Centro de Enseñanza Técnica Industrial

Desarrollo de Software

Actividad 1 - Clase 9

Jesús Alberto Aréchiga Carrillo

22310439 6N

Profesor

Clara Margarita Fernández Riveron

Abril de 2025

Guadalajara, Jalisco

Introducción

En el análisis de datos y la construcción de modelos para inteligencia artificial, resulta esencial comprender cómo se distribuyen las variables aleatorias que describen fenómenos del mundo real. Las distribuciones discretas —como la binomial, que modela el número de éxitos en un número fijo de ensayos, y la de Poisson, que cuenta eventos que ocurren en un intervalo de tiempo o espacio— nos permiten cuantificar la probabilidad de sucesos de conteo. Por otro lado, las distribuciones continuas, especialmente la normal, son fundamentales para modelar variables que toman valores en una escala continua (peso, porcentajes, medidas). Finalmente, el análisis muestral y el cálculo de tamaño de muestra garantizan que nuestras estimaciones de proporciones y medias sean precisas y tengan el nivel de confianza deseado. Juntas, estas herramientas constituyen la base probabilística sobre la cual se edifican técnicas de inferencia, clasificación y detección de anomalías en IA.

Ejercicios:

1. El 2% de los DVD de una determinada marca son defectuosos. Si se venden en lotes de 25 unidades, calcular la probabilidad de que hay como máximo dos defectuosos.

```python
import math

def prob_max_defectuosos(n, p, k_max):
    """
    Calcula la probabilidad de que en un lote de n unidades
    con probabilidad p de defecto, haya como máximo k_max defectuosos.
    """
    return sum(math.comb(n, k) * p**k * (1 - p)**(n - k) for k in
range(k_max + 1))

# Parámetros del problema
n = 25       # tamaño del lote
p = 0.02     # probabilidad de defecto
k_max = 2    # máximo defectuosos

# Cálculo de la probabilidad
prob = prob_max_defectuosos(n, p, k_max)
print(f"P(X ≤ {k_max}) = {prob:.6f}")
```

```
P(X ≤ 2) = 0.986757
```

2. Una empresa conservera enlata alimentos y el peso neto de las latas sigue una distribución normal N(400,10), calcula: a) La probabilidad de que una lata pese menos de 380 gramos. b) La probabilidad de que una lata pese entre 385 y 410 gramos.

```
import math

def normal_cdf(x, mu=0, sigma=1):
    """Función de distribución acumulada de la normal N(mu, sigma)."""
    return 0.5 * (1 + math.erf((x - mu) / (sigma * math.sqrt(2))))

# Parámetros de la distribución
mu = 400      # media
sigma = 10    # desviación estándar

# a) Probabilidad de que una lata pese menos de 380 g
p_less_380 = normal_cdf(380, mu, sigma)

# b) Probabilidad de que una lata pese entre 385 y 410 g
p_between_385_410 = normal_cdf(410, mu, sigma) - normal_cdf(385, mu,
sigma)

# Resultados
print(f"P(X < 380) = {p_less_380:.6f}")
print(f"P(385 < X < 410) = {p_between_385_410:.6f}")
```

```
P(X < 380) = 0.022750
P(385 < X < 410) = 0.774538
```

3. Una compañía aérea observa que el número de componentes que fallan antes de cumplir 100 horas de funcionamiento es una variable aleatoria de Poisson. Si el número promedio de fallos es ocho. Se pide: a) ¿Cuál es la probabilidad de que falle un componente en 25 horas? b) ¿Cuál es la probabilidad de que fallen menos de dos componentes en 50 horas? c) ¿Cuál es la probabilidad de que fallen por lo menos tres componentes en 125 horas?

```
import math

def poisson_pmf(k, lam):
    """Calcula la probabilidad de masa de Poisson P(X=k) con parámetro
lam."""
    return math.exp(-lam) * lam**k / math.factorial(k)

def poisson_cdf(k, lam):
    """Calcula la función de distribución acumulada P(X ≤ k) para
Poisson."""
    return sum(poisson_pmf(i, lam) for i in range(k + 1))
```

```
# Parámetros del problema
tasa_promedio = 8 / 100  # fallos por hora
# Intervalos de tiempo
horas_a = 25
horas_b = 50
horas_c = 125

# Parámetros λ para cada intervalo
lam_a = tasa_promedio * horas_a
lam_b = tasa_promedio * horas_b
lam_c = tasa_promedio * horas_c

# a) P(X=1 en 25 horas)
p_a = poisson_pmf(1, lam_a)

# b) P(X<2 en 50 horas) = P(0) + P(1)
p_b = poisson_cdf(1, lam_b)

# c) P(X≥3 en 125 horas) = 1 - P(X≤2)
p_c = 1 - poisson_cdf(2, lam_c)

# Mostrar resultados
print(f"a) P(X = 1 en {horas_a} h)        = {p_a:.6f}")
print(f"b) P(X < 2 en {horas_b} h)        = {p_b:.6f}")
print(f"c) P(X ≥ 3 en {horas_c} h)        = {p_c:.6f}")
```

```
a) P(X = 1 en 25 h)        = 0.270671
b) P(X < 2 en 50 h)        = 0.091578
c) P(X ≥ 3 en 125 h)        = 0.997231
```

