Probabilidad y Estadística - LCC

Trabajo Práctico Final 2021

Problema 1

Se tiene un suceso para el cual la probabilidad de éxito es *p*. Se realizan intentos independientes de este suceso hasta obtener *k* éxitos consecutivos. Considere el proceso N_k que denota el número de ensayos necesarios para obtener k éxitos consecutivos.

a. Simule un cierto número de trayectorias del proceso N_k considerando un valor de $p < 0.5$ e indique en cada caso cuántos realizaciones de la experiencia fueron necesarias hasta obtener k éxitos consecutivos. (Elija un valor de k que satisfaga la siguiente desigualdad $1< k < 5$).

b. A partir de los realizado en a) estime, de ser posible, la $E(N_k)$.

Problema 2

Considere el proceso $D_n = 2I_n - 1$, donde D_n representa el cambio de posición de una partícula que se mueve a lo largo de una linea recta con saltos de magnitud 1 en cada momento, ya que el mismo depende del proceso Bernoulli I_n : la señal emitida en el momento n es correcta.

a. Simule 50 pasos de una trayectoria de D_n para un proceso Bernoulli con $p > 0.7$.

b. Simule una realización del proceso S_n : posición de la partícula en el momento n.

Problema 3

Considere un jugador que tiene una apuesta inicial de k dólares y repetidamente apuesta 1 dólar en un juego en el cual la probabilidad de ganar es p y la probabilidad de perder es 1-p. La fortuna sucesiva del jugador es una simple caminata aleatoria iniciada en k (estado inicial del proceso $X_0 = k$). Supongamos que el jugador decide detenerse cuando su fortuna alcanza S dólares ($S > k$), o cae a 0 dólar, lo que ocurra primero.

a. Siimule y visualice la evolución del capital del jugador para un valor de k y S pero para distintos valores de p ($p < 0.5$, $p=0.5$ y $p> 0.5$).

b. Estime, mediante la simulación de un número adecuado de trayectorias del capital del jugador, la probabilidad de ruina de dicho jugador para los distintos escenarios planteados en a).

Nota: Este es el clásico problema de la ruina del jugador, discutido por primera vez por los matemáticos Blaise Pascal y Pierre Fermat en 1656 y el cual sirve para describir muchos fenómenos similares en la actualidad.

Problema 4

Suponga que deseamos ordenar en forma creciente un conjunto de *n* valores distintos: $x_1, \ldots, x_n$. Un procedimiento eficiente para lograrlo es el algoritmo *Quick-Sort*, el cual está definido recursivamente como sigue:

- ✓ Cuando $n = 2$ el algoritmo compara los dos valores y los ordena adecuadamente.
- ✓ Cuando $n > 2$ el algoritmo comienza por elegir aleatoriamente uno de los *n* valores, por ejemplo: x_i y luego compara cada uno de los otros n -1 valores con x_i, determinando cuáles de ellos son menores y cuáles son mayores que x_i.

Si se denota con $\underline{S}_i$ al conjunto de elementos menores que x_i y con S_i al conjunto de elementos mayores que x_i, el algoritmo ordena los conjuntos $\underline{S}_i$ y S_i y el orden final, por lo tanto, consiste de los conjuntos ordenados de elementos en $\underline{S}_i$, luego x_i, y finalmente el conjunto ordenado de los elementos de S_i.

