Inteligencia Artificial 2025

Lab 2

05.febrero.2025

1. Escribe un código en Python para simular lanzamientos de una moneda en la computadora. Se debe permitir que el usuario elija parámetro $0 < p < 1$ que indica la probabilidad de obtener un éxito en el lanzamiento de la moneda. Hacer lo siguiente:

   a) Obtener, mediante repeticiones, una estimación de la densidad del número de lanzamientos necesarios para obtener el primer éxito. Por ejemplo, simular un experimento de estos N veces. Usar $N = 1000$.

   b) Elaborar 3 visualizaciones de la función de densidad o masa, y cambiando los valores de p.

2. Elaborar una función en Python que permita comparar dos muestras (puede ser dos muestras provenientes de distribuciones teóricas, una teórica y una a partir de datos, o dos muestras provenientes a partir de datos). La función debe mostrar

   a) Las funciones de densidad f_1 y f_2.

   b) Las funciones de distribución F_1 y F_2.

   c) Una gráfica PP (prob-prob).

   d) Una gráfica QQ (quantil-quantil).

   Además, debe calcular la distancia de Kolmogorov-Smirnov (KS), e ilustrar en las gráficas de densidad y de distribución, el punto donde se alcanza esta distancia KS. Realizar una prueba de hipótesis de Kolmogorov-Smirnov para comparar dichas muestras.

   Usar alguno de los experimentos del ejercicio anterior (con un valor p y N fijo), y comparar la distribución obtenida del experimento, contra una muestra generada aleatoriamente de la distribución geométrica

   i) $Geom(p)$,

   ii) $Geom(q)$, para $q = 1.2p$ (cuidar que $0 < q < 1$).

3. La **ley de Benford**, (o ley de Newcomb-Benford, también conocida como la ley del primer dígito), asegura que, en gran variedad de conjuntos de datos numéricos que existen en la vida real, la primera cifra es 1 con mucha más frecuencia que el resto de los números. Además, según crece este primer dígito, menos probable es que se encuentre en la primera posición. Esta ley empírica establece que la probabilidad que el dígito d ($1 \leq d \leq 9$) aparezca como el primer dígito no-nulo en un conjunto de datos está dada por

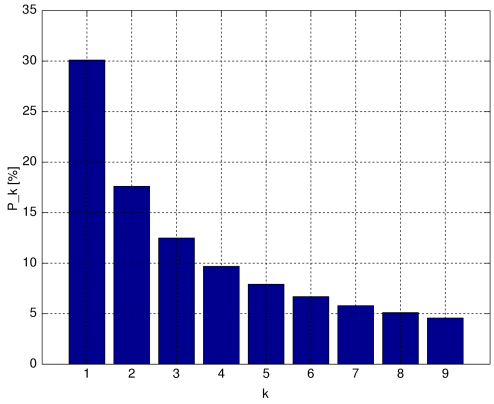
$$\mathbb{P}(X = d) = \log_{10}\left(1 + \frac{1}{d}\right) = \log_{10}(d+1) - \log_{10}(d), \qquad 1 \leq d \leq 9.$$

   El archivo `areas.csv` contiene información de las áreas de todos los países.

   Aplicar las comparaciones del Ejercicio 4, así como la prueba estadística de Kolmogorov-Smirnov para determinar si los datos del primer dígito no-nulo en el conjunto de áreas se comporta de acuerdo a la ley de Benford o no. Explique sus conclusiones.

4. Generar una muestra aleatoria de una distribución gaussiana multivariada de dimensión n (con $n \geq 4$), con una media $\mu \in \mathbb{R}^n$ y covarianza $\Sigma \in \mathbb{R}^{n \times n}$ especificadas por el usuario.
   A partir de la muestra, graficar un `pairplot` que permita visualizar todas las densidades de cada variable y todas las nubes de puntos o densidades bivariadas entre pares de variables.

   Calcular el vector de medias y la covarianza de la muestra aleatoria y verificar que son similares a la media y covarianza teóricas especificadas.



5. Considerar el conjunto de datos `weather.csv`. Se trata de los promedios mensuales de la temperatura (en Celsius) en 35 estaciones canadienses de monitoreo. El interés es comparar las estaciones entre sí con base en sus curvas de temperatura.

   Considerando las 12 mediciones por estación como un vector $\mathbf{x}$, aplicar un análisis de componentes principales. Como $\mathbf{x}$ representa (un muestreo de) una curva, este tipo de datos se llama datos funcionales.

   - Interpretar y dibujar (como curva) los primeros dos componentes p_1 y p_2. Esto es, graficar $\{(i, p_{1i})\}$ y $\{(i, p_{2i})\}$.
   - Agrupar e interpretar las estaciones en el biplot (tener en mente un mapa de Canadá puede ayudar).

6. Históricamente uno de los primeros usos de PCA en el área de procesamiento de imágenes fue como método de compresión. Para ello, si tenemos una imagen de tamaño $H \times W$ píxeles, ésta se subdivide en bloques de $C \times C$ pixeles (por ejemplo, tomar C un factor común de las dimensiones H y W de la imagen). Con los valores de los píxeles en cada bloque se forma un vector

   $$\mathbf{b}_i = (x_1, x_2, \ldots, x_{c^2}) \in \mathbb{R}^{c^2}$$

   La matriz de datos se forma con todos estos vectores provenientes de los bloques $_i$ vectorizados. La compresión consiste en proyectar los datos sobre los primeros k componentes principales, mientras que la decompresión consiste en reconstruir la imagen a su tamaño original $H \times W$ a partir de estas proyecciones.

   Implementar lo anterior para varias imágenes (en escala de gris o a color) y mostrar el efecto del valor de k sobre la calidad de la reconstrucción. (Analizar cómo cambia el error de reconstrucción y la calidad visual a medida que se incrementa o se disminuye k). Mostrar los resultados obtenidos de al menos 3 imágenes.

No se olviden de redactar su informe, y subirlo a Canvas en formato .pdf.