UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

INTELIGENCIA ARTIFICIALY REDES NEURONALES

Actividad 5: Procesamiento de conjuntos de datos en Python

Catedrático: Daniel Isaías López Páez

Semestre: agosto-diciembre 2025

Grupo: 008

Hora: N4 LMV

Matrícula	Nombre completo	Carrera
2173882	Diana Sánchez Arévalo	IMTC
2057039	Elizabeth Spinoso Castillo	IMTC
2041473	Luis Daniel Aguilar Miranda	IMTC
2053928	Melenie azeneth Domínguez González	IMTC
2052196	Luis Antonio Pérez Espinoza	IMTC

Objetivo. -

- Familiarizarse con el lenguaje de programación Python y recordar conceptos de lógica de programación en general.
- Conocer la interfaz y funcionamiento de Google Colab.
- Usar librerías como Matplotlib, Pandas y Numpy.

Ejercicio 1.- Análisis y normalización de datos de sensores en un robot móvil

1. Carga del Dataset

Se utilizó la biblioteca pandas para cargar el archivo CSV con los datos de los sensores del robot móvil.
Este dataset contiene columnas como Tiempo (s), Distancia (cm), Velocidad (cm/s) y Temperatura (°C).
Se mostraron las primeras filas para verificar la estructura y el contenido del dataset.

In [9]:
```
url = "https://raw.githubusercontent.com/dilp90/InteligenciaArtificial_y_RedesNeuronales_UANL_FIME/main/MachineLearning/D
df = pd.read_csv(url)

# Mostrar primeras filas
df.head()
```

Out[9]:

	Tiempo (s)	Distancia (cm)	Velocidad (cm/s)	Temperatura (°C)
0	0	43.708611	9.256646	23.925585
1	1	95.564288	27.095047	23.704682
2	2	75.879455	43.647292	33.593819
3	3	63.879264	36.611244	23.743193
4	4	24.041678	40.328057	24.079246

In [10]:
```
# Información general
df.info()

# Valores estadísticos
df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 4 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Tiempo (s)        1000 non-null   int64  
 1   Distancia (cm)    970 non-null    float64
 2   Velocidad (cm/s)  970 non-null    float64
 3   Temperatura (°C)  970 non-null    float64
dtypes: float64(3), int64(1)
memory usage: 31.4 KB
```

Out[10]:

	Tiempo (s)	Distancia (cm)	Velocidad (cm/s)	Temperatura (°C)
count	1000.000000	970.000000	970.000000	970.000000
mean	499.500000	54.011809	25.203708	27.518110
std	288.819436	26.408040	14.632979	4.386129
min	0.000000	10.416882	0.160913	20.000175
25%	249.750000	30.581348	11.877176	23.890799
50%	499.500000	54.633590	25.874517	27.507535
75%	749.250000	76.891589	37.989158	31.409083
max	999.000000	99.974591	49.970686	34.967313

2. Limpieza de Datos

Se identificaron valores nulos y se eliminaron las filas que contenían datos faltantes.

Además, se filtraron los valores atípicos, como velocidades negativas, ya que no son físicamente posibles.

Este proceso es fundamental para evitar sesgos en el análisis posterior.

In [11]:
```
# Eliminar filas con valores nulos
df_clean = df.dropna()

# Eliminar valores atípicos (ejemplo: velocidad negativa)
df_clean = df_clean[df_clean["Velocidad (cm/s)"] >= 0]

df_clean.head()
```

Out[11]:

	Tiempo (s)	Distancia (cm)	Velocidad (cm/s)	Temperatura (°C)
0	0	43.708611	9.256646	23.925585
1	1	95.564288	27.095047	23.704682
2	2	75.879455	43.647292	33.593819
3	3	63.879264	36.611244	23.743193
4	4	24.041678	40.328057	24.079246

In [12]:
```
df_clean.isnull().sum()
```

Out[12]:

	0
Tiempo (s)	0
Distancia (cm)	0
Velocidad (cm/s)	0
Temperatura (°C)	0

3. Normalización de Datos

Para normalizar los datos numéricos se aplicó la técnica Min-Max Scaling, que ajusta los valores a un rango de 0 a 1.

