Probabilidad y Estadística para I.O.

Investigación Operativa

Universidad de San Andrés

Fundamentos de Probabilidad

- La probabilidad es fundamental en IO para:
  - Optimizar inventarios con demanda aleatoria
  - Evaluar riesgos en proyectos
  - Modelar tiempos de espera en colas
  - Analizar confiabilidad de sistemas

Conceptos Básicos

- **Experimento aleatorio:** Proceso con resultado impredecible
- **Espacio muestral (Ω):** Conjunto de resultados posibles
- **Evento:** Subconjunto del espacio muestral

Propiedades y Teoremas

Para eventos A y B en Ω:

- $0 \leq P(A) \leq 1$, $P(\Omega) = 1$, $P(\emptyset) = 0$
- $P(A \cup B) = P(A) + P(B) - P(A \cap B)$
- Probabilidad condicional: $P(A|B) = \frac{P(A \cap B)}{P(B)}$
- Independencia: $P(A \cap B) = P(A)P(B)$
- Teorema de Bayes: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

Variables Aleatorias

- **Discretas:** Valores específicos y aislados
  - Llegadas a un banco por hora
  - Productos defectuosos en un lote
  - Intentos hasta primer éxito
- **Continuas:** Valores en rango continuo
  - Tiempo de servicio
  - Peso de un producto
  - Distancia recorrida

Control de Calidad - Binomial

Problema:

Una empresa fabrica lotes de 1200 tornillos. Se sabe que el 3 % de los tornillos son defectuosos. Para controlar la calidad, se toma una muestra aleatoria de 50 tornillos. El lote se rechaza si se encuentran más de 2 tornillos defectuosos en la muestra. **¿Cuál es la probabilidad de que un lote con 3 % de defectuosos sea rechazado?**

Modelo:

- $X \sim \text{Binomial}(50, 0{,}03)$
- $P(X > 2) = 1 - P(X \leq 2)$

Control de Calidad - Solución

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

Para $n = 50$, $p = 0{,}03$:

$$P(X = 0) \approx 0{,}218$$
$$P(X = 1) \approx 0{,}337$$
$$P(X = 2) \approx 0{,}266$$
$$P(X \leq 2) \approx 0{,}821$$
$$P(X > 2) = 0{,}179$$

Control de Calidad - Python

```python
from scipy.stats import binom

# Parametros
n = 50      # Tamano de la muestra
p = 0.03    # Probabilidad de defecto

# Probabilidad de rechazar el lote
prob_rechazo = 1 - binom.cdf(2, n, p)
print(f"Probabilidad de rechazar el lote: {prob_rechazo:.4f}")
```

Problema:

En un centro de distribución, los pedidos llegan con una tasa media de 8 pedidos por hora. El sistema colapsa si se reciben más de 10 pedidos en una hora. **¿Cuál es la probabilidad de que el sistema colapse?**

Modelo:

- $X \sim \text{Poisson}(8)$
- Calcular $P(X > 10)$

Colas - Solución

Para $\lambda = 8$:

$$P(X \leq 10) = \sum_{k=0}^{10} \frac{8^k e^{-8}}{k!} \approx 0{,}815$$
$$P(X > 10) = 1 - 0{,}815 = 0{,}185$$

Colas - Python

```python
from scipy.stats import poisson

# Parametros
lambd = 8   # Tasa de pedidos por hora

# Probabilidad de colapso
prob_colapso = 1 - poisson.cdf(10, lambd)
print(f"Probabilidad de colapso: {prob_colapso:.4f}")
```

Problema:

El tiempo de fabricación tiene una media de 120 minutos y desviación estándar de 15 minutos. Un contrato exige que la pieza se produzca en menos de 100 minutos. **¿Con qué porcentaje de las veces se incumplirá el contrato?**

Modelo:

- $X \sim N(120, 15^2)$
- Calcular $P(X > 100)$

Contratos - Solución

$$\begin{aligned}
P(X > 100) &= 1 - P(X < 100) \\
&= 1 - P\left(Z < \frac{100 - 120}{15}\right) \\
&\Rightarrow 1 - P(Z < -1{,}33) \\
&= P(Z > -1{,}33) \\
&\approx 0{,}9082
\end{aligned}$$

Contratos - Python

```python
from scipy.stats import norm

# Parametros
mu = 120      # Media del tiempo de fabricacion
sigma = 15    # Desviacion estandar

# Probabilidad de incumplir contrato: P(X > 100)
prob_incumplimiento = 1 - norm.cdf(100, loc=mu, scale=sigma)
print(f"Probabilidad de incumplimiento del contrato: {prob_incumplimiento:.4f}")
```

Monte Carlo

Pasos:

1. Identificar variables aleatorias
2. Elegir distribuciones
3. Generar muestras
4. Ejecutar modelo
5. Analizar resultados

Inventario - Problema

Problema:

Una tienda vende un producto cuya demanda diaria sigue una distribución normal con media 80 unidades y desviación estándar 10 unidades. Cada semana decide cuántas unidades pedir.

Costos:

- Mantener stock: \$2/unidad
- Faltante: \$5/unidad
- Venta: \$20/unidad
- Compra: \$10/unidad

Objetivo: Encontrar el nivel de pedido semanal óptimo que maximiza el beneficio esperado.

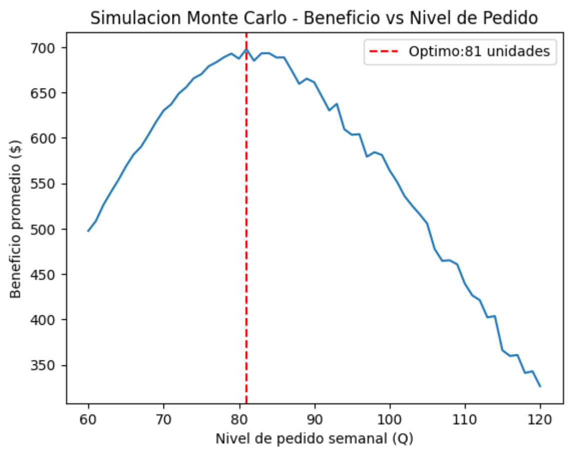
Inventario - Python (Parte 1)

```python
import numpy as np
import matplotlib.pyplot as plt

# Parametros del problema
media_demanda = 80
desvio_demanda = 10
precio_venta = 20
costo_unitario = 10
costo_inventario = 2
costo_faltante = 5
semanas = 1000
np.random.seed(42)  # Para reproducibilidad
niveles_pedido = np.arange(60, 121, 1)  # De 60 a 120 unidades
beneficio_promedio = []
```

Inventario - Python (Parte 2)

```
# Simulacion Monte Carlo
for Q in niveles_pedido:
    demanda_simulada = np.random.normal(media_demanda, desvio_demanda,
         semanas).round().astype(int)
    demanda_simulada = np.maximum(demanda_simulada, 0)  # No hay demanda
         negativa
    ventas = np.minimum(Q, demanda_simulada)
    stock_sobrante = np.maximum(Q - demanda_simulada, 0)
    faltantes = np.maximum(demanda_simulada - Q, 0)
    beneficio = (ventas * precio_venta) - (Q * costo_unitario) - \
                (stock_sobrante * costo_inventario) - (faltantes *
                     costo_faltante)
    beneficio_promedio.append(np.mean(beneficio))

plt.plot(niveles_pedido, beneficio_promedio)
plt.xlabel('Nivel de pedido semanal (Q)')
plt.ylabel('Beneficio promedio ($)')
plt.title('Simulacion Monte Carlo - Beneficio vs Nivel de Pedido')
plt.axvline(niveles_pedido[np.argmax(beneficio_promedio)], color='r',
           linestyle='--', label=f'Optimo:
                {niveles_pedido[np.argmax(beneficio_promedio)]} unidades')
plt.legend()
plt.show()
```

Resultado de la Simulación



¿Preguntas?