

# **ANEXOS: FUNDAMENTOS DE MANIPULACIÓN DE DATOS CON PANDAS Y NUMPY**

## **SESION 1 / SEMANA 1**

## Anexo 01: Tarea Aprendizaje 01

Alumno: [Juan Piero Vincha Loza]

### 1.- Capturar pantalla de la version python (3 ptos)

```

A juanito ~/senati/machine-learning-main >> ls
A01 README
A juanito ~/senati/machine-learning-main >> cd A01
A juanito ~/senati/machine-learning-main/A01 >> python --version
Python 3.12.5
A juanito ~/senati/machine-learning-main/A01 >> python -m venv venv
A juanito ~/senati/machine-learning-main/A01 >> source venv/bin/activate

(venv) A juanito ~/senati/machine-learning-main/A01 >> pip install numpy
Collecting numpy
  Downloading numpy-2.1.1-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (60 kB)
  Downloading numpy-2.1.1-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (16.0 MB)
    16.0/16.0 MB 9.1 MB/s eta 0:00:00
Installing collected packages: numpy
Successfully installed numpy-2.1.1
(venv) A juanito ~/senati/machine-learning-main/A01 >> pip freeze
numpy==2.1.1
(venv) A juanito ~/senati/machine-learning-main/A01 >> pip freeze > requirements.txt
(venv) A juanito ~/senati/machine-learning-main/A01 >> cat requirements.txt
File: requirements.txt
1 numpy==2.1.1
(venv) A juanito ~/senati/machine-learning-main/A01 >>

```

### 2.- Instalar el paquete numpy con pip install numpy y ejecutar en el terminal pip freeze para ver que version de numpy tiene (4 ptos)

```

A juanito ~ >> su
Contraseña:
[root@pcdejuanito juanito]# pacman -S python-numpy
resolviendo dependencias...
buscando conflictos entre paquetes...

Paquetes (1) python-numpy-2.0.1-1

Tamaño total de la instalación: 47,29 MiB

:: ¿Continuar con la instalación? [S/n] s
(1/1) comprobando las claves del depósito
(1/1) verificando la integridad de los paquetes
(1/1) cargando los archivos de los paquetes
(1/1) comprobando conflictos entre archivos
(1/1) comprobando el espacio disponible en el disco
:: Procesando los cambios de los paquetes...
(1/1) instalando python-numpy
Dependencias opcionales para python-numpy
blas-openblas: faster linear algebra
:: Ejecutando los «hooks» de posinstalación...
(1/1) Arming ConditionNeedsUpdate...
[root@pcdejuanito juanito]# pip show numpy
Name: numpy
Version: 2.0.1
Summary: Fundamental package for array computing in Python
Home-page: https://numpy.org
Author: Travis E. Oliphant et al.
Author-email:
License: Copyright (c) 2005-2024, NumPy Developers.

```

### 3.-Ejecutar el código y mostrar el resultado (7 pts)

```

A juanito ~ >> sudo pacman -Syu
:: Sincronizando las bases de datos de los paquetes...
core está actualizado
extra está actualizado
multilib                                     135,8 KiB  90,7 KiB/s 00:01 [#####] 100%
:: Iniciando actualización completa del sistema...
...el sistema ya está actualizado.
A juanito ~ >> sudo pacman -S python-virtualenv

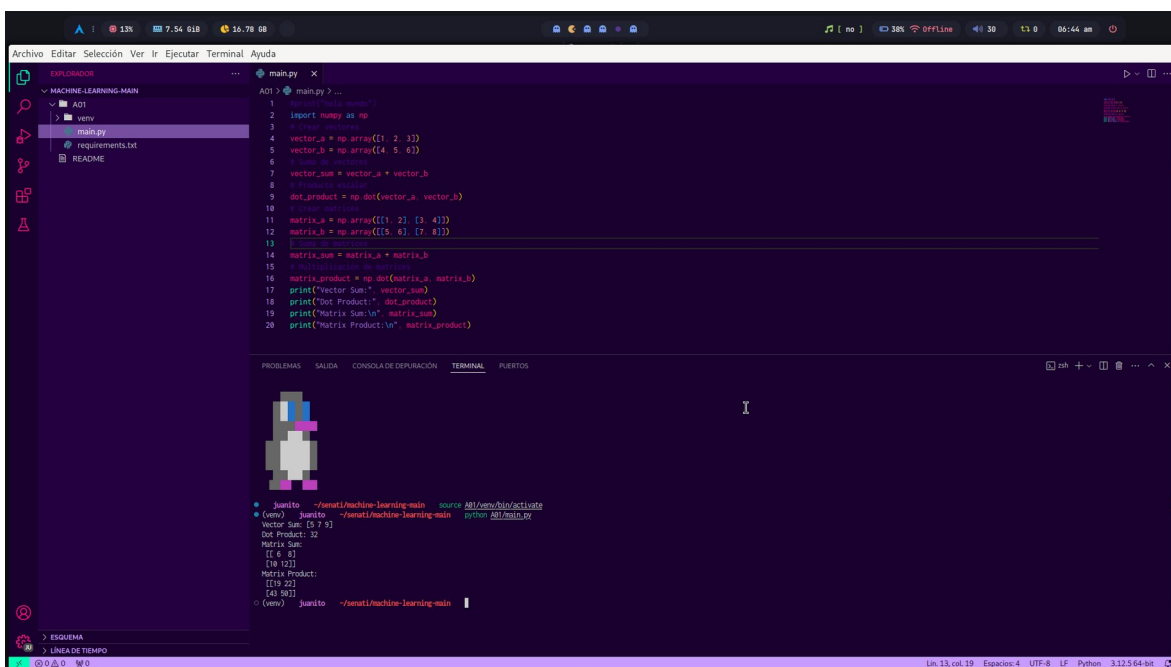
resolviendo dependencias...
buscando conflictos entre paquetes...

Paquetes (3) python-distlib-0.3.8-2 python-platformdirs-4.2.2-1 python-virtualenv-20.26.2-1

Tamaño total de la descarga:    3,69 MiB
Tamaño total de la instalación: 6,28 MiB

:: ¿Continuar con la instalación? [S/n] s
:: Obteniendo los paquetes...
python-virtualenv-20.26.2-1-any             3,5 MiB   399 KiB/s 00:09 [#####] 100%
python-distlib-0.3.8-2-any                 245,1 KiB 583 KiB/s 00:00 [#####] 100%
Total (2/2)                               3,7 MiB   395 KiB/s 00:10 [#####] 100%
(3/3) comprobando las claves del depósito
(3/3) verificando la integridad de los paquetes
(3/3) cargando los archivos de los paquetes
(3/3) comprobando conflictos entre archivos
(3/3) comprobando el espacio disponible en el disco
:: Procesando los cambios de los paquetes...
(1/3) instalando python-distlib
(2/3) instalando python-platformdirs
(3/3) instalando python-virtualenv
:: Ejecutando los «hooks» de posinstalación...
(1/1) Arming ConditionNeedsUpdate...
A juanito ~ >>

