: ml-dtypes==0.3.1 in c:\users\evale\anaconda3\envs\lab01\lib\site-packages (from tensorflow-intel==2.16.1->tensorflow) (0.3.2)

[4]: #Verificamos la instalacion
import tensorflow as tf
print(tf.__version__)

2.16.1

[10]: # Import modules.
import tensorflow as tf
import numpy as np
# Create two constant tensors.
mat1 = tf.constant(np.array([[1.0, 2.0]]))
mat2 = tf.constant(np.array([[2.],[2.]])
# Create a multiplication operation.
mul = tf.matmul(mat1,mat2)
print(mul)

tf.Tensor([[6.]], shape=(1, 1), dtype=float64)
```

Paso 3 Verificando a través de un ejemplo usando Keras

```
[13]: # Probando Tensor flow y keras
import tensorflow as tf
mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

[14]: model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(10, activation='softmax')
])

[15]: model.compile(optimizer='adam',
    loss=['sparse_categorical_crossentropy'],
    metrics=['accuracy'])
model.fit(x_train, y_train, epochs=5)

Epoch 1/5
1875/1875 — 4s 2ms/step - accuracy: 0.8558 - loss: 0.4904
Epoch 2/5
1875/1875 — 3s 1ms/step - accuracy: 0.9555 - loss: 0.1521
Epoch 3/5
1875/1875 — 3s 1ms/step - accuracy: 0.9668 - loss: 0.1104
Epoch 4/5
1875/1875 — 3s 1ms/step - accuracy: 0.9733 - loss: 0.0879
Epoch 5/5
1875/1875 — 3s 1ms/step - accuracy: 0.9764 - loss: 0.0758

[15]: <keras.src.callbacks.history.History at 0x17b3d86e790>

[16]: model.evaluate(x_test, y_test)
313/313 — 0s 1ms/step - accuracy: 0.9744 - loss: 0.0859

[16]: [0.07293939590454102, 0.9789999723434448]
```

Este experimento verifica si el entorno de experimentación está configurado correctamente.

Si no se reportan errores durante la ejecución del código, el entorno de experimentación se ha configurado correctamente.

2. Lab01 EDA (Exploratory Data Analysis)

La EDA (análisis exploratorio de datos) nos permite analizar los datos que tenemos en un database de manera intuitiva y gráfica, y así poder comprender mejor sus características o features.

La definición formal es: En estadística, el análisis exploratorio de datos es un enfoque para analizar conjuntos de datos para resumir sus principales características, a menudo con métodos visuales. Se puede usar o no un modelo estadístico, pero principalmente EDA es para ver qué nos pueden decir los datos más allá del modelado formal o la tarea de prueba de hipótesis.

Podemos por lo tanto tener unos insight que nos serán útiles luego tanto para preparar los datos para las fases sucesivas (removiendo las irregularidades) y tener una visión aproximativa de los mismos.

Basándonos en este análisis podemos empezar a tomar algunas decisiones en la estructuración de los datos y en la definición de los modelos a utilizar:

- Si lo hacemos bien podemos ahorrarnos mucho trabajo
- Si lo hacemos mal arriesgamos a perder información valiosa

En EDA tenemos 5 fases principales:

1. Recompilación y carga de datos
2. Limpieza de datos
3. Análisis univariado
4. Análisis bivariado
5. Análisis multivariado