Usaremos:

Herramienta	¿La usaremos?
Subsistema Linux para Windows (WSL)	No
Jupyter Server en WSL	No
ANACONDA 3	Sí (**ver siguiente apartado**)
Make Sense	No
Label Studio	Sí
KNIME	No
ROBOBO Sim	Probablemente
Servicios virtualizados / servidores	No

Instalación de Python sobre Windows

Dado que en los cursos vamos a usar, como sistema operativo anfitrión, Windows; tenemos que adaptarnos e instalar Python para el SO de Microsoft. Seguidamente se muestra el enlace para su descarga: Python 3.11 para windows

Esta estrategia, diferente a la inicialmente planteada, implica que no necesitamos Anaconda, ni crear un *script* para arrancar el servidor Jupyter. La otra cara es que - en principio - no tendremos entornos virtuales, así que un error con las versiones (o debido a ellas) nos llevará al *pozo de la desesperación*. Sin embargo también podríamos desinstalarlo y volver a instalarlo para partir de cero.

Los pasos son:

- ☐ Descargar e instalar Python para Windows (ver enlace anterior).
- ☐ Iniciar una terminal de línea de comandos (`cmd`).
- ☐ Podemos comprobar que Python está instalado: `python --version`
- ☐ Ejecutar los siguientes comandos:
  - ☐ `python -m pip install --upgrade pip`
  - ☐ `python -m pip install ipykernel jupyter-server ml4teens`

Subsistema Linux para Windows (WSL)

Este elemento nos permite tener un subsistema Linux en **Windows 11**.

Antes de empezar:

- ☐ Ir a **Activar o Desactivar las Características de Windows**.
- ☐ Ejecutar la aplicación; pedirá permisos de administrador.
- ☐ En el interfaz de la aplicación buscar (en la parte de abajo) la característica denominada "*Subsistema de Windows para Linux*". Activar estas características, seguir los pasos y finalmente pulsar aceptar para cerrar la aplicación. Si no aparece esta característica, hay que actualizar Windows 11 a la última versión. Si aún así sigue sin aparecer, ponerse en contacto con la persona o empresa encargada del mantenimiento.

Instalación de un sistema operativo Linux hospedado en Windows 11.

- ☐ Abrir una terminal (comando **`cmd`**).
- ☐ Introducir (si es la primera vez) el siguiente comando: **`wsl –install -d Ubuntu`**
- ☐ Seguidamente se abrirá otro terminal, que nos preguntará por:
    - En nombre de usuario: pon el mismo que usas en Windows.
    - La contraseña: pon la misma que la de windows.
    - Repetir la contraseña: la misma del paso anterior.
- ☐ Si sale todo bien, ya deberíamos tener hospedada una distribución Linux (Ubuntu) en nuestro SO. Ahora hay que actualizarla ejecutando los siguientes comandos:
    - **`sudo apt update`**
    - **`sudo apt upgrade`**
- ☐ Al finalizar sin errores escribiremos **`exit`** en el terminal y pulsaremos enter/intro.

Ahora tenemos un SO Linux hospedado en Windows. Para iniciar una terminal en este SO debemos de invocar el comando **`wsl`**. Sólo así arrancamos una consola en la que podemos introducir comandos GNU/Linux.

Jupyter Server en WSL

Servicio backend para ejecutar notebooks Jupyter.

Debido a una serie de complicaciones que genera Windows con Anaconda, la mejor opción - por ahora - para instalar un *backend* para Jupyter en WSL.

- ☐ Arrancar la consola: comando **`wsl`**
- ☐ Ejecutar **`sudo apt install python3-pip`** y seguir las indicaciones.
- ☐ Ejecutar **`sudo apt install python3-venv`** y seguir las indicaciones.
- ☐ Ejecutar **`python3 -m venv .JS`**
- ☐ Ejecutar **`ln -s .JS/bin/activate activate`**
- ☐ Ejecutar **`source activate`**
- ☐ Ejecutar **`pip install jupyter-server`**
- ☐ Ejecutar **`pip install ipykernel`**
- ☐ Ejecutar **`sudo apt install git-all`** y seguir las indicaciones.