```



```

main.py
1 import sys
2 import numpy as np
3
4 vector_a = np.array([1, 2, 3])
5 vector_b = np.array([4, 5, 6])
6
7 # Suma de vectores
8 vector_sum = vector_a + vector_b
9
10 dot_product = np.dot(vector_a, vector_b)
11
12 # Crear matrices
13 matrix_a = np.array([[1, 2], [3, 4]])
14 matrix_b = np.array([[5, 6], [7, 8]])
15
16 # Suma de matrices
17 matrix_sum = matrix_a + matrix_b
18
19 # Producto de matrices
20 matrix_product = np.dot(matrix_a, matrix_b)
21
22 print("Vector Sum:", vector_sum)
23 print("Dot Product:", dot_product)
24 print("Matrix Sum:\n", matrix_sum)
25 print("Matrix Product:\n", matrix_product)

```

```

A juanito ~/senati/machine-learning-main source ABI/virtualenv/activate
(venv) juanito ~/senati/machine-learning-main python ABI/main.py
Vector Sum: [5 7 9]
Dot Product: 32
Matrix Sum:
[[6 8]
 [8 12]]
Matrix Product:
[[19 22]
 [43 58]]

```

Este código ejecuta operaciones básicas con vectores y matrices utilizando la biblioteca NumPy en Python.

#### 4.- Cual es la diferencia entre vector y un matriz ? (2 PTOS)

La principal diferencia entre un vector y una matriz es su dimensionalidad. Un vector es una estructura unidimensional, como una lista de números en una sola fila o columna. Por ejemplo, [1, 2, 3, 4] es un vector. Una matriz, en cambio, es bidimensional, como una tabla con filas y columnas. Por ejemplo: [[1, 2, 3], [4, 5, 6], [7, 8, 9]] es una matriz de 3x3. Los vectores son útiles para representar listas simples de datos, mientras que las matrices se usan para datos más complejos o relaciones entre variables.

#### 5.- Cual menciona 5 metodos que tenga pandas y 5 metodos que tenga numpy que hacen cada uno(4 PTOS)

Pandas:

1. `read_csv()`: Lee un archivo CSV y lo convierte en un DataFrame. Súper útil para cargar datos.
2. `groupby()`: Agrupa el DataFrame por una o más columnas. Lo uso mucho para análisis de datos.
3. `merge()`: Combina dos DataFrames basándose en una columna común. Genial para unir diferentes conjuntos de datos.
4. `dropna()`: Elimina las filas o columnas que contienen valores nulos. Ayuda a limpiar los datos.
5. `describe()`: Genera estadísticas descriptivas del DataFrame. Da una visión rápida de los datos.

Numpy:

1. `array()`: Crea un array de numpy. Básico pero esencial para empezar a trabajar con numpy.
2. `reshape()`: Cambia la forma de un array sin modificar sus datos. Útil para reorganizar datos.
3. `mean()`: Calcula el promedio de los elementos en un array. Lo uso mucho en estadísticas.
4. `random.rand()`: Genera números aleatorios en un array. Genial para simulaciones.
5. `dot()`: Realiza el producto punto entre arrays. Muy usado en álgebra lineal y cálculos matriciales.

Estos métodos se usan bastante en proyectos de análisis de datos y machine learning. Pandas es genial para manejar datos estructurados, mientras que numpy es súper eficiente para operaciones numéricas.