**UNIVERSIDAD TECNOLÓGICA DE PANAMÁ**

FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

**Parcial II – Parte Práctica (40%)**



**Materia**: Inteligencia Artificial aplicada a la Ciberseguridad **Profesor**: Dr. Carlos A. Rovetto

**Fecha**: miércoles, 11 de junio de 2025 Grupo: \_1S3134\_\_

Estudiante: Anthony Marbal Cédula: 20-70-7744 Fila: \_\_\_

Estudiante: Samuel Mejia Cédula: 8-1012-1580 Fila: \_\_\_

**Objetivo**

Aplicar los conocimientos recibidos sobre la creación de datasets utilizando Python y el análisis de estos.

**Descripción**: Crear un dataset desde procesos de su computadora y subirlo a un sitio público como los siguientes (GitHub, Google Drive, Kaggle, AWS S3, Dropbox, etc). Posteriormente, elabore el código Python para acceder a él a través de python y genere la siguiente información para el dataset generado.

* Estadísticas básicas: Media, mediana, moda y desviación estándar.
* Correlación entre variables: Matriz de correlación para ver las relaciones entre variables numéricas.

1. Eventos del Sistema (Logs): Identificar patrones de eventos que puedan señalar intentos de intrusión o fallos de seguridad.

**Observaciones**: Se le dará un enlace de un formulario para que suba el código Python y el enlace del dataset público. Debe asegurarse que a través de la ejecución del código se puede tener acceso al dataset.

# CODIGO

import os

import pandas as pd

import numpy as np

from scipy import stats

import requests

from openpyxl import Workbook

import base64

# === PASO 1: GENERAR DATOS SIMULADOS DE LOGS DEL SISTEMA ===

np.random.seed(42)  # Para reproducibilidad

num\_registros = 3000

data = {

    "timestamp": pd.date\_range(start="2025-01-01", periods=num\_registros, freq="h"),

    "event\_type": np.random.choice(["login", "logout", "error", "warning"], size=num\_registros),

    "user\_id": np.random.randint(1, 10, size=num\_registros),

    "response\_time\_ms": np.random.normal(loc=500, scale=100, size=num\_registros).astype(int),

    "cpu\_usage\_percent": np.random.uniform(10, 90, size=num\_registros).round(2),

    "memory\_usage\_percent": np.random.uniform(20, 80, size=num\_registros).round(2),

}

df = pd.DataFrame(data)

# === PASO 2: GUARDAR EL ARCHIVO EN EL ESCRITORIO ===

desktop\_path = os.path.join(os.path.expanduser("~"), "Desktop")

file\_path = os.path.join(desktop\_path, "system\_logs.xlsx")

df.to\_excel(file\_path, index=False)

print(f"Archivo guardado en: {file\_path}")

# === PASO 3: SUBIR EL ARCHIVO A GITHUB ===

github\_token = "ghp\_FNOrQnPQvVKFfv77lKXNHtR2bqFKkO17PwuA"

github\_username = "AnthonMttv"

repository\_name = "ParcialdeAI"

file\_name\_in\_repo = "data/system\_logs.xlsx"

commit\_message = "Subir archivo de logs"

# Leer el contenido del archivo local

with open(file\_path, "rb") as file:

    content = file.read()

# Codificar el contenido en Base64

encoded\_content = base64.b64encode(content).decode("utf-8")

# URL de la API de GitHub

url = f"https://api.github.com/repos/{github\_username}/{repository\_name}/contents/{file\_name\_in\_repo}"

headers = {

    "Authorization": f"token {github\_token}",

    "Accept": "application/vnd.github.v3+json",

}

payload = {

    "message": commit\_message,

    "content": encoded\_content,

}

# Subir el archivo por primera vez (no existe aún)

response = requests.put(url, headers=headers, json=payload)

if response.status\_code in [200, 201]:

    print("Archivo subido exitosamente a GitHub.")

else:

    print(f"Error al subir el archivo a GitHub: {response.status\_code}, {response.text}")

    exit()

# === PASO 4: DESCARGAR Y LEER EL ARCHIVO DESDE GITHUB ===

github\_url = "https://raw.githubusercontent.com/AnthonMttv/ParcialdeAI/main/data/system\_logs.xlsx"

response = requests.get(github\_url)

if response.status\_code == 200:

    print("Archivo descargado exitosamente desde GitHub.")

else:

    print("Error al descargar el archivo desde GitHub.")

    exit()

df\_github = pd.read\_excel(response.content)

# === PASO 5: CALCULAR ESTADÍSTICAS BÁSICAS ===

numeric\_columns = df\_github.select\_dtypes(include=[np.number])

media = numeric\_columns.mean()

mediana = numeric\_columns.median()

moda = numeric\_columns.mode().iloc[0]

desviacion\_estandar = numeric\_columns.std()

print("\nEstadísticas básicas:")

print("Media:\n", media)

print("\nMediana:\n", mediana)

print("\nModa:\n", moda)

print("\nDesviación Estándar:\n", desviacion\_estandar)

# === PASO 6: MATRIZ DE CORRELACIÓN ===

correlation\_matrix = numeric\_columns.corr()

print("\nMatriz de correlación:\n", correlation\_matrix)

# === PASO 7: GUARDAR LAS ESTADÍSTICAS EN EL MISMO ARCHIVO EXCEL ===

output\_stats = pd.DataFrame({

    "Media": media,

    "Mediana": mediana,

    "Moda": moda,

    "Desviacion\_Estandar": desviacion\_estandar

}).reset\_index()

with pd.ExcelWriter(file\_path, engine="openpyxl", mode="a", if\_sheet\_exists="replace") as writer:

    output\_stats.to\_excel(writer, sheet\_name="Estadisticas", index=False)

print("\nEstadísticas guardadas en la hoja 'Estadisticas' del archivo Excel.")

# CAPTURAS DE SALIDA





