

Machine Learning con Python

Intel® oneAPI

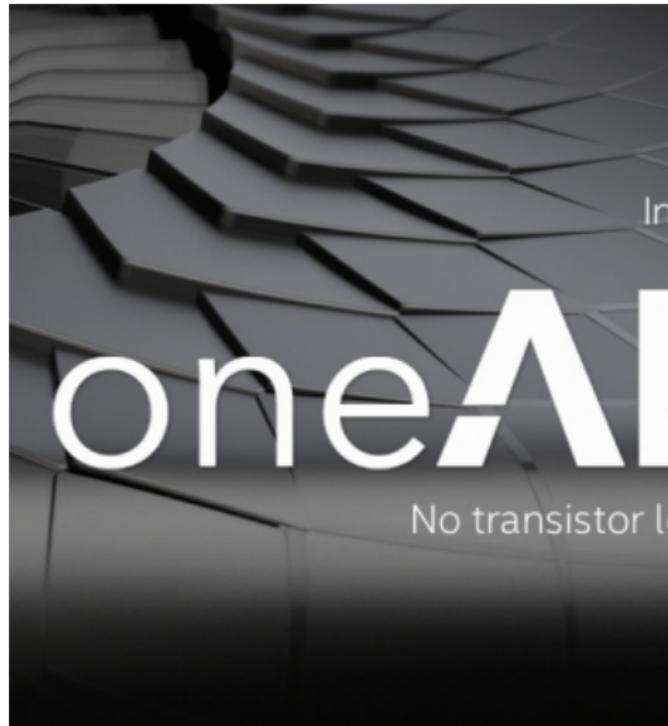
CGS

13 de julio de 2023

- “Machine Learning Using oneAPI”, <https://www.intel.com/content/www/us/en/developer/tools/oneapi/training/machine-learning-using-oneapi.html>
- “Intel® oneAPI Programming Guide”,
<https://www.intel.com/content/www/us/en/develop/documentation/oneapi-programming-guide/top.html>

Outline

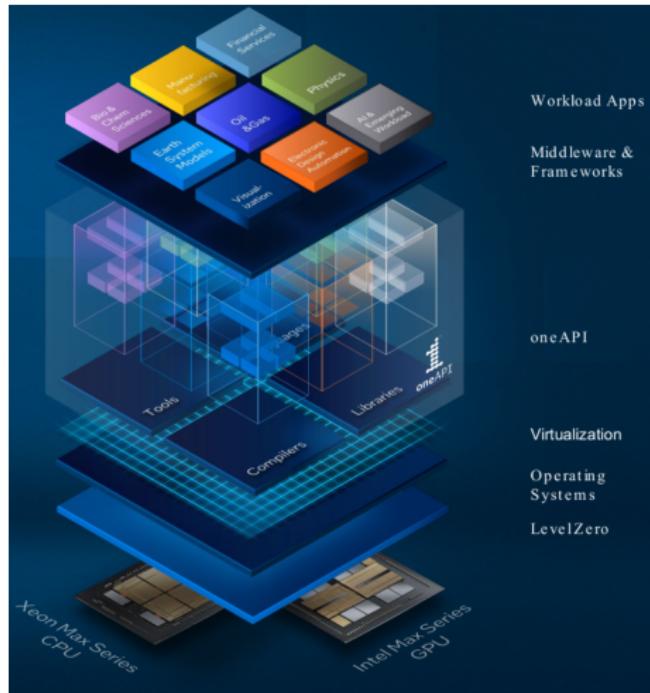
- 1 Introducción
- 2 Python en oneAPI
- 3 Pandas
- 4 Numpy
- 5 Machine-Learning en Python
- 6 Data Parallel
- 7 Otros



Introducción

Introducción

- Modelo de programación unificado: diversas arquitecturas
- Lenguaje y bibliotecas optimizados
- Rendimiento equivalente lenguaje nativo de alto nivel
- Basado en estándares de la industria y especificaciones abiertas
- Compatible con los modelos de programación HPC existentes



Introducción

- Un lenguaje basado en estándares: C++ y SYCL
- Potentes API para acelerar funciones de dominio específico

Soluciones a proveedor único

- Estándar abierto para promover el apoyo de la comunidad y la industria
- Permite la reutilización de código en diferentes arquitecturas y proveedores



Intel oneAPI AI Analytics Toolkit

- Acelera el flujo de trabajo desde un extremo al otro para aplicaciones IA y analítica de datos mediante librerías optimizadas para arquitecturas Intel
- ¿Para quién es interesante?
 - Desde científicos de datos, investigadores en IA, desarrolladores de aplicaciones IA y ML...
- Beneficios
 - Rendimiento en aplicaciones de DL desde el entrenamiento e inferencia: frameworks optimizados para archs Intel
 - Aceleraciones en aplicaciones de analítica de datos y ML intensivas en cómputo basadas en paquetes **python**

Intel oneAPI AI Analytics Toolkit

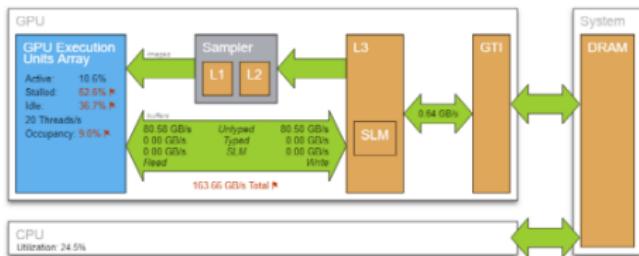
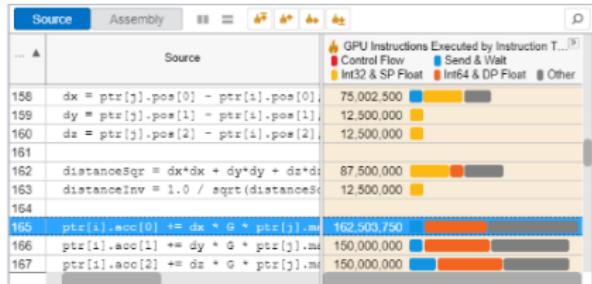
- Acelera el flujo de trabajo en desarrollos de IA y analítica de datos desde un extremo al otro mediante librerías optimizadas para arquitecturas Intel
- ¿Para quién es interesante?
 - Desde científicos de datos, investigadores en IA, desarrolladores de aplicaciones IA y ML...
- Beneficios: analítica de datos y ML intensivas en cómputo basadas en paquetes **python**

The screenshot shows the "What's Inside" section of the toolkit. It is organized into three main categories: DEEP LEARNING, MACHINE LEARNING, and DATA ANALYTICS. Under DEEP LEARNING, it lists Intel Optimization for TensorFlow, Intel Optimization for PyTorch, Model Zoo for Intel Architecture, and Intel Low Precision Optimization Tool. Under MACHINE LEARNING, it lists Intel Extension for Scikit-learn and Intel-optimized XGBoost. Under DATA ANALYTICS, it lists Intel Distribution of Modin and OmniSci Back End. Below these, a section for Core Python Libraries (Intel Distribution for Python) includes Intel-Opt NumPy, Intel-Opt SciPy, Intel-Opt Numba, Intel-Opt Pandas, and DPPY. At the bottom, there are sections for Samples and End2End Workloads, featuring icons for CPU and Future GPU Accelerators, with a note about Supported Hardware Architectures.

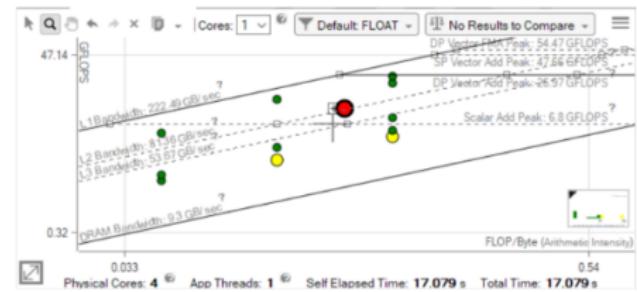
Intel® VTune™ Profiler

- Analiza Data Parallel C++ (DPC++)
- Ajusta para CPU, GPU & FPGA

- Optimiza para todos los aceleradores soportados
- Amplio rango de perfilados de rendimiento
 - CPU, GPU, FPGA, threading, memoria, cache, storage...
- Soportado por los lenguajes más comunes
 - DPC++, C, C++, Fortran, Python...