In [13]:

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

# Normalizamos solo las columnas numéricas
cols = ["Tiempo (s)", "Distancia (cm)", "Velocidad (cm/s)", "Temperatura (°C)"]
df_norm = pd.DataFrame(scaler.fit_transform(df_clean[cols]), columns=cols)

df_norm.head()
```

Out[13]:

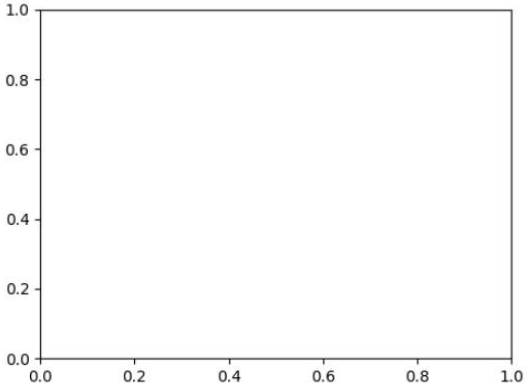
	Tiempo (s)	Distancia (cm)	Velocidad (cm/s)	Temperatura (°C)
0	0.000000	0.371735	0.182609	0.262269
1	0.001001	0.950755	0.540740	0.247509
2	0.002002	0.730954	0.873049	0.908233
3	0.003003	0.596960	0.731791	0.250082
4	0.004004	0.152134	0.806411	0.272535

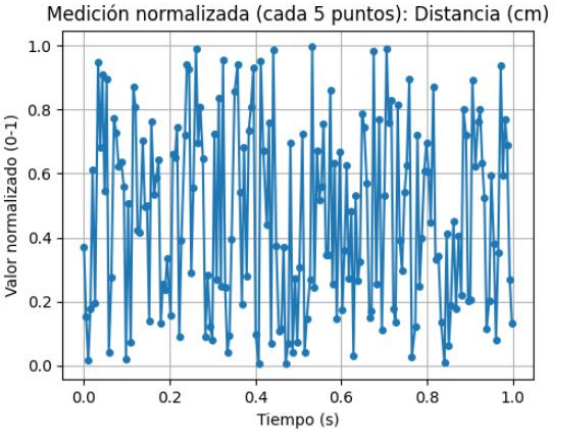
4. Visualización de los Datos

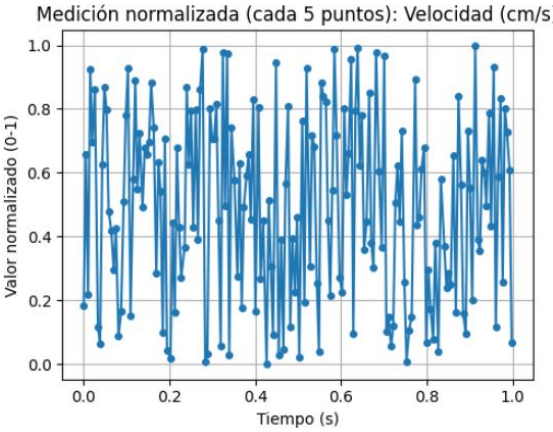
Finalmente, se utilizaron gráficos con Matplotlib para visualizar los datos normalizados y analizar las relaciones entre las variables. Se generaron gráficos de líneas y dispersión que muestran las tendencias y comparaciones entre las características del datasheet.

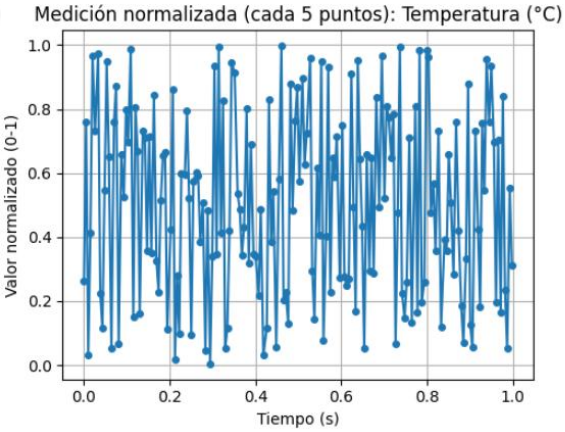
In [20]:

```
import matplotlib.pyplot as plt

# Crear subplots en cuadrícula 2x2
fig, axs = plt.subplots(2, 2, figsize=(10, 8))
axs = axs.flatten()

# Graficar cada columna contra el tiempo con muestreo
for i, col in enumerate(df_norm.columns):
    if col != "Tiempo (s)":  # Evitar graficar el tiempo contra sí mismo
        axs[i].plot(df_norm["Tiempo (s)"][::5], df_norm[col][::5],
                    marker='o', linestyle='-', markersize=4)
        axs[i].set_title(f"Medición normalizada (cada 5 puntos): {col}")
        axs[i].set_xlabel("Tiempo (s)")
        axs[i].set_ylabel("Valor normalizado (0-1)")
        axs[i].grid(True)

plt.tight_layout()
plt.show()
```




Medición normalizada (cada 5 puntos): Distancia (cm)
Valor normalizado (0-1)
Tiempo (s)


Medición normalizada (cada 5 puntos): Velocidad (cm/s)
Valor normalizado (0-1)
Tiempo (s)


Medición normalizada (cada 5 puntos): Temperatura (°C)
Valor normalizado (0-1)
Tiempo (s)

Ejercicio 2.- Preguntas

Responder las siguientes preguntas:

1. **¿Cómo afecta la limpieza de datos al análisis del dataset?**

   La limpieza de datos también es un componente fundamental de la gestión eficaz de datos, que ayuda a garantizar que los datos sigan siendo precisos, seguros y accesibles en cada etapa de su ciclo de vida.

   Los datos de alta calidad o "limpios" son cruciales para adoptar eficazmente la inteligencia artificial (IA) y las herramientas de automatización.

2. **¿Por qué es importante la normalización en Machine Learning?**

   Ayuda a que los modelos converjan más rápido durante el entrenamiento. Cuando las diferentes funciones tienen rangos diferentes, el descenso del gradiente puede "rebotar" y ralentizar la convergencia.

3. **¿Cómo interpretarías los datos visualizados?**

   - Puedes ver cómo evoluciona la distancia, velocidad y temperatura con respecto al tiempo.
   - La normalización facilita comparar tendencias: por ejemplo, identificar si la velocidad aumenta al mismo tiempo que la distancia o si la temperatura se mantiene estable.
   - El muestreo ayuda a visualizar los datos sin tanto ruido, mostrando patrones generales.

Link del repositorio

https://github.com/2173882Diana/Inteligencia-Artificial-y-Redes-Neuronales/blob/main/Actividades/AF5_Procesamiento_de_datos.ipynb

Conclusión

Con esta actividad pude reforzar lo aprendido en clase, como el uso de las librerías pandas, matplotlib y numpy, que resultan fundamentales para el procesamiento y análisis de datos. Además, ahora conozco una forma más práctica y automática de realizar este tipo de tareas, evitando procesos manuales que consumen más tiempo y son más propensos a errores. Con Python es posible trabajar de manera más eficiente, precisa y ordenada, lo que no solo facilita la interpretación de los resultados, sino que también mejora la presentación de la información mediante visualizaciones claras. Este aprendizaje me permitirá abordar futuros proyectos con mayor seguridad y confianza, aplicando las herramientas adecuadas para obtener conclusiones más rápidas y confiables.

Referencias

Rogers, J., & Jonker, A. (2025, 16 mayo). Limpieza de datos. *IBM*. https://www.ibm.com/mx-es/think/topics/data-cleaning

Datos numéricos: Normalización. (n.d.). Google for Developers. https://developers.google.com/machine-learning/crash-course/numerical-data/normalization?hl=es-419