Crear un fichero de texto llamado **`jupyter_server.py`**, que será el que inicie el servidor local:

```
#!/bin/env python3

import subprocess;

def start_server(token):
    colab="https://colab.research.google.com";
    command=[ "jupyter-server",
              "--no-browser",
             f"--IdentityProvider.token={token}",
              "--port=8888",
             f"--ServerApp.allow_origin='{colab}'",
            ];
    subprocess.call(command);

if __name__ == "__main__":
   start_server("1234Abcd");
```

Substituye el *token* "**`1234Abcd`**" por otro propio que sólo conste de números y letras. Este *token* hace las veces de contraseña.

Ahora puedes levantar un servidor *jupyter* local con el siguiente comando desde la consola:

```
python3 jupyter_server.py
```

ANACONDA 3

https://anaconda.com/download

Aplicación para la instalación de python en un ordenador, herramientas *open source* y tecnologías asociadas.

Pasos para su instalación en Windows.

- ☐ Descargar desde el navegador.
- ☐ Una vez descargado, ejecutarlo.
- ☐ Ventana de bienvenida: pulsar "*next*".
- ☐ Aceptar, si es el caso, el acuerdo de licencia: pulsar "*I agree*".
- ☐ Seleccionar el tipo de instalación: seleccionar "*just for me*".
- ☐ Seleccionar el lugar de la instalación: el valor por defecto es recomendable.
- ☐ Opciones de instalación avanzadas: los valores por defecto seleccionados son correctos, excepto el último "*clear the package cache upon completion*". Por defecto se encuentra desactivado. Activarlo. Seguidamente pulsar "*install*". La instalación suele ocupar 5 GB, comprobar que se tiene el espacio de almacenamiento suficiente. La instalación suele tardar, dependiendo de la velocidad de conexión que se posea.
- ☐ Al finalizar la instalación, pulsar "*next*" y seguidamente "*finish*".
- ☐ Se abre una pestaña en un navegador web y se inicia la aplicación "*anaconda navigator*".
- ☐ Es posible que se indique, inmediatamente después de lanzarse "*anaconda navigator*", una ventana de diálogo que nos ofrezca actualizar la aplicación. Pulsar sí, y volver a indicarlo cuando se nos pregunte si queremos cerrar la aplicación (necesario para actualizarla). Seguir las indicaciones para actualizarla.

Una vez instalado Anaconda 3 junto con su Navigator (su GUI), sigue los pasos necesarios para crear tu primer **entorno virtual**:

- ☐ Abre el "*anaconda navigator*" y crea un entorno virtual llamado "JupyterServer".
    - ☐ Para ello ve a "Environments", en la segunda columna a la izquierda, abajo, pulsa el icono "Create", dale el nombre "JupyterServer" y pulsa el botón para crearlo.
- ☐ Ábrelo en modo terminal ("*Open terminal*", pulsando en la flecha verde del entorno).

- [ ] Ejecutar **`pip install jupyter-server`**
- [ ] Ejecutar **`pip install ipykernel`**

Crear un fichero de texto llamado **`jupyter_server.py`**, que será el que inicie el servidor local; te recomiendo que lo hagas en tu escritorio. Introduce el siguiente texto en él:

```
import subprocess;

def start_server(token):
    colab="https://colab.research.google.com";
    command=[ "jupyter-server",
              "--no-browser",
             f"--IdentityProvider.token={token}",
              "--port=8888",
             f"--ServerApp.allow_origin='{colab}'",
            ];
    subprocess.call(command);

if __name__ == "__main__":
   start_server("1234Abcd");
```

Substituye el *token* "**`1234Abcd`**" por otro propio que sólo conste de números y letras. Este *token* hace las veces de contraseña.

Ahora puedes levantar un servidor *jupyter* local con el siguiente comando desde la consola:

```
python3 jupyter_server.py
```

Make Sense

https://www.makesense.ai

Aplicación web local para etiquetar imágenes. No es necesario instalarla.

Label Studio

Servicio web para etiquetado de datos.

Para instalar la aplicación es necesario tener iniciada la aplicación "*anaconda navigator*".

- [ ] En la aplicación, seleccionar en la columna izquierda la opción "*Environments*".
- [ ] Aparecerá a su derecha otra columna con los entornos creados, por defecto siempre hay uno llamado "*base*". Sin embargo vamos a crear otro: Pulsa en la parte de abajo de esa columna, el icono con nombre "*create*".