- Offload Advisor
 - Estima el rendimiento alcanzado en acelerador (offload)
- Roofline Analysis
 - Optimiza la codificación tanto en memoria como cómputo para CPU/GPU
- Vectorization Advisor
 - Explota y optimiza la vectorización
- Threading Advisor
 - Añade multihilos a las aplicaciones



- Un entorno disponible para desarrollar, probar y ejecutar sus cargas de trabajo en una amplia gama de CPU, GPU y FPGA de Intel utilizando el software de oneAPI
- <https://software.intel.com/en-us/devcloud/oneapi>
- Sin Descargas, sin comprar HW, sin actualizar SW

Use Intel oneAPI Toolkits

Learn Data Parallel C++

Evaluate Workloads

Build Heterogenous Applications

Prototype your project

Cuenta en DevCloud

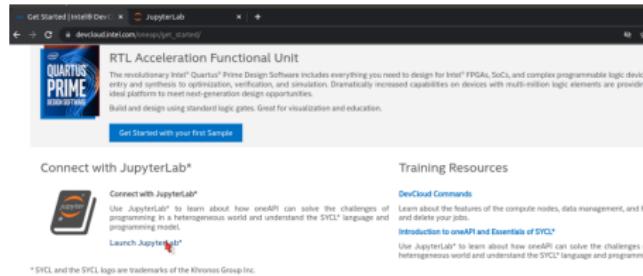
- El Intel® DevCloud for oneAPI es un espacio de desarrollo **gratuito** para que la comunidad de desarrolladores puedan programar aplicaciones
 - Múltiples **hw**:
 - **CPUs**: desktop *i9-11900* y servidor tipo Xeon diferentes arquitecturas (Skylake, Ice Lake, Sapphire Rapids)
 - **GPUs**: integradas UHD Intel® Core™ Gen9 y Gen11
 - **FPGAs**: Arria 10 y Stratix 10
 - **sw**: oneAPI divididos en **Toolkits**
 - Compiladores: C/C++ y Fortran
 - Herramientas de perfilado: VTune, Advisor, GDB
 - Librerías optimizadas: oneMKL, oneDPL, oneVPL, oneDNN...
- Solicitud de cuenta gratuita [rellenando formulario](#)
 - o bien en la web del Intel® DevCloud for oneAPI en la opción **Enroll**
 - **Importante** usar correo de UCM porque tiene una duración de uso mayor
 - Se recibirá un correo electrónico con instrucciones de uso

Pasos

- ① Acceder a la web del Intel DevCloud oneAPI
- ② Creación de nueva cuenta
- ③ Rellenar Datos personales
- ④ Aceptar términos y condiciones
- ⑤ Consultar instrucciones en correo electrónico
- ⑥ Acceso al DevCloud

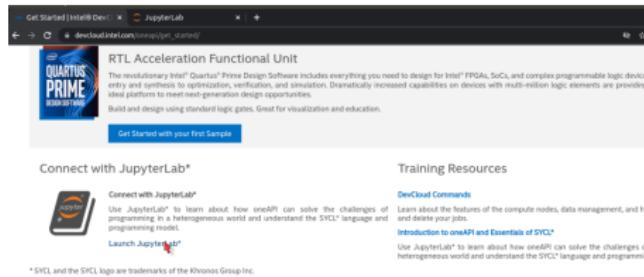
Jupyter-Notebooks

- La más sencilla es abrir un cuaderno de Jupyter
 - ① Una vez logeado en la web del **Intel® DevCloud for oneAPI** en la opción **Sign In** (esquina superior derecha)
 - ② Ir a la opción “**Get Started**” en la banda superior azul
 - ③ Clicar sobre “**Launch JupyterLab**” en la parte inferior izquierda o en el [atajo](#)



Entorno Jupyter

- El Intel® DevCloud for oneAPI contiene un entorno JupyterLab



- En la parte de la izquierda tiene un navegador de ficheros del usuario
 - Como funcionalidad útil, se pueden arrastrar fichero del equipo del *host* y automáticamente se llevan al DevCloud sin necesidad de hacer un sftp
- En la parte de la derecha contiene las principales aplicaciones disponibles:
 - Notebook o cuaderno de Jupyter** que usaremos en el taller para ilustrar el funcionamiento del “Data Parallel C++”
 - Consola o terminal** para interactuar con el sistema

Hands-on

① Conéctate al Intel DevCloud

- O sino funciona puedes instalar el [Intel® Distribution for Python](#)
- Instalar el entorno [Jupyter](#)

