## **Simulación de demanda y ventas con inventario para canales físicos y digitales**

### **1. Carga de archivos**

* Cargar archivo ventas\_tiendas.csv: contiene ventas físicas por producto, tienda y día.
* Cargar archivo ventas\_zonas.csv: contiene ventas digitales por producto, zona y día.
* Cargar archivo resumen\_tiendas.csv: contiene mejor ajuste de distribución por producto para canal físico.
* Cargar archivo resumen\_zonas.csv: contiene mejor ajuste de distribución por producto para canal digital.
* Cargar archivo reorden.csv: contiene punto de reorden y frecuencia de revisión por tienda y producto.

### **2. Inicialización de inventario**

Para cada par producto-tienda en reorden.csv:

* Inicializar inventario = reorden.

### **3. Cálculo de estadísticas**

**Ventas físicas:**

Para cada combinación de (producto, tienda) en ventas\_tiendas.csv:

* Calcular media, desviación estándar y mínimo de venta\_tienda.
* Guardar como tabla de estadísticas físicas.

**Ventas digitales:**

Para cada combinación de (producto, zona) en ventas\_zonas.csv:

* Calcular media, desviación estándar y mínimo de venta\_digital.
* Guardar como tabla de estadísticas digitales.

### **4. Simulación de demanda**

**Para ventas físicas:**

Para cada fila en la tabla de estadísticas físicas:

* Obtener:
  + id\_producto
  + id\_tienda
  + media, desviación, mínimo
  + distribución desde resumen\_tiendas según id\_producto
* Según el tipo de distribución (normal, poisson, gamma, lognorm):
  + Generar 40 valores aleatorios simulados
  + Redondear y asegurar que no sean negativos
* Crear tabla con columnas:
  + id\_producto, id\_tienda, día (1 al 40), venta\_tienda (simulada)

**Para ventas digitales:**

Para cada fila en la tabla de estadísticas digitales:

* Obtener:
  + id\_producto
  + id\_zona
  + media, desviación, mínimo
  + distribución desde resumen\_zonas según id\_producto
* Generar 40 valores simulados
* Redondear y asegurar que no sean negativos
* Crear tabla con columnas:
  + id\_producto, id\_zona, día (1 al 40), venta\_digital (simulada)

### **5. Simulación con inventario – Canal físico**

Para día en 1 a 40:

* Para cada producto y tienda:
  + Simular demanda diaria usando la distribución asignada y sus parámetros
  + **Si demanda ≤ inventario**:
    - venta\_tienda = demanda
    - venta\_completa = "Sí"
  + **Si demanda > inventario**:
    - venta\_tienda = inventario
    - venta\_completa = "No"
  + **Actualizar inventario**:
    - inventario -= venta\_tienda
  + **Revisión de stock**:
    - Si día % frecuencia\_reorden == 0: reponer hasta reorden
  + Guardar en tabla de resultados:
    - id\_producto, id\_tienda, día, demanda\_simulada, venta\_tienda, inventario, venta\_completa

### **6. Simulación con inventario – Canal digital (inventario por tienda)**

Para día en 1 a 40:

* Para cada zona y producto:
  + Simular demanda según distribución y parámetros
  + Buscar tienda asociada: tienda = tienda\_zona[id\_zona]
  + Obtener inventario disponible en esa tienda para ese producto
  + **Si demanda ≤ inventario**:
    - venta\_digital = demanda
    - venta\_completa = "Sí"
  + **Si demanda > inventario**:
    - venta\_digital = inventario
    - venta\_completa = "No"
  + **Actualizar inventario**:
    - inventario -= venta\_digital
* **Reposición**:
  + Para cada tienda y producto:
    - * Si día % frecuencia\_reorden == 0 : reponer hasta reorden
* Guardar resultado del día:
  + id\_producto, id\_zona, día, demanda\_simulada, venta\_digital, inventario, venta\_completa

### **7. Exportar resultados**

Exportar resultados a:

* simulacion\_ventas\_tienda\_inventario.csv
  + Columnas: id\_producto, id\_tienda, día, demanda\_simulada, venta\_tienda, inventario, venta\_completa
* simulacion\_ventas\_digital\_inventario.csv
  + Columnas: id\_producto, id\_zona, día, demanda\_simulada, venta\_digital, inventario, venta\_completa

## **Control de entrega inmediata vs. entrega al día siguiente**

En esta etapa se incorpora la probabilidad de que un cliente reciba su producto el mismo día o al día siguiente. Esta decisión se basa en una función lineal que depende del nivel de demanda simulada para ese producto.

### **Función de probabilidad de entrega inmediata**

Se utiliza la siguiente función:

* A mayor demanda, menor es la probabilidad de entrega inmediata.
* La probabilidad se acota entre 0 y 1 para evitar valores inválidos.

### **Lógica de decisión por fila simulada**

Para cada fila simulada (producto, tienda/zona, día):

1. Calcular la probabilidad:
   1. P\_entrega = max(0, min(1, -0.0152 \* demanda\_simulada + 0.9))
2. Generar un número aleatorio:
   1. u ∈ [0, 1]
3. Definir tipo de entrega:
   1. Si u <= P\_entrega: entrega = "inmediata"
   2. Si u > P\_entrega: entrega = "día siguiente"

### **Columna adicional en los resultados**

Se agrega una nueva columna entrega a los archivos de simulación:

* simulacion\_ventas\_tienda\_inventario.csv
* simulacion\_ventas\_digital\_inventario.csv

La **mejor forma de reordenar** sería aplicar un **modelo (R, S)**, donde:

[ S = