Clase 6: Visualización de información

FM849 - Programación Científica para Proyectos de Inteligencia Artificial (IA)

7 de enero de 2026

Visualización de información

- La información puede almacenarse en diferentes formatos: **grafos, arboles, tablas, diccionarios, entre otros.**
- En este curso nos enfocaremos en datos tabulares (tablas). Este es el formato mas simple y utilizado.
- En la clase anterior vimos como abrir un datasets de este tipo usando Pandas. Ahora, la idea es visualizar estos datos.

Pero antes de eso, por que nos gustaria visualizar los datos ?

- Como vimos en la clase de estadística, los datasets pueden ser bastante complejos para analizar dato a dato.
- Las tecnicas de visualización nos permiten encontrar patrones o tendencias en los datos.
- Se puede decir que la visualización de datos nos permite contar una historia de estos.
- Esto nos ayuda a tomar decisiones sobre los datos.

Matplotlib

Matplotlib es la librería mas popular para visualizar datos en Python. Esta libreria tiene funciones acopladas al paquete de Pandas.

```
import matplotlib.pyplot as plt
```

Función	Uso	Ejemplo
`plot`	Gráfico de linea	`plt.plot(valores)`
`scatter`	Gráfico de puntos	`plt.scatter(valores)`
`bar`	Gráfico de barras	`plt.bar(etiquetas, conteos)`
`boxplot`	Gráfico de caja	`plt.boxplot(valores)`
`histogram`	Histograma	`plt.hist(valores)`
`imshow`	Histograma	`plt.plot(valores)`

Tabla 1: Funciones en matplotlib.

Analizaremos cada una de estas funciones y veremos como aplicarlas en un dataset.

https://colab.research.google.com/drive/1kKzcxZf8Ova4Rd2Yrb-Zui5163hWYb6G?usp=sharing

Matplotlib: plot

El gráfico de líneas se usa para visualizar la evolución de una variable continua, generalmente respecto al tiempo o al orden de observación.

- Se usan para analizar tendencias temporales y comparación la evoluación de variables.

Los ejes x e y representan los valores que adoptan las variables analizadas.

```
>>> import matplotlib.pyplot as plt

>>> y = [1, 3, 2, 5, 4]
>>> plt.plot(y)
>>> plt.show()
```

Matplotlib: scatter

El gráfico de dispersión muestra la relación entre dos variables numéricas.

- Permite detectar correlaciones, identificar patrones, clusteres y encontrar outliers.

Los ejes x e y representan los valores que adoptan las variables analizadas.

```
x = [1, 2, 3, 4, 5]
y = [2, 1, 3, 5, 4]

# plot de (1,2), (2,1), ... (5,4)
plt.scatter(x, y)
plt.show()
```

Matplotlib: hist

El histograma muestra la distribución de una variable numérica.

- Se utiliza para analizar la forma de la distribución, asimetria y concentración de valores.

En general, el eje x representa la variable analizada y el eje y el numero de veces que ocurre esa variable (frecuencia).

```
import numpy as np
# 1000 datos random
temperatura_santiago = np.random.randn(1000) + 25 #media 25 y desviacion es
plt.hist(temperatura_santiago, bins=20)
plt.xlabel("Temperatura [C]")
plt.ylabel("Frecuencia [-]")
plt.show()
```

Matplotlib: boxplot

El boxplot resume una distribución mediante cuartiles y valores extremos.

- Se usa para comparar las distribuciones de diversas categorías en función de sus cuartiles y outliers.

Requiere un atributo categorico (eje x) y un atributo cuantitativo (eje y).

```
datos = [1, 2, 2, 3, 4, 10]
plt.boxplot(datos)
plt.show()
```

Matplotlib: imshow

`imshow` se utiliza para visualizar datos matriciales como imágenes.

- Imágenes.
- Matrices.
- Mapas.

```
import numpy as np

# matrix de 10 filas x 10 columnas
matriz = np.random.rand(10, 10)
plt.imshow(matriz)
plt.colorbar()
plt.show()
```

Aplicación sobre Iris Dataset

- Dataset con 150 datos tabulares.
- Cada instancia (fila) representa una flor.
- En este dataset tiene 5 columnas: **sepallength, sepalwidth, petallength, petalwidth, class**.
- Las clases (class) son categoricas: **setosa, versicolor, virginica**.

Figura 1: Ejemplo de Series.

Por ahora nos limitaremos a visualizar los datos.

https://colab.research.google.com/drive/1kKzcxZf8Ova4Rd2Yrb-Zui5163hWYb6G?usp=sharing

Seaborn

Seaborn es una librería de visualización que permite generar gráficos de manera simple, rápida y con muchos estilos. Con Matplotlib podemos lograr lo mismo pero requiere mayor trabajo.

```
import seaborn as sns
```

- **lineplot**: Gráfico de líneas con estimación estadística e intervalos de confianza.
  `sns.lineplot(data=df, x="x", y="y")`
- **scatterplot**: Gráfico de dispersión con agrupación por color, tamaño o estilo.
  `sns.scatterplot(data=df, x="x", y="y", hue="clase")`
- **histplot**: Histograma y densidad de una variable.
  `sns.histplot(data=df, x="x", kde=True)`
- **boxplot**: Comparación de distribuciones mediante cajas.
  `sns.boxplot(data=df, x="grupo", y="valor")`
- **pairplot**: Relación entre todas las variables numéricas.
  `sns.pairplot(df, hue="clase")`

Referencias:

- Wes McKinney. (2022). Python for Data Analysis. Third Edition.
- https://seaborn.pydata.org/tutorial/function_overview.html