② Vamos a clonar todo el repositorio del taller

```
user@system:~$ git clone https://github.com/garsanca/oneAPI_ML_Jul23.git
Cloning into 'oneAPI_ML_Jul23'...
remote: Enumerating objects: 105, done.
remote: Counting objects: 100% (105/105), done.
remote: Compressing objects: 100% (73/73), done.
remote: Total 105 (delta 34), reused 91 (delta 29), pack-reused 0
Receiving objects: 100% (105/105), 22.45 MiB | 21.69 MiB/s, done.
Resolving deltas: 100% (34/34), done.
```

③ Vamos a abrir el primer cuaderno de Jupyter relacionado con el [uso](#) [DevCloud e introducción a Jupyter-Notebook](#)



Python en oneAPI

Distribución de Python

- Distribución optimizada en [plataformas Intel](#)
- Rendimiento cercano al código nativo optimizado para computación científica y HPC
 - Soporte hardware mediante librerías como oneMKL o oneDNN
 - Soporte de las últimas optimizaciones para CPU y GPU
- También disponible en el Toolkit [Intel® AI Analytics Toolkit \(AI Kit\)](#)



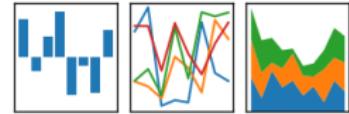
Pandas

Qué son

- Librería para computación con datos estructurados
 - Tipos mixtos en un tabla permitidos
- Se pueden nombrar columnas y filas de datos
- Agregación avanzada de datos y funciones estadísticas

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



Estructuras de datos

- Vectores de 1D: **Series**
- Arrays 2D...: **DataFrame**

pandas_series.py

```
import pandas as pd
step_data = [3620, 7891, 9761,
             3907, 4338, 5373]
step_counts = pd.Series(step_data, name='steps')
print(step_counts)
```

Terminal #1

```
user@host:~/ $ python panda_creation.py
0    3620
1    7891
2    9761
3    3907
4    4338
5    5373
Name: steps, dtype: int64
```

Hands-on

- ① Conéctate al Intel DevCloud
- ② Vamos a abrir el segundo cuaderno de Jupyter relacionado con el [uso Pandas](#)
- ③ Te animamos a que pongas en práctica el contenido con los [ejercicios](#) y puedes consultar también las [soluciones](#)



Numpy

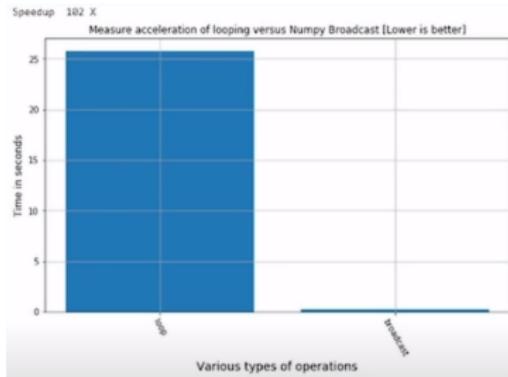
- Ventajas

- Es rápido para desarrollar ideas
- Se puede codificar casi todo lo imaginable
- Desarrolla un proyecto rápidamente... se pueden encontrar ejemplos en cualquier sitio
- Fácil: sin tipado (dinámico) lo que hace la programación más sencilla
- Rápido: gran cantidad de librerías disponibles y fáciles de instalar
- Aplicaciones de IA: sencillo de lograr portabilidad de los modelos desarrollados entre arquitecturas

- ... pero es **lento**
 - Tareas de bajo nivel repetitivas
 - Bucles largos
 - Bucles anidados
- Pero hay formas de mitigar los problemas
 - Librerías optimizadas para las arquitecturas Intel
 - NumPy es un ejemplo
 - Otras librerías están también optimizadas

Vectorización

- La explotación de la vectorización **no es una teoría**
 - Grandes aceleraciones son posibles
 - Altamente recomendable utilizar librerías optimizadas en Intel oneAPI como **NumPy**, **SciPy** y otras
 - Explotar la bondad Numpy (paralelismo inherente) mediante el uso de las librerías optimizadas en oneAPI
- Ej: 100x de aceleración empleando Numpy *broadcasting* respecto al código descrito como bucle



