Taller de distribuciones de muestreo y pruebas de hipótesis
Análisis de Datos en Ingeniería
Universidad del Norte

May 25, 2024

Objetivo El taller es la primera parte del corte final y consta de 3 objetivos:

- Observar experimentalmente, mediante la creación, descripción y análisis de datos sintéticos (obtenidos por un modelo de simulación), el cumplimiento del teorema del límite central aplicado para el cálculo de la distribución de muestreo de la media.
- Analizar experimentalmente el concepto de riesgo de una prueba de hipótesis.
- Afianzar el uso de herramientas informáticas para el análisis y manipulación de datos. Específicamente, el uso del lenguaje de programación Python y sus librerías para visualización de datos (Matplotlib y/o Seaborn), manipulación y almacenamiento de datos (Pandas y Numpy) y librerías científicas (Scipy).

Contexto En este taller, se analizarán los datos generados por un simulador que modela la operación de un sistema de entrega de paquetes en una ciudad utilizando el concepto de "Crowdshipping". El simulador toma información básica como el tiempo diario de operación del sistema, tiempo máximo de entrega por paquete, número promedio de paquetes diarios, una razón oferta (viajeros de la multitud) - demanda (paquetes) que determina el número promedio diario de viajeros de la multitud disponibles (crowdshippers), costos unitarios de envío por paquete (para la multitud) y para couriers profesionales, capacidad de los casilleros y tipo de topología de la red (balanceada o no balanceada). Estos parámetros del modelo son definidos en el archivo `masterfile.csv`. El código simula, de acuerdo con una distribución de probabilidad, la entrada de paquetes al sistema que se mueven desde cualquiera y hacia cualquier nodo de una red de casilleros ubicados estratégicamente en la ciudad. De la misma manera, el código simula la llegada aleatoria de viajeros de la multitud que viajan desde y hacia cualquier nodo de la red de casilleros. El algoritmo simula, de acuerdo con una política determinada, la decisión del sistema de enrutar un paquete con un viajero de la multitud con la opción de poder luego enrutarlo con otro viajero de la multitud o con un courier profesional, a mayor costo, si el destino de un viajero no coincide con el destino final del paquete. El algoritmo entrega dos salidas (entre otras) necesarias para el análisis: los tiempos simulados de llegada (horas) de cada paquete y cada viajero de la multitud (crowd) y su par origen-destino en los archivos: `outputs_parr.csv` para paquetes y `outputs_carr.csv` para viajeros de la multitud. Estos archivos de salida serán los datos fuente para el análisis. El archivo ´masterfile.csv´ define dos experimentos distintos, uno para cada tipo de topología de la red. Los resultados de cada experimento son mostrados en carpetas separadas. Estos resultados se encontrarán en de la carpeta ´50_nodes´ dentro de las sub-carpetas ´Experiment1´ y ´Experiment2´, respectivamente. Para los pasos 1 al 7 deben utilizar sólo los resultados para el experimento 1, el cual es con una topología

de red balanceada. Para los pasos 8 y 9, deben hacer lo que se pide con ambos experimentos, el de red balanceada (experimento 1) y el de red no balanceada (experimento 2).

Paso 0: Correr la simulación para generar los archivos de salida Deben correr la simulación de tal manera que obtengan como mínimo 500 mil (500000) observaciones de cada variable. Para garantizar esto deben decidir cuánto tiempo de simulación o cuántas réplicas de la simulación correr. Estos dos parámetros se modifican en el archivo `masterfile.csv` en los campos `Time_per_day` y `num_replications`.

Paso 1: Lectura de Archivos CSV Leer los archivos que resgistran los tiempos simulados de llegada de paquetes (`outputs_parr.csv`) y los tiempos de llegada de viajeros de la multitud (`outputs_carr.csv`). Guardar estos datos en data frames. Utilizar la función `read_csv` de pandas para leer los archivos en formato .csv.