4. La utilización de la tarjeta VISA en operaciones comerciales, en la población de una gran ciudad, sigue en porcentajes una distribución normal de media 4,5 y desviación típica 0,5. Se pide calcular las siguientes probabilidades: a) Que un ciudadano tomado al azar utilice la tarjeta más del 5% en sus operaciones b) Tanto por ciento de la ciudad que utiliza la tarjeta menos del 3,75% c) Porcentaje de operaciones con tarjeta que utiliza el 20% más alto de la población d) Porcentaje de operaciones con tarjeta que utiliza el 10% más bajo de la población e) Porcentaje de operaciones del 80% más próximo a la media.

```
import numpy as np
from scipy.stats import norm

# Parámetros de la distribución
mu = 4.5      # media (% de uso)
sigma = 0.5   # desviación típica

# a) P(X > 5%)
```

```python
p_a = 1 - norm.cdf(5, loc=mu, scale=sigma)

# b) P(X < 3.75%)
p_b = norm.cdf(3.75, loc=mu, scale=sigma)

# c) Umbral para el 20% más alto de la población
umbral_c = norm.ppf(0.80, loc=mu, scale=sigma)

# d) Umbral para el 10% más bajo de la población
umbral_d = norm.ppf(0.10, loc=mu, scale=sigma)

# e) Intervalo central que contiene al 80% más próximo a la media
limite_inferior_e = norm.ppf(0.10, loc=mu, scale=sigma)
limite_superior_e = norm.ppf(0.90, loc=mu, scale=sigma)

# Mostrar resultados
print(f"a) P(uso > 5%)       = {p_a:.6f}")
print(f"b) P(uso < 3.75%)    = {p_b:.6f}")
print(f"c) Límite 20% alto   = {umbral_c:.2f}%")
print(f"d) Límite 10% bajo   = {umbral_d:.2f}%")
print(f"e) Intervalo central 80%: [{limite_inferior_e:.2f}%,
{limite_superior_e:.2f}%]")
```

```
a) P(uso > 5%)       = 0.158655
b) P(uso < 3.75%)    = 0.066807
c) Límite 20% alto   = 4.92%
d) Límite 10% bajo   = 3.86%
e) Intervalo central 80%: [3.86%, 5.14%]
```

5. El peso de un determinado tipo de manzanas fluctúa normalmente con media 150 gramos y desviación típica 30 gramos. Una bolsa de llena con 15 manzanas seleccionadas al azar. ¿Cuál es la probabilidad de que el peso total de la bolsa sea inferior a 2 kilos?

```python
import numpy as np
from scipy.stats import norm

# Parámetros del problema
n = 15          # número de manzanas
mu = 150        # media (g) de cada manzana
sigma = 30      # desviación típica (g) de cada manzana

# Distribución de la suma
mu_sum = n * mu
sigma_sum = np.sqrt(n) * sigma

# Umbral de 2000 g (2 kg)
threshold = 2000
```

```python
# Probabilidad de que el peso total sea < 2000 g
p_total_less_2000 = norm.cdf(threshold, loc=mu_sum, scale=sigma_sum)

print(f"P(peso total < {threshold} g) = {p_total_less_2000:.6f}")
```

```
P(peso total < 2000 g) = 0.015712
```

6. El departamento comercial de una industria alimenticia conoce que 2 de cada 10 consumidores reconocen su producto en una prueba a ciegas. ¿Cuántas pruebas ciegas de sabor deberían hacerse para que la proporción de que los que conocen la marca oscile entre el 16% y el 24% con una probabilidad mínima de 0,8?

```python
import math
from scipy.stats import norm

def calcular_tamano_muestra(p0, delta, probabilidad_deseada):
    """
    Calcula el tamaño de muestra n necesario para que la proporción muestral
    se encuentre dentro de p0 ± delta con probabilidad al menos probabilidad_deseada.
    """
    # Nivel de confianza bilateral
    alpha = 1 - probabilidad_deseada
    # Cuantil z para P(|Z| < z) = probabilidad_deseada
    z = norm.ppf(1 - alpha/2)

    # Fórmula para tamaño de muestra en proporciones
    n = (z**2 * p0 * (1 - p0)) / (delta**2)
    return math.ceil(n)

def main():
    p0 = 0.2               # proporción verdadera (2 de cada 10)
    delta = 0.04           # margen: ±4%
    probabilidad = 0.8     # probabilidad mínima de estar dentro del margen

    n = calcular_tamano_muestra(p0, delta, probabilidad)
    print(f"Número mínimo de pruebas ciegas requeridas: {n}")

if __name__ == "__main__":
    main()
```

```
Número mínimo de pruebas ciegas requeridas: 165
```

Conclusiones:

El dominio de las distribuciones de probabilidad y de las técnicas de muestreo es clave para diseñar experimentos rigurosos y para interpretar correctamente los resultados en proyectos de inteligencia artificial. Al saber elegir y aplicar la distribución adecuada —binomial, Poisson o normal— y al dimensionar correctamente el tamaño de la muestra, podemos construir modelos más robustos, estimar con mayor precisión la incertidumbre y tomar decisiones basadas en evidencia estadística. Estos cimientos probabilísticos no solo facilitan la validación de hipótesis, sino que también potencian la fiabilidad y la eficacia de los sistemas inteligentes en ámbitos tan diversos como la visión por computadora, el procesamiento de lenguaje natural y la analítica predictiva.