Numpy (vectorización)

- Disponible como paquete de [python](#)
- Versión optimizada incluida en oneAPI
 - Última versión disponible en el [Intel® AI Analytics Toolkit \(AI Kit\)](#)

Como funciona

- Reemplazando los bucles explícitos asociados a las operaciones entre vectores/arrays: SIMD
- En general operaciones entre arrays pueden lograrse entre 1 o 2 órdenes de magnitud de velocidad que las equivalentes *python puras*

Hands-on

- ① Conéctate al Intel DevCloud
- ② Vamos a abrir el tercer cuaderno de Jupyter relacionado con el [uso Numpy](#)
- ③ Te animamos a que pongas en práctica el contenido con los [ejercicios](#) y puedes consultar también las [soluciones](#)



Machine-Learning en Python

- Machine Learning o **Aprendizaje automático**: ciencia que hace que los computadores “aprendan”
 - A partir de unos datos de entrada
 - El computador extra patrones de los diferentes tipos de datos
 - Aprende las reglas del problema

Python scikit-learn

- Scikit-learn es una librería de aprendizaje automático para el lenguaje de programación Python
 - Integración con otras bibliotecas de Python para el procesamiento de datos eficiente NumPy y Pandas o SciPy
 - Numpy: para el procesamiento y operaciones eficiente sobre forma vectorial o matricial
 - Pandas: para manipulación de datos en forma de tablas o series
 - SciPy: algoritmos de optimización, integración, interpolación...

- Desarrollada originalmente como parte del proyecto Google Summer of Code en el año 2007
 - Mantenido y desarrollado por una comunidad de desarrolladores
- Amplia variedad de herramientas para la minería y el análisis de datos
 - Algoritmos para clasificación, regresión, clustering y reducción de dimensionalidad

Características

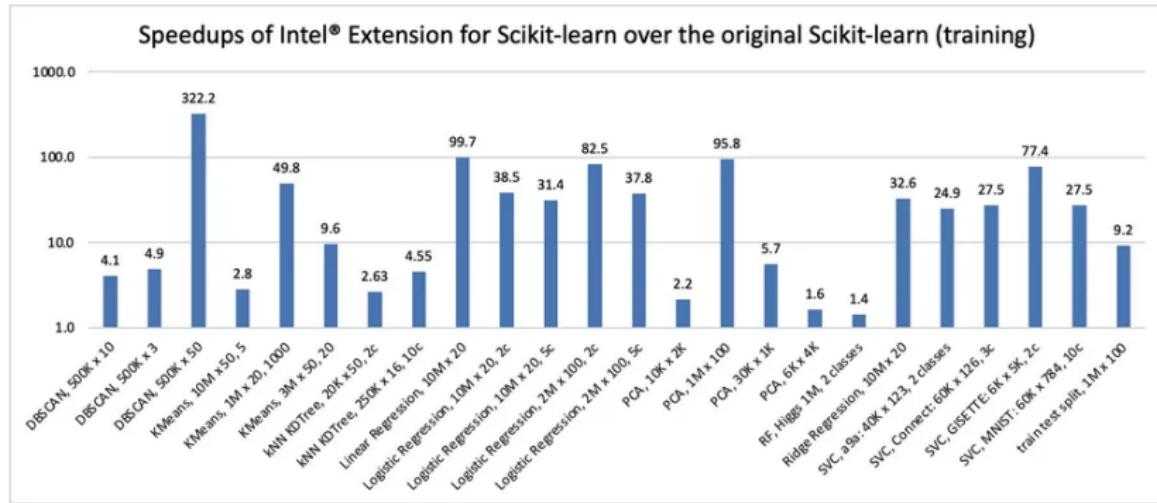
- Implementación de algoritmos de ML: árboles de decisión, SVM, regresión logística, k-means y otros
- Integración con otras bibliotecas como NumPy, Pandas y SciPy
- Herramientas para la extracción de características y reducción de dimensionalidad
- Soporte de aprendizaje supervisado y no supervisado

Tipo	Algoritmos
Supervisado	Regresión lineal, Regresión logística, Árboles de decisión, Bosques aleatorios, SVM, KNN, Gradient Boosting, Redes neuronales, Naive Bayes
No Supervisado	Clustering (K-means, DBSCAN), PCA, ICA, LDA