Paso 2: Separar Origen y Destino Cada archivo `.csv` tiene un campo con un texto entre paréntesis que muestra una tupla escrita como ('origen', 'destino') y otro campo numérico con el tiempo de llegada en horas. Extraer el text correspondiente al origen y al destino del texto del archivo ('origen', 'destino') y escribirlos en dos campos distintos. El data frame resultante debería tener un campo para el origen (texto), un campo para el destino (texto) y un campo para el tiempo de llegada. Usar la función `extract` y `strip` de pandas.

Paso 3: Calcular el Tiempo entre Llegadas Crear un nuevo campo para cada data frame que contiene la información de paquetes y viajeros que calcule el tiempo entre llegadas de cada paquete (o viajero). El campo de tiempo original es el tiempo total transcurrido cuando el paquete (o viajero) llegó. Ordenar el data frame de acuerdo con el tiempo de llegada y luego crear un campo que calcule el tiempo entre llegadas, es decir, la diferencia entre el tiempo de llegada de un paquete (o viajero) y el paquete (o viajero) anterior.

Paso 4: Análisis de la Distribución del Tiempo entre Llegadas Analizar la distribución de la variable "tiempo entre llegadas" para cada variable (tiempo entre llegadas de paquetes y viajeros) guardadas en los data frames respectivos. Realizar la descripción estadística de cada variable, es decir, presentar estadísticos descriptivos de cada muestra (media, Q1, Q2, Q3, desviación estándar, coeficiente de asimetría y de curtosis). Adicionalmente, graficar en matplotlib el histograma de frecuencias relativas y el diagrama de cajas para cada variable en un **mismo lienzo** de matplotlib. Discutir acerca de las medidas descriptivas incluyendo simetría y sesgo. Comparar la distribuciones de ambas variables y discutir diferencias y similitudes entre ambas. Teniendo en cuenta los estadísticos más adecuados y el contexto del sistema simulado, discutir acerca de la capacidad del sistema para suplir la demanda a través de la multitud de viajeros en la red. Usar la función `describe` de pandas para calcular estadísticos descriptivos y la función `hist` y `boxplot` de matplotlib para construir los histogramas y diagramas de caja, respectivamente. Crear un archivo de salida `.png` que muestre el gráfico creado en matplotlib, el archivo debe estar guardado en una subcarpeta con el nombre `outputs`. Ver método `savefig` de matplotlib.

Paso 5: Plantear Hipótesis sobre la Distribución y Realizar Prueba de Bondad de Ajuste Plantear una hipótesis acerca de la distribución de probabilidad de cada variable. Basándose en la forma de las distribuciones observadas y los estadísticos descriptivos, plantear una hipótesis sobre la distribución de probabilidad de cada variable (por ejemplo, distribución exponencial, normal, log-normal, etc.). Luego, realizar una prueba de bondad de ajuste Kolmogorov-Smirnov para verificar si los tiempos entre llegadas siguen la distribución planteada. Usar la función `kstest` de Scipy.

Paso 6: Creación de data frame de Muestras Aleatorias Desde el data frame respectivo que contiene las (N) observaciones de tiempos entre llegadas de paquetes y viajeros, extraer aleatoriamente (M=10000) muestras de tamaño (n=50). Construir un data frame (para cada variable) que contenga ordenadas las (M) muestras como columnas y las (n) observaciones en las filas.

Paso 7: Análisis de las Medias Muestrales y Prueba de Hipótesis Calcular la media muestral para cada una de las (M) muestras, guardando sus valores en un vector que puede ser una lista, un array de NumPy o un data frame de pandas. Describir la distribución del conjunto de las (M) medias muestrales y compararla con la distribución de frecuencias de los datos de tiempo entre llegadas originales. ¿Qué puede concluir respecto a la distribución de probabilidad de las medias muestrales? ¿Qué diferencias hay entre la distribución de muestreo de la media y la distribución de probabilidad de la variable original (población)? ¿Esto es evidencia empírica de alguna teoría vista en clase (soportada por algún teorema)?. Luego, con cada muestra extraída de **cada variable**, realizar una prueba de hipótesis para inferir si el tiempo promedio entre llegadas al sistema de crowdshipping es de 2 segundos. Establecer un nivel de significancia, y ejecutar las ($M \times 2$) pruebas de hipótesis. Calcular la proporción de pruebas de hipótesis que resultaron en rechazo y no rechazo. **Proporcionar una interpretación de sus resultados** asumiendo cada uno de los siguientes escenarios:

- Para la variable Tiempo entre llegada de paquetes:

  Escenario a: En realidad llegan en promedio alrededor de 25198 paquetes en un día de operación de 14 horas (equivalente a un promedio de 2 segundos entre llegadas de paquetes al sistema).

  Escenario b: En realidad llegan en promedio 20000 paquetes en un día de operación de 14 horas.

- Para la variable Tiempo entre llegada de viajeros:

  Escenario a: En realidad llegan en promedio alrededor de 25198 viajeros en un día de operación de 14 horas (equivalente a un promedio de 2 segundos entre llegadas de paquetes al sistema).

  Escenario b: En realidad llegan en promedio 30000 viajeros en un día de operación de 14 horas.

Planteamiento de Hipótesis

- **Hipótesis nula (H0)**: El tiempo promedio entre llegadas es de 2 segundos.
- **Hipótesis alternativa (H1)**: El tiempo promedio entre llegadas no es de 2 segundos.

Paso 8: Conteo de Paquetes y Viajeros por Origen-Destino Para cada variable de tiempo entre llegadas, contar el número de paquetes (y de viajeros) que se observaron para cada par origen-destino, es decir, una matriz que muestre el numero de paquetes y viajeros, respectivamente, que viajaron desde cada origen y hacia cada destino. Guardar en data frames nuevos. **Escribrir los resultados** en archivos `.csv` separados en la carpeta `outputs`.

Paso 9: Data frame con Paquetes por Ubicación y Mapa de Calor Para cada variable, crear data frames que muestren el número total de paquetes que salen de cada ubicación en una columna y en la otra el número total de paquetes que llegaron a esa ubicación. Usar los nombres de las estaciones de tren subterráneo de Múnich (U-Bahn) como índices de cada fila en cada data frame. Luego, utilizando las coordenadas de estas estaciones (que pueden consultar con Gmaps, por ejemplo), construir un mapa de calor en matplotlib que ilustre el número total de paquetes que salieron y entraron en cada ubicación, es decir, la distribución geográfica de del flujo de paquetes y de viajeros en la ciudad, respectivamente. Crear un gráfico separado para cada variable. Guardar los gráficos en archivos `.png` separados en la carpeta `outputs`.

Instrucciones

- El taller debe ser realizado en grupos de máximo 4 estudiantes.
- comentar el código y utilizar markdown para dar una descripción del procedimiento ejecutado luego celda de código si es relevante, o si hay que dar y argumentar alguna respuesta.
- Numerar los pasos en el Jupyter Notebook (o Colab notebook) de acuerdo con la organización de este documento.
- Escribir un reporte donde describan punto a punto el procedimiento realizado y los análisis descriptivos con las conclusiones que saquen en cada punto (si aplica).
- La fecha de entrega es a más tardar el miércoles 06 de junio de 2024.
- Abriré un link en Brightspace para la entrega del reporte con sus soportes. Los archivos a enviar con sus nombres respectivos son:
  - `reporte` (`.doc` o `.pdf`)
  - `notebook.ipynb`

    Dentro de una sub-carpeta `outputs`:
  - `distribucion_tiempos.png`
  - `paquetes_o-d.csv`
  - `viajeros_o-d.csv`
  - `mapa_calor_paquetes.png`
  - `mapa_calor_viajeros.png`
- Adjuntar 1 sólo archivo comprimido en `.zip`.