**Data-frame creation using NumPy and cuPY**

**# TensorFlow and tf.keras**

* import tensorflow as tf
* import cupy as cp
* import cudf

**# Helper libraries**

* import numpy as np
* import matplotlib.pyplot as plt
* import pandas as pd
* print(tf.\_\_version\_\_)

**# You may need to run following pip if you are using Kaggle**

* pip install cupy-cuda(version)
* pip install cudf

**# Test 1**

* cudf.DataFrame({
* 'foo': [1,2,3,4]
* , 'bar': ['a','b','c',None]
* })

**# Test 2**

* df\_sample = cudf.DataFrame()
* df\_sample['foo'] = [1,2,3,4]
* df\_sample['bar'] = ['a','b','c',None]

**# Test 3**

* cudf.DataFrame([
* (1, 'a')
* , (2, 'b')
* , (3, 'c')
* , (4, None)
* ], columns=['ints', 'strings'])

**# Test 4**

Write the code to read csv file or load data from GitHub for both cuPY and NumPY

**CPU vs GPU (Performance/Time**)

**#Numpy vs CuPY**

* import cupy as cp
* import numpy as np
* import time

**# Create array**

**# Run 1**

* x\_cpu = np.array([1, 2, 3])

**# Run 2**

* x\_gpu = np.array([1, 2, 3])

**# Run 3**

* l2\_cpu = np.linalg.norm(x\_cpu)
* l2\_gpu = cp.linalg.norm(x\_gpu)

**# Run 4**

* print("Using Numpy: ", l2\_cpu)
* print("\nUsing Cupy: ", l2\_gpu)

**# Run 5**

**# NumPy and CPU Runtime**

* s = time.time()
* x\_cpu = np.ones((1000, 1000, 100))
* e = time.time()
* print("Time consumed by numpy: ", e - s)

**# Run 6**

**# CuPy and GPU Runtime**

* s = time.time()
* x\_gpu = cp.ones((1000, 1000, 100))
* e = time.time()
* print("\nTime consumed by cupy: ",e - s)

**# Run 6**

**# Numpy and CPU**

* s = time.time()
* x\_cpu \*= 5
* e = time.time()
* print(e - s)

**# Run 6**

**# CuPy and GPU**

* s = time.time()
* x\_gpu \*= 5
* cp.cuda.Stream.null.synchronize()
* e = time.time()
* print(e - s)

**# Run 7**

**# Numpy and CPU**

* s = time.time()
* x\_cpu \*= 5
* x\_cpu \*= x\_cpu
* x\_cpu += x\_cpu
* e = time.time()
* print(e - s)

**# Run 8**

**# CuPy and GPU**

* s = time.time()
* x\_gpu \*= 5
* x\_gpu \*= x\_gpu
* x\_gpu += x\_gpu
* cp.cuda.Stream.null.synchronize()
* e = time.time()
* print(e - s)

**# Run 8**

**# Move array from a device to the host**

* x\_gpu = cp.array([1, 2, 3]) # create an array in the current device
* x\_cpu = cp.asnumpy(x\_gpu) # move the array to the host.
* x\_cpu