Estadística No Paramétrica y Pruebas Robustas - Día 5

Change Status

pending 40 min

Learning Objectives

1. Aplicar pruebas no paramétricas cuando fallan los supuestos tradicionales
2. Usar técnicas de remuestreo (bootstrap) para estimación confiable
3. Evaluar robustez de conclusiones estadísticas con métodos alternativos

Practical exercise to apply the concepts learned.

Ejercicio: Análisis robusto de segmentación de clientes con técnicas no paramétricas

Preparación de dataset con características no normales:

```
import pandas as pd
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
from itertools import combinations

# ========================================
# 1. GENERACIÓN DE DATOS (NO NORMALES)
# ========================================
np.random.seed(42)
n_clientes = 300

# Definimos segmentos primero
segmentos_lista = np.random.choice(['Bronce', 'Plata', 'Oro'], n_clientes, p=[0.5, 0.3, 0.2])

# Generamos gasto dependiente del segmento para asegurar diferencias
# Usamos distribución exponencial (no normal) con diferentes escalas (medias)
gasto_data = []
for seg in segmentos_lista:
    if seg == 'Oro':
        val = np.random.exponential(scale=450) # Media 450
    elif seg == 'Plata':
        val = np.random.exponential(scale=250) # Media 250
    else: # Bronce
        val = np.random.exponential(scale=150) # Media 150
    gasto_data.append(val)

df = pd.DataFrame({
    'cliente_id': range(1, n_clientes + 1),
    'segmento': segmentos_lista,
    'gasto_total': gasto_data,
    'frecuencia_visitas': np.random.poisson(3, n_clientes),  # Conteos (Poisson)
    'satisfaccion': np.random.beta(5, 2, n_clientes) * 10,   # Beta (asimetría negativa, alta satisfacción)
    'antiguedad_dias': np.random.exponential(365, n_clientes).astype(int)
})

print("DATASET PARA ANÁLISIS NO PARAMÉTRICO")
print("=" * 40)
print(f"Clientes analizados: {len(df)}")
print("Distribuciones generadas:")
print("- Gasto total: Exponencial (Distinta media por grupo)")
print("- Frecuencia visitas: Poisson (discreta)")
print("- Satisfacción: Beta (acotada 0-10)")

# Verificar no normalidad (Shapiro-Wilk)
print("\nTEST DE NORMALIDAD (Shapiro-Wilk):")
print("-" * 40)
# Solo probamos gasto y satisfacción para no saturar la salida
for col in ['gasto_total', 'satisfaccion']:
    stat, p = stats.shapiro(df[col])
    normal = "SÍ" if p > 0.05 else "NO"
    print(f"{col:20} | Normal: {normal} (p={p:.4e})")

# ========================================
# 2. COMPARACIONES NO PARAMÉTRICAS
# ========================================

# Preparar datos por segmento
segmentos_dict = {}
orden_segmentos = ['Bronce', 'Plata', 'Oro']
for seg in orden_segmentos:
    segmentos_dict[seg] = df[df['segmento'] == seg]['gasto_total'].values

print("\nCOMPARACIÓN DE GASTO TOTAL ENTRE SEGMENTOS")
print("=" * 50)
```

```
# Estadísticas descriptivas robustas (Mediana y Rango Intercuartil)
print(f"{'Segmento':10} | {'Mediana':>10} | {'IQR':>10} | {'n':>5}")
print("-" * 45)
for seg in orden_segmentos:
    datos = segmentos_dict[seg]
    mediana = np.median(datos)
    q25, q75 = np.percentile(datos, [25, 75])
    iqr = q75 - q25
    print(f"{seg:10} | ${mediana:9.0f} | ${iqr:9.0f} | {len(datos):5}")

# Prueba Kruskal-Wallis (ANOVA no paramétrico)
# H0: Las medianas de todos los grupos son iguales
# Desempaquetamos los valores del diccionario en orden
h_stat, p_kw = stats.kruskal(*segmentos_dict.values())

print("\nPRUEBA KRUSKAL-WALLIS (Global):")
print("-" * 30)
print(f"Estadístico H: {h_stat:.3f}")
print(f"Valor p: {p_kw:.4e}") # Notación científica para p muy pequeños
print(f"¿Existen diferencias significativas?: {'SÍ' if p_kw < 0.05 else 'NO'}")

# Comparaciones pareadas con Mann-Whitney U (Post-hoc)
if p_kw < 0.05:
    print("\nCOMPARACIONES PAREADAS (Mann-Whitney U):")
    print("Nota: Se aplica corrección de Bonferroni (alpha = 0.05 / 3 = 0.0167)")
    print("-" * 75)

    alpha_corregido = 0.05 / 3  # Corrección para 3 comparaciones

    # Generamos pares
    pares = list(combinations(orden_segmentos, 2))

    print(f"{'Comparación':<20} | {'U Stat':<10} | {'p-value':<10} | {'Sig? (Bonferroni)'}")
    print("-" * 75)

    for seg1, seg2 in pares:
        # alternative='two-sided' es el estándar
        u_stat, p_mw = stats.mannwhitneyu(segmentos_dict[seg1], segmentos_dict[seg2], alternative='two-sided')

        significativo = "SÍ" if p_mw < alpha_corregido else "NO"
        print(f"{seg1} vs {seg2:<9} | {u_stat:<10.1f} | {p_mw:.4e}   | {significativo}")
else:
    print("\nNo se realizan pruebas post-hoc porque Kruskal-Wallis no fue significativo.")
```

Verificación: ¿Por qué las pruebas no paramétricas son más apropiadas que las paramétricas para este dataset? ¿Cómo afectan los intervalos de confianza bootstrap la interpretación de los resultados?

Requerimientos:

Python con SciPy y NumPy
Pandas para manipulación de datos
Matplotlib para visualizaciones
Jupyter para análisis iterativo