- ☐ Aparecerá una ventana de diálogo, invitándonos a introducir un nombre (introduce "*LabelStudio*") y dos opciones. Asegúrate que las opciones son "python" activado (con la versión que indica por defecto) y "R" desactivado: pulsa "create".
- ☐ Una vez creado, seleccionamos el nuevo entorno y pulsamos sobre la flecha que tiene el nombre a su derecha.
- ☐ Se abrirá un menú contextual cuya primera opción es "*Open Terminal*": pulsamos en la opción.
- ☐ Se abrirá una ventana para la introducción de comandos.
- ☐ Introduciremos el siguiente comando: **`pip install label-studio`** y pulsaremos enter/intro.
- ☐ Al finalizar la instalación introduciremos el comando **`exit`** y pulsaremos enter/intro.

Para ejecutar esta aplicación, abriremos la terminal del entorno y ejecutaremos: **`label-studio`** y pulsaremos enter/intro. Tras unos segundos se abrirá el navegador predeterminado y/o se creará una nueva pestaña en el navegador abierto con la interfaz de la aplicación.

KNIME

https://www.knime.com/downloads

Aplicación para la creación y ejecución de soluciones de análisis de datos y *machine learning*.

- ☐ Ir a la página web de descarga.
- ☐ Pulsar el botón "*download*".
- ☐ Se nos muestra un formulario de registro; no es necesario rellenarlo, pero sí debemos activar la opción "*I have read and accept the Privacy Policy*". Una vez seleccionado, en la parte de abajo del formulario podemos ver otro botón con el nombre: "download". Pulsarlo.
- ☐ En el siguiente paso nos invitan a seleccionar la instalación más adecuada. Seleccionar "*KNIME Analytics Platform for Windows (installer)*"; inmediatamente se iniciará la descarga.
- ☐ Al finalizar la descarga, ejecutar la aplicación descargada. Inmediatamente se nos mostrarán dos opciones: escoger "*install only for me*".
- ☐ En la siguiente ventana se nos pedirá aceptar la licencia; si es el caso, aceptar y pulsar "*next*".

- [ ] Se nos sugiere un lugar en el espacio de almacenamiento: aceptar pulsando “*next*”.
- [ ] Se nos sugiere el nombre de la carpeta que contendrá la aplicación: pulsar “*next*”.
- [ ] Se nos sugiere un conjunto de opciones. Todas son recomendables: pulsar “*next*”.
- [ ] Se nos sugiere un parámetro avanzado (máximo de memoria usara por KNIME): aceptar pulsando “*next*”.
- [ ] La siguiente sugerencia es importante. Se nos da la opción de dar soporte a longitudes de paths largas, debemos aceptar: pulsa “*next*”.
- [ ] Se nos sugiere añadir una excepción a Windows Defender (el antivirus): pulsa “*next*”.
- [ ] Finalmente se nos muestra un resumen de la instalación que se va a realizar: pulsa “*install*”.

ROBOBO Sim

https://aiplus-app.therobobоproject.com/RobоboSimulator-2.13.2-CE-Win64.zip

Simulador de mundos virtuales para un simpático robot.

- [ ] Descargar el .zip del enlace anterior.
- [ ] Descomprimir su contenido, al hacerlo creará una carpeta.
- [ ] Mueve dicha carpeta al escritorio.

Dentro de dicha carpeta se encuentra el fichero ejecutable que inicia la aplicación. Es recomendable crear un acceso directo a dicho ejecutable y moverlo al escritorio.

Para poder usar Robobo desde una aplicación escrita en Python, necesitamos instalar la librería que nos da acceso al simulador (o al robot real).

- [ ] Obraremos de forma parecida a la instalación de LabelStudio, creando un nuevo entorno con el nombre “Robobo”.
- [ ] Una vez creado el nuevo entorno, lo seleccionamos y abrimos un terminal.
- [ ] introducimos lo siguiente: **pip install `robobopy`** y pulsamos enter/intro.
- [ ] Al finalizar la instalación, escribimos **`exit`** y pulsamos enter/intro.

Servicios virtualizados / servidores

En windows 11:

- [ ] Abrir el **Panel de Control**.

- ☐ Seleccionar **Programas**.
- ☐ Seleccionar **Activar o Desactivar las Características de Windows**.
- ☐ Buscar la característica '**Hyper-v**' y seleccionarla.
- ☐ Pulse "*aceptar*" para actualizar la configuración.
- ☐ Si te pide reiniciar, hazlo.

Con esto la máquina está lista para soportar la virtualización de servicios.