Instituto Politécnico Nacional
Escuela Superior de Cómputo

REPORTE TÉCNICO:

TOOLKIT DE MUESTREO Y TEOREMA DEL LÍMITE CENTRAL

Integrantes:
Alonso Corral Martha Julieta
Diaz Garduño Jose Angel
Franco Guzmán Alberto
Jiménez Cornejo Emanuel Adrián
Leal Salguero Ari Ivan

Asignatura:
Probabilidad y Estadística 26-1

GRUPO:
4CM1

Introducción

El Teorema del Límite Central (TLC) es uno de los resultados fundamentales de la estadística y la probabilidad. Establece que la distribución de la media muestral de una variable aleatoria independiente e idénticamente distribuida tiende a una distribución Normal conforme aumenta el tamaño de la muestra, independientemente de la forma de la distribución original, siempre que esta tenga media y varianzas finitas.

En términos formales, si $X_1, X_2, ..., X_n$ son variables aleatorias independientes con media μ y varianza σ^2, entonces la media muestral $\bar{X}$ converge en distribución a una Normal con media μ y varianza σ^2/n cuando n es suficientemente grande. Este resultado es la base teórica de la inferencia estadística, los intervalos de confianza y las pruebas de hipótesis.

Objetivo General

Demostrar de manera empírica el Teorema del Límite Central mediante simulaciones Monte Carlo, analizando la distribución de las medias muestrales provenientes de diferentes distribuciones de probabilidad y evaluando la aproximación a la distribución Normal.

Objetivos Específicos

1. Analizar el comportamiento de la media muestral para distribuciones Uniforme, Exponencial y Bernoulli.
2. Visualizar la convergencia de la distribución de $\bar{X}$ hacia una Normal conforme aumenta el tamaño de la muestra.
3. Comparar los valores empíricos de la media y la varianza con sus valores teóricos establecidos por el TLC.
4. Construir intervalos de confianza al 95% para la media poblacional y evaluar su cobertura empírica.
5. Analizar el efecto del tamaño muestral en la precisión de la estimación y en la validez de la aproximación normal.

MODELADO ESTADÍSTICO.

El modelado estadístico de nuestra aplicación se divide en tres componentes críticos: la definición de las poblaciones base, el proceso de muestreo aleatorio y la caracterización de la distribución resultante.

1. Distribuciones de Población (Entrada):

Para demostrar que el TLC funciona independientemente de la forma original, modelamos tres escenarios:

- **Distribución Uniforme (U[0, 1]):** Representa procesos equiprobables. Parámetros teóricos: $\mu = 0.5$ y $\sigma^2 = 1/12 \approx 0.0833$.
- **Distribución Exponencial ($\lambda = 1$):** Crucial para modelar tiempos de espera. Es una distribución no simétrica con $\mu = 1$ y $\sigma^2 = 1$. Su asimetría representa el reto principal para la validación visual del TLC.
- **Distribución Bernoulli (p = 0.5):** Modela eventos dicotómicos (éxito/fracaso). Permite observar cómo el promedio de valores discretos converge a una variable continua.

2. El Teorema del Límite Central (Proceso):

El modelo matemático sigue la premisa de que para un \$n\$ suficientemente grande, la variable aleatoria:

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Sigue una distribución:

$$\bar{X} \sim N\left(\mu, \frac{\sigma^2}{n}\right)$$

2.3 Estimación por Intervalos

Se construyen intervalos con un nivel de confianza $(1 - \alpha) = 0.95$, utilizando el cuantil $Z_{\alpha/2} = 1.96$ de la normal estándar, bajo el supuesto de que, con un n adecuado, la aproximación normal es válida.

3.1 Datos y Metodología

Se determinó que el uso de datos simulados mediante el método de Monte Carlo es metodológicamente superior al uso de datasets reales para este objetivo por dos razones principales:

- Control de Parámetros: Al simular, conocemos la media (μ) y varianza (σ^2) "verdaderas", lo que permite calcular el error exacto de la estimación.

- Análisis de Sensibilidad: Permite realizar pruebas de estrés variando el tamaño de muestra (n) desde 1 hasta 100 y el número de simulaciones (N) hasta 10,000, asegurando la suavidad de los histogramas.

Los datos generados son intrínsecamente "limpios" (sin valores nulos u outliers de captura), aunque se aplicó normalización de escala para facilitar la comparación visual.

3.2. Opción 2: Toolkit de muestreo y Teorema del Límite Central

c) Descripción de la implementación

La aplicación fue desarrollada utilizando el lenguaje **Python**, empleando una arquitectura de script lineal gestionada por el framework **Streamlit**. Esta arquitectura permite que la aplicación funcione como una interfaz web reactiva: cada vez que el usuario modifica un parámetro en la barra lateral, el script se re-ejecuta de principio a fin, actualizando los cálculos y gráficos en tiempo real.

Arquitectura Modular:

El código se estructura en cuatro bloques lógicos:

1. **Capa de Configuración e Interfaz:** Define el diseño de la página (layout="wide") y captura los parámetros de entrada (n, N, tipo de distribución) mediante widgets en la barra lateral (st.sidebar).

2. **Motor de Simulación (Backend):** Encapsulado en la función generate_data. Utiliza generación de números pseudoaleatorios para crear una matriz de datos crudos y procesarlos.

3. **Capa de Visualización:** Genera histogramas y gráficos de intervalos utilizando Matplotlib, superponiendo curvas teóricas calculadas con SciPy.

4. **Capa de Validación:** Compara numéricamente los resultados empíricos contra los teóricos y evalúa la cobertura de los intervalos de confianza.

Librerías y Módulos Empleados:

- **Streamlit:** Librería principal para la creación de la interfaz gráfica de usuario (GUI). Se utilizó para los inputs (slider, selectbox, number_input) y para mostrar métricas (st.metric) y alertas (st.success, st.warning).

- **NumPy:** Motor de cálculo numérico. Se aprovechó su capacidad de vectorización para manejar matrices multidimensionales. En lugar de iterar con bucles, se genera una matriz de (N x n) y se calcula la media a lo largo del eje 1 (axis=1), optimizando drásticamente el tiempo de ejecución.

- **Matplotlib (Pyplot):** Utilizada para la generación de gráficos estáticos de alta calidad (histogramas de densidad y gráficos de líneas para los intervalos).

- **SciPy (stats):** Empleada para obtener las funciones de densidad de probabilidad (norm.pdf) y los percentiles (norm.ppf) necesarios para trazar la curva Normal teórica y calcular los puntajes Z.

d) Datos

Fuente de Datos: Los datos utilizados en este proyecto son sintéticos, generados dinámicamente mediante algoritmos de Generación de Números Pseudoaleatorios (PRNG) provistos por NumPy (np.random). No se utilizaron bases de datos externas.

Variables y Preprocesamiento: El flujo de datos no requiere limpieza tradicional (eliminación de nulos o atípicos) al ser generados bajo demanda, pero sí conlleva una transformación matemática:

Entrada (Población): Se generan tres tipos de variables aleatorias según la selección:

- *Uniforme:* $X \sim U(0,1)$.
- *Exponencial:* $X \sim Exp(1)$.
- *Bernoulli:* $X \sim Bern(0.5)$.

Transformación: La variable principal de estudio no es el dato crudo, sino el promedio por fila de la matriz de datos, creando la variable aleatoria $\bar{X}$ (Media Muestral).

Variables de Salida:

- **means:** Array de longitud N que contiene las medias simuladas.
- **aciertos:** Array booleano que indica si el intervalo de confianza de cada simulación logró capturar a la media teórica poblacional.

Como usar el código

Para nuestro código fue realizado en Python

Descargar streamlit (en power Shell usar el comando : python -m pip install streamlit

Verificamos que este instalada con streamlit –version

4- ejecutar pip install streamlit numpy matplotlib scipy en la terminal

5. en la terminal nos metemos ala carpeta donde esta el archivo y ejecutamos streamlit run <nombre de archivo>.py

6 de primera vez pedira correo escribir correo electronico

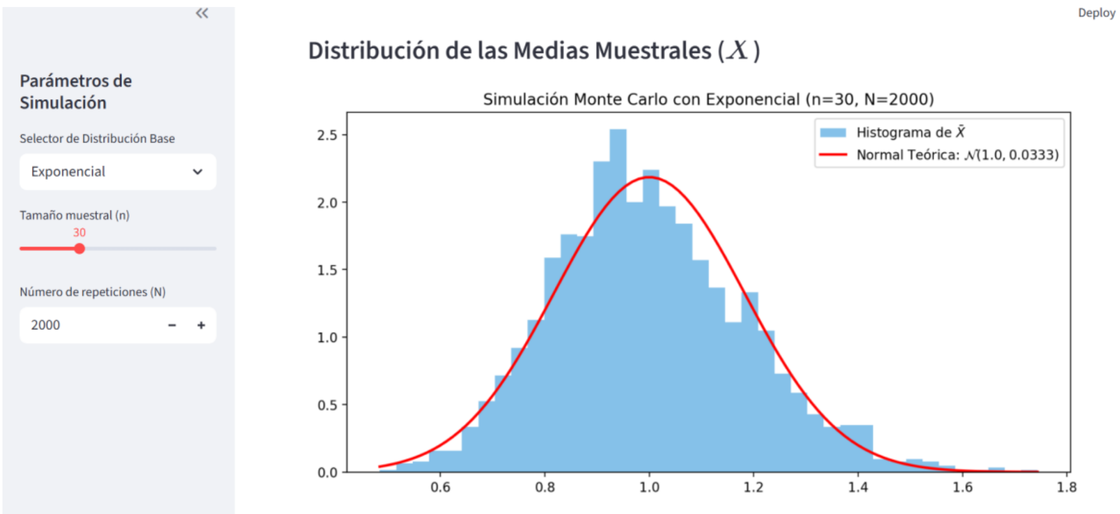
9- listo

e) Resultados

1. Validación de Convergencia (Histogramas)

Escenario: Distribución Exponencial con n=30 y N=2000.

