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: viernes, 13 de junio de 2025 Grupo: _1S3132__

Estudiante: Juan Ignacio Zetina Cédula: 293641331 Fila: 2

Estudiante: Jefry Rangel Cédula: 8-1010-1690 Fila: 2

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.

Ejemplos de tipos de datasets que puede crear

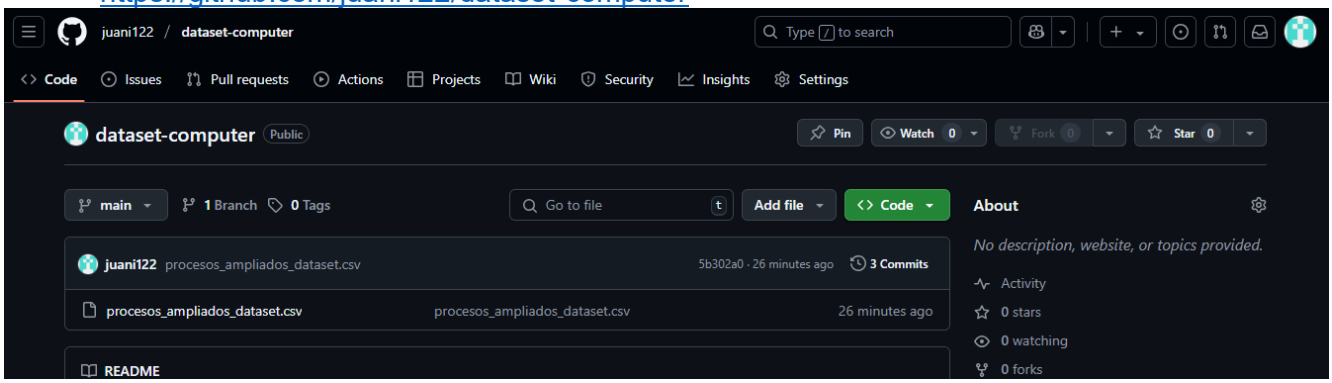
1. Uso de la CPU: Detectar anomalías en el comportamiento de la CPU para identificar posibles ataques de denegación de servicio (DDoS).
2. Uso de la Memoria RAM: Monitorear el uso excesivo de memoria para detectar malware o procesos maliciosos.
3. Uso del Disco Duro: Identificar cambios sospechosos en el uso del disco que podrían indicar filtración de datos o actividad maliciosa.
4. Uso de la Red: Detectar patrones de tráfico anómalos para identificar intrusiones o ataques de red.
5. Lista de Procesos Activos: Clasificar procesos maliciosos y legítimos para prevenir accesos no autorizados o malware.
6. Temperatura de Componentes del Sistema: Detectar sobrecalentamientos anómalos que puedan indicar un ataque físico o manipulación del hardware.
7. Tiempo de Actividad del Sistema: Analizar tiempos de actividad inusuales para detectar sistemas comprometidos o accesos no autorizados.
8. Eventos del Sistema (Logs): Identificar patrones de eventos que puedan señalar intentos de intrusión o fallos de seguridad.
9. Consumo de Energía: Detectar patrones inusuales de consumo energético relacionados con ataques físicos o manipulaciones de hardware.
10. Actividades de Entrada del Usuario (Teclado): Analizar patrones de comportamiento del usuario para detectar accesos no autorizados o actividades maliciosas.

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.

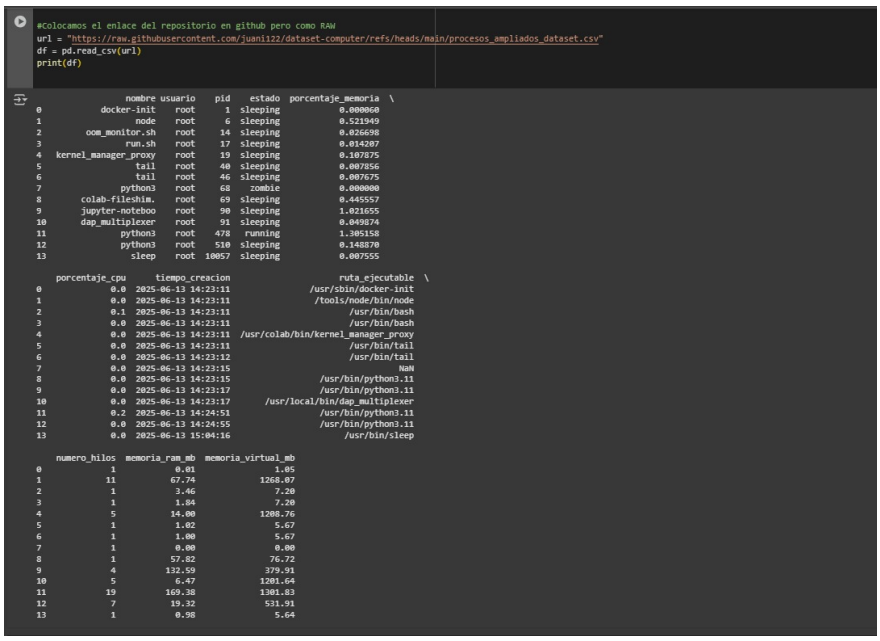
Desarrollo

Paso 1:
Primero que todo creamos el dataset y lo subimos a GITHUB:

Al dataset se puede acceder por medio del enlace a github:
https://github.com/juani122/dataset-computer



Paso 2:
Luego creamos el código en Python para poder descargar el código desde GitHub y que aparezca en Python, mostrando así el dataset.



```
#Colocamos el enlace del repositorio en github pero como RAW
url = "https://raw.githubusercontent.com/juani122/dataset-computer/refs/heads/main/procesos_ampliados_dataset.csv"
df = pd.read_csv(url)
print(df)
```

```
                nombre usuario    pid    estado  porcentaje_memoria  \
0          docker-init    root      1  sleeping            0.000060
1                 node    root      6  sleeping            0.521949
2         oom_monitor.sh    root     14  sleeping            0.026698
3                 run.sh    root     17  sleeping            0.014207
4   kernel_manager_proxy    root     19  sleeping            0.107875
5                   tail    root     40  sleeping            0.007856
6                   tail    root     46  sleeping            0.007675
7                python3    root     68    zombie            0.000000
8       colab-fileshim.    root     69  sleeping            0.445557
9       jupyter-noteboo    root     90  sleeping            1.021655
10      dap_multiplexer    root     91  sleeping            0.049874
11               python3    root    478   running            1.305158
12               python3    root    510  sleeping            0.148870
13                 sleep    root  10057  sleeping            0.007555

    porcentaje_cpu     tiempo_creacion                    ruta_ejecutable  \
0              0.0  2025-06-13 14:23:11               /usr/sbin/docker-init
1              0.0  2025-06-13 14:23:11                /tools/node/bin/node
2              0.1  2025-06-13 14:23:11                     /usr/bin/bash
3              0.0  2025-06-13 14:23:11                     /usr/bin/bash
4              0.0  2025-06-13 14:23:11  /usr/colab/bin/kernel_manager_proxy
5              0.0  2025-06-13 14:23:11                     /usr/bin/tail
6              0.0  2025-06-13 14:23:12                     /usr/bin/tail
7              0.0  2025-06-13 14:23:15                               NaN
8              0.0  2025-06-13 14:23:15                /usr/bin/python3.11
9              0.0  2025-06-13 14:23:17                /usr/bin/python3.11
10             0.0  2025-06-13 14:23:17       /usr/local/bin/dap_multiplexer
11             0.2  2025-06-13 14:24:51                /usr/bin/python3.11
12             0.0  2025-06-13 14:24:55                /usr/bin/python3.11
13             0.0  2025-06-13 15:04:16                     /usr/bin/sleep

    numero_hilos  memoria_ram_mb  memoria_virtual_mb
0              1            0.01                1.05
1             11           67.74             1268.07
2              1            3.46                7.20
3              1            1.84                7.20
4              5           14.00             1288.76
5              1            1.02                5.67
6              1            1.00                5.67
7              1            0.00                0.00
8              1           57.82               76.72
9              4          132.59              379.91
10             5            6.47             1201.64
11            19          169.38             1301.83
12             7           19.32              531.91
13             1            0.98                5.64
```

Paso 3:
Sacamos algunos datos adicionales como la moda y estadísticas básicas sobre las memorias RAM, los procesos del CPU etc.

```
print(" Estadísticas básicas:")
print(df[['porcentaje_cpu', 'porcentaje_memoria', 'memoria_ram_mb', 'memoria_virtual_mb']].describe())

print("\n Moda:")
print(df[['porcentaje_cpu', 'porcentaje_memoria', 'memoria_ram_mb', 'memoria_virtual_mb']].mode())
```

```
 Estadísticas básicas:
       porcentaje_cpu  porcentaje_memoria  memoria_ram_mb  memoria_virtual_mb
count       14.000000           14.000000       14.000000           14.000000
mean         0.021429            0.261785       33.973571          428.662143
std          0.057893            0.420266       54.541575          559.438712
min          0.000000            0.000000        0.000000            0.000000
25%          0.000000            0.007720        1.005000            5.670000
50%          0.000000            0.038286        4.965000           41.960000
75%          0.000000            0.371386       48.195000         1034.207500
max          0.200000            1.305158      169.380000         1301.830000

 Moda:
    porcentaje_cpu  porcentaje_memoria  memoria_ram_mb  memoria_virtual_mb
0              0.0            0.000000            0.00                5.67
1              NaN            0.000060            0.01                7.20
2              NaN            0.007555            0.98                 NaN
3              NaN            0.007675            1.00                 NaN
4              NaN            0.007856            1.02                 NaN
5              NaN            0.014207            1.84                 NaN
6              NaN            0.026698            3.46                 NaN
7              NaN            0.049874            6.47                 NaN
8              NaN            0.107875           14.00                 NaN
9              NaN            0.148870           19.32                 NaN
10             NaN            0.445557           57.82                 NaN
11             NaN            0.521949           67.74                 NaN
12             NaN            1.021655          132.59                 NaN
13             NaN            1.305158          169.38                 NaN
```

Paso 4:

Sacamos la matriz de correlación, pera ver la relación entre las variables numéricas.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Solo seleccionamos columnas numéricas relevantes
correlation_matrix = df[['porcentaje_cpu', 'porcentaje_memoria', 'memoria_ram_mb', 'memoria_virtual_mb']].corr()

# Graficamos el heatmap
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title(" Matriz de Correlación entre Variables")
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 128279 (\N{LINK SYMBOL}) missing from font(s) DejaVu Sans.
  fig.canvas.print_figure(bytes_io, **kw)
```