- Otros:
 - Basados en árboles de decisión: Árboles de decisión, Gradient Boosting...
 - Basados en redes neuronales: Redes neuronales convolucionales (CNN), Redes neuronales recurrentes (RNN), Long short-term memory (LSTM)

Extensión de Intel para Scikit-learn

- Intel ha creado una librería equivalente a Scikit-learn con mejor rendimiento que contiene versiones parcheadas de [32 algoritmos scikit-learn](#)



Introducción

- La [Intel Extension para Scikit-learn](#) proporciona una forma de acelerar el código scikit-learn
 - En un código existente, importaremos **sklearnex**: nombre de la biblioteca de python para el Extensión Intel
 - Usando “**patching**”: los algoritmos de scikit-learn son intercambiadas por la versión optimizada de Intel
 - Varias maneras:
 - Sin editar el código: invocando *python* con un flag adicional
 - Modificando el código: importando e invocando a las funciones de `sklearnex`
 - Añadiendo parches: sin modificar el resto del código

Alternativas

- Línea de comando

```
python -m sklearnex my_application.py
```

- Dentro del script o el cuaderno de Jupyter, **parcheando** el código

```
from sklearnex import patch_sklearn  
patch_sklearn()
```

- Inhabilitando el parche

```
from sklearnex import unpatch_sklearn  
unpatch_sklearn()
```

Hands-on

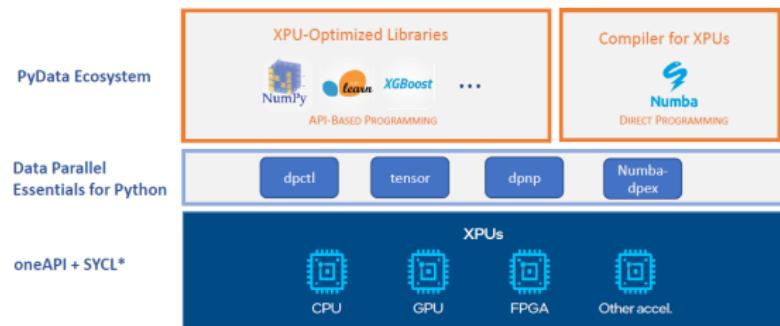
- ① Conéctate al Intel DevCloud
- ② Vamos a abrir el cuarto cuaderno de Jupyter relacionado con el ML con la extensión Intel(R) Extension for Scikit-learn
- ③ Te animamos a que pongas en práctica el contenido con los [ejercicios](#) y puedes consultar también las [soluciones](#)



Data Parallel

Data Parallel y Python

- Desarrollo de aplicaciones para alto rendimiento codificando en python y ejecutándose en los Intel® XPU
- Numba traduce funciones Python usando LLVM



Data Parallel para Python

- Las extensiones de Data Parallel para Python amplían las capacidades numéricas de Python
 - Más allá uso CPU: dispositivos de alto rendimiento como GPU
- Tres paquetes básicos:
 - **dpctl**: librería de control para seleccionar de dispositivos, la asignación de datos, datos tensoriales
 - **dppn**: extensiones paralelas para Numpy
 - Implementa un subconjunto de Numpy
 - **numba_dpex**: extensiones del compilador Data Parallel para Numba
 - Extensión para el compilador Numba para programar dispositivos paralelos
- La implementación se basa en el estándar de **SYCL**

- dpctl: contenedor ligero de Python sobre un subconjunto de la API de DPC++/SYCL
- Ej: selección de dispositivos

devices.py

```
import dpctl

d = dpctl.select_gpu_device()
d.print_device_info()
```

Terminal #1

```
Name           Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz
Driver version 2022.14.7.0.30_160000
Vendor         Intel(R) Corporation
Profile        FULL_PROFILE
Filter string  opencl:cpu:0

Name           Intel(R) UHD Graphics 620 [0x5917]
Driver version 21.38.21026
Vendor         Intel(R) Corporation
Profile        FULL_PROFILE
Filter string  opencl:gpu:0
```

NumPy-dpnp

- La librería dpnp (NumPy Drop-In Replacement for Intel(R) XPU) permite desarrollar aplicaciones en entorno NumPy en dispositivos paralelos
 - Partiendo de un script en [NumPy](#)

```
# Original numpy code
import numpy

X = np.array([1,2,3])
Y = X * 4
```

- Reemplazar las instancias a **numpy** ejecutandolo el código en GPU

```
# numpy as dpnp
import dpnp as numpy

X = np.array([1,2,3])
Y = X * 4
print("Array X allocated on the device:", X.device)
```

Numba

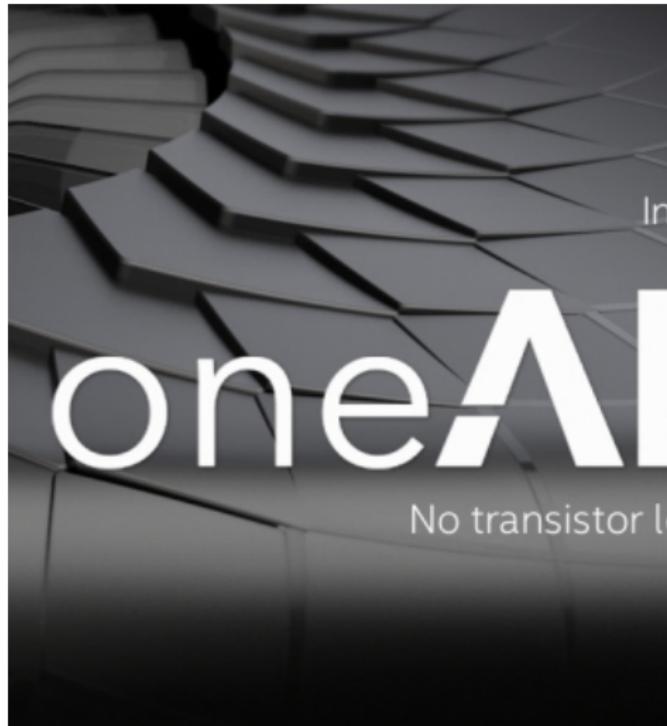
- Operaciones que paralleliza Numba de forma automática
 - Expresiones for-loop

```
from numba import njit, prange

@njit(parallel=True)
def prange_test(A):
    s = 0
    # Without "parallel=True" in the jit-decorator
    # the prange statement is equivalent to range
    for i in prange(A.shape[0]):
        s += A[i]
    return s
```

Hands-on

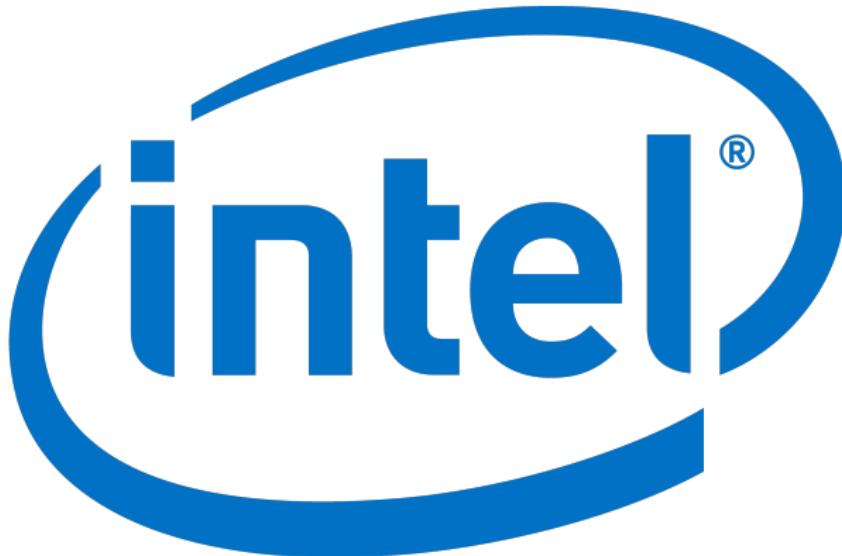
- ① Conéctate al Intel DevCloud
- ② Vamos a abrir el último cuaderno de Jupyter relacionado con el ML en XPU



Otros

Recursos disponibles

- Curso ML con oneAPI
- Iniciativa oneAPI
- Intel oneAPI Base & HPC Toolkit



Software

¡¡¡Gracias!!!



Dirección

Avda. de la industria 4, edif. 1
28108 Alcobendas | Madrid | España

Correo

info@danysoft.com

Teléfono

[+34] 91 663 8683

Sitio Web

www.danysoft.com/intel