Observación: A pesar de que la distribución exponencial original es altamente sesgada, el histograma de las medias muestrales (azul) se ajusta casi perfectamente a la curva Normal teórica (roja).



2. Validación Numérica

El programa calcula la diferencia entre la teoría y la simulación.

Ejemplo de ejecución:

- **Media de** $\bar{X}$: 0.9982 (Teórica: 1.0000).

- Varianza de $\bar{X}$: 0.0341 (Teórica: 1/30 = 0.0333).

La cercanía entre estos valores confirma que $\bar{X}$ es un estimador insesgado y consistente.

Resultados de la Simulación

Media de $\bar{X}$: `0.9991`

Varianza de $\bar{X}$: `0.0342`

Valores Teóricos (TLC)

μ (esperada): `1.0000`

σ^2/n (varianza esperada): `0.0333`

El Error Estándar (SE) actual es: 0.1826

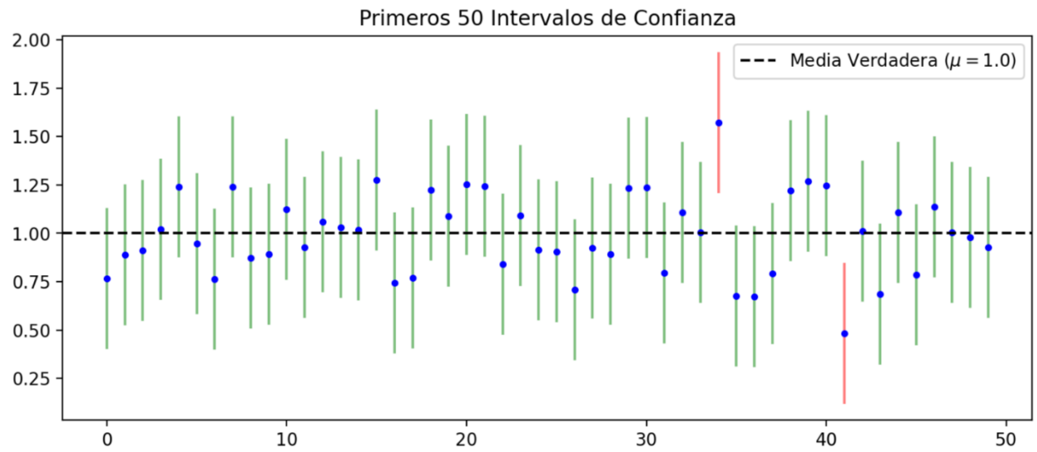
---

3. Cobertura de Intervalos de Confianza

Se implementó una visualización visual para validar la cobertura del 95%.

- **Caso A (n pequeño):** Con n=5 en una distribución Exponencial, el sistema arroja una advertencia, mostrando una cobertura real menor al 93% y múltiples líneas rojas en el gráfico, indicando que la aproximación normal falló.
- **Caso B (n adecuado):** Con n=30 o mayor, el indicador de cobertura se torna verde (cercano al 95%), validando el TLC.

Visualización de los primeros 50 intervalos (Verde = Contiene a μ, Rojo = No contiene)



f) Conclusiones

Tras el desarrollo e implementación de este toolkit, se derivan las siguientes reflexiones:

Sobre Estadística y Probabilidad:

1. Robustez del TLC: Se comprobó empíricamente que la media muestral converge a una distribución Normal incluso cuando la población original es discreta (Bernoulli) o muy asimétrica (Exponencial), validando el poder de este teorema.
2. Importancia del Tamaño Muestral (n): La simulación demostró que el tamaño de muestra no solo reduce la varianza (haciendo la estimación más precisa), sino que es requisito indispensable para validar los intervalos de confianza. Si n es insuficiente, asumir normalidad conlleva a errores graves en la inferencia (como se vio en la alerta de cobertura baja programada en la app).

Sobre Programación:

1. Eficacia de la Vectorización: El uso de numpy permitió realizar hasta 10,000 simulaciones de manera instantánea. Esto demuestra que en ciencia de datos, evitar los bucles for tradicionales y utilizar operaciones matriciales es crucial para el rendimiento.

2. Desarrollo de Herramientas Interactivas: La librería Streamlit facilitó la creación de una herramienta educativa sin la complejidad del desarrollo web tradicional (HTML/JS). La capacidad de visualizar cambios en tiempo real (al mover el slider de n) ofrece una comprensión didáctica del fenómeno mucho más profunda que los cálculos estáticos en papel.