Por ejemplo, supongamos que el conjunto de elementos es {10, 5, 8, 2, 1, 4, 7}. Se elige uno de los valores aleatoriamente (esto significa, cada uno de los 7 valores tiene probabilidad 1/7 de ser elegido). Suponga que el valor 4 es el elegido. Luego se compara el 4 con cada uno de los otros de valores y se obtiene:

{2, 1}, 4, {10, 5, 8, 7}

Ahora se ordena el conjunto {2, 1} para obtener:

1, 2, 4, {10, 5, 8, 7}

Se continúa eligiendo un nuevo valor aleatoriamente del conjunto {10, 5, 8, 7} – por ejemplo: 7 – y se compara cada uno de los otros tres valores con 7, y se obtiene:

1, 2, 4, 5, 7, {10, 8}

Finalmente, se ordena el conjunto {10, 8}, logrando:

1, 2, 4, 5, 7, 8, 10

Una forma de medir la efectividad de este algoritmo es a través del número esperado de comparaciones que hace. Se denota con M_n al número esperado de comparaciones necesarias para ordenar un conjunto de n valores distintos utilizando el algoritmo *quick-sort* Mediante el uso de simulaciones, calcule que valor M_n , para un conjunto de n números. Considere un valor de n que satisfaga $3 < n < 7$.

a. Simule el algoritmo *Quick-Sort*, para ordenar un conjunto de 7 números y calcule el número promedio de comparaciones

b. Explicite en forma analítica cómo puede obtener ese resultado sin recurrir al proceso de simulación.

Problema 5

El poder de los motores de búsqueda de Internet reside en su capacidad para responder a la consulta de un usuario con una lista ordenada de sitios web clasificados por importancia y relevancia. El corazón del motor de búsqueda de Google es la familia de algoritmos *PageRank,* que asignan un valor de importancia a cada página web, llamado rango de página. Google interpreta un enlace de una página *A* a una página *B* como un voto, de la página *A*, para la página *B*. No solo se tiene en cuenta el volumen de votos, o enlaces que una página recibe; también se analiza la página que emite el voto. Los votos emitidos por las páginas consideradas "importantes", es decir con un *PageRank* elevado, valen más, y ayudan a hacer a otras páginas "importantes". Por lo tanto, el *PageRank* de una página refleja la importancia de la misma en Internet. Aunque el algoritmo real de *PageRank* es complejo y tiene muchos aspectos técnicos (además de detalles secretos para intentar impedir que alguien manipule el rango de su página web), el rango de página de una página web particular se describe fácilmente por medio de un modelo estocástico.

Considere un usuario web hipotético que navega a través de Internet moviéndose de una página a otra al azar. Cuando el usuario está en una página web en particular, elige uno de los enlaces de hipertexto disponibles en esa página de manera uniforme al azar y luego pasa a esa página. El modelo puede describirse como una caminata aleatoria por parte del internauta en un gráfico gigante llamado el *webgraph*. En el gráfico web, los vértices (nodos) son páginas web. El vértice “x” se une al vértice “y” por un arco dirigido si hay un enlace de hipertexto en la página “x” que conduce a la página “y”. Cuando la persona que está navegando se encuentra en el vértice “x”, elige al azar un arco del conjunto de arcos disponibles. Imagine que el internauta ha estado caminando aleatoriamente por la web durante mucho tiempo, y se calcula la probabilidad de que esté en la página “x”. Esta probabilidad a largo plazo es precisamente el rango de página de la página x. Intuitivamente, la probabilidad a largo plazo de estar en una página en particular tenderá a ser mayor para las páginas con más enlaces entrantes y más pequeños para páginas con pocos enlaces, y es una medida de la importancia o popularidad de una página. El algoritmo de *PageRank* se puede entender como una asignación de probabilidades a cada sitio en la web. La Figura 1 muestra una red simplificada de siete páginas.

a. Modele el comportamiento de visitas a las páginas como una cadena de Markov.

b. Simule 100 pasos para visualizar una trayectoria de este proceso.

c. Determine el rango de cada página.

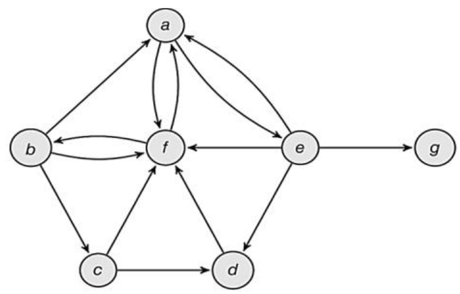

Figura 1. Red simplificada.

