Estadística Descriptiva — Pima Indians Diabetes

Autor: Aaron Cuevas y Fabiola Ochoa · **Fecha:** 2025-10-30

Variables a analizar: **Pregnancies**, **DiabetesPedigreeFunction** e **Outcome** (incluida como referencia).

1. Lectura de datos con pandas

In [ ]:
```
import pandas as pd, numpy as np
import matplotlib.pyplot as plt
from pathlib import Path

pd.set_option("display.max_columns", None)
pd.set_option("display.width", 120)

# Detecta CSV en ruta típica del repo o en el cwd como respaldo
candidates = [Path("data/diabetes.csv"), Path("diabetes.csv")]
for p in candidates:
    if p.exists():
        CSV_PATH = p
        break
else:
    raise FileNotFoundError("No se encontró 'data/diabetes.csv' ni './diabet

df = pd.read_csv(CSV_PATH)
print("CSV:", CSV_PATH.resolve())
print("Dimensiones (filas, columnas):", df.shape)
df.head()
```

2. Inspección general

In [ ]:
```
print("Columnas:", df.columns.tolist())
print("\nTipos de dato y memoria:")
df.info()
print("\nValores nulos por columna:")
df.isna().sum()
```

Normalización de faltantes conocidos

En este dataset, `0` suele significar faltante en: `Glucose`, `BloodPressure`, `SkinThickness`, `Insulin` y `BMI`. **No** aplicamos esta regla a `Pregnancies` (0 puede ser válido) ni a `DiabetesPedigreeFunction`.

In [ ]:
```
cols_zero_na = ["Glucose","BloodPressure","SkinThickness","Insulin","BMI"]
df[cols_zero_na] = df[cols_zero_na].replace(0, np.nan)
df.isna().sum()
```

3. Estadísticos descriptivos (variables seleccionadas)

Variables seleccionadas:

- **Pregnancies**: cuantitativa discreta (conteo).
- **DiabetesPedigreeFunction**: cuantitativa continua (índice).
- **Outcome**: categórica binaria (0 = no diabetes, 1 = diabetes).

In [ ]:
```
vars_sel = ["Pregnancies","DiabetesPedigreeFunction","Outcome"]

# Estadísticos clave
stats = df.agg({
    "Pregnancies": ["min","max","mean","median","std"],
    "DiabetesPedigreeFunction": ["min","max","mean","median","std"],
    "Outcome": ["min","max","mean"]
})
stats
```

In [ ]:
```
# IQR para dispersión robusta en las continuas/discretas
def iqr(s):
    return s.quantile(0.75) - s.quantile(0.25)

pd.DataFrame({
    "Pregnancies_IQR": [iqr(df["Pregnancies"])],
    "DiabetesPedigreeFunction_IQR": [iqr(df["DiabetesPedigreeFunction"])]
})
```

In [ ]:
```
# Correlación simple (Outcome como numérico 0/1)
df[vars_sel].corr(numeric_only=True)
```

4. Visualización rápida

In [ ]:
```
for c in ["Pregnancies","DiabetesPedigreeFunction"]:
    ax = df[c].dropna().plot(kind="hist", bins=30, alpha=0.75)
    ax.set_title(f"Histograma de {c}")
    ax.set_xlabel(c); ax.set_ylabel("Frecuencia")
    plt.show()

# Distribución binaria de Outcome
ax = df["Outcome"].value_counts().sort_index().plot(kind="bar")
ax.set_title("Distribución de Outcome (0=no, 1=sí)")
ax.set_xlabel("Outcome"); ax.set_ylabel("Conteo")
plt.show()
```

5. Tres consultas sobre los datos

In [ ]:
```
# Q1: Pacientes con ≥5 embarazos y Outcome=1 (diabetes)
q1 = (df.query("Pregnancies >= 5 and Outcome == 1")
        [["Pregnancies","DiabetesPedigreeFunction","Outcome"]]
        .sort_values(["Pregnancies","DiabetesPedigreeFunction"], ascending=F
q1.head(10)
```

In [ ]:
```
# Q2: Pr(Outcome=1) por categorías de número de embarazos
preg_bins = [0,1,3,5,10,100]  # 0, 1–2, 3–4, 5–9, 10+
preg_labels = ["0","1–2","3–4","5–9","10+"]
q2 = (df.assign(preg_cat=pd.cut(df["Pregnancies"], bins=preg_bins, labels=pr
        .groupby("preg_cat")["Outcome"].mean()
        .rename("Pr(Outcome=1)")).to_frame()
q2
```

In [ ]:
```
# Q3: Pr(Outcome=1) por cuartiles de DiabetesPedigreeFunction
q3 = (df.assign(dpf_q=pd.qcut(df["DiabetesPedigreeFunction"], q=4, duplicate
        .groupby("dpf_q")["Outcome"].mean()
        .rename("Pr(Outcome=1)")).to_frame()
q3
```

6. Conclusiones (redacta aquí)

- **Pregnancies:** tipo (discreta), rango, media vs mediana, dispersión; comenta si aumentos en `Pregnancies` elevan la proporción de `Outcome=1` (ver Q2).
- **DiabetesPedigreeFunction:** tipo (continua), rango e IQR; relación con `Outcome` por cuartiles (ver Q3).
- **Outcome:** proporción global de casos (media de Outcome).

7. Exportar a PDF (como se pidió)

En JupyterLab: **File → Print → Save as PDF**.