Importante: La figura 1 No corresponde al grafo de la cadena.

Problema 6

La Figura 2 muestra una trayectoria del proceso de conteo en el cual los eventos ocurren en los tiempos t_1, t_2, t_3, t_4, t_5. Suponga que el evento bajo estudio es el número de vuelos que recibe un aeropuerto a una tasa λ (aterrizajes/hora).

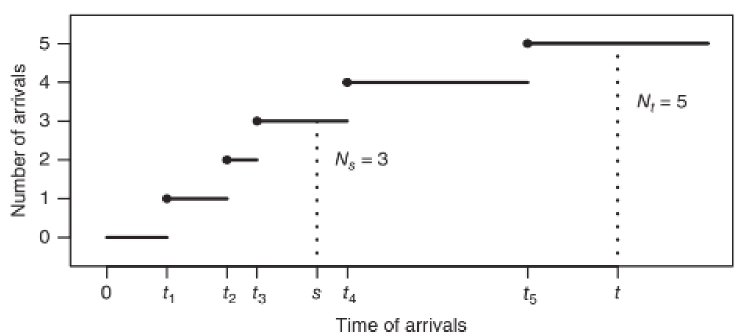

Figura 2. Trayectoria de proceso de conteo.

a. El aeropuerto recibe vuelos comenzando a las 2 am a una tasa de 3 aterriizajes por hora, de acuerdo a un Proceso Poisson. Simule el comportamiento de dicho proceso durante 24 horas y grafique la trayectoria obtenida.

b. Simule una trayectoria muestral durante un intervalo de tiempo suficientemente largo [0,T], Calcule los tiempos entre llegadas. Represente gráficamente los tiempos observados entre llegadas a través de un histograma y comente cuál es el modelo apropiado para para describir esa variable.

Probabilidad y Estadística - LCC
Trabajo Práctico Final 2021

Problema 7

En aplicaciones de seguridad informática, un *honeypot* (o sistema trampa) es una herramienta dispuesta en una red o sistema informático para ser el objetivo de un posible ataque informático, y así poder detectarlo y obtener información del mismo y del atacante. Los datos del *honeypot* son estudiados utilizando cadenas de markov. Se obtienen datos desde una base de datos central y se observan ataques contra cuatro puertos de computadoras - 80, 135, 139 y 445- durante un año. Los estados de la cadena de Markov son los cuatro puertos y se incluye un nodo indicando que ningún puerto está siendo atacado. Los datos de monitorean semanalmente y el puerto más atacado durante la semana es guardado. La matriz de transición para la cadena estimada para los ataques semanales es:

$$
\boldsymbol{P} = \begin{array}{c} \\ 80 \\ 135 \\ 139 \\ 445 \\ \text{No} \end{array}
\begin{array}{c}
\begin{array}{ccccc} 80 & 135 & 139 & 445 & \text{No attack} \end{array} \\
\left(\begin{array}{ccccc}
0 & 0 & 0 & 0 & 1 \\
0 & 8/13 & 3/13 & 1/13 & 1/13 \\
1/16 & 3/16 & 3/8 & 1/4 & 1/8 \\
0 & 1/11 & 4/11 & 5/11 & 1/11 \\
0 & 1/8 & 1/2 & 1/8 & 1/4
\end{array}\right)
\end{array},
$$

con distribución inicial $\pi_0 = (0,0,0,0,1)$.

a. Después de tres semanas, ¿cuáles son los puertos con más y menos probabilidad de ser atacados?

b. Encuentre la distribución límite (si es que existe) de los puertos atacados. Justifique.

Este TP se finaliza con las respuestas a las preguntas que se realicen el día del examen final. Además puede requerir una defensa oral si los miembros de la cátedra lo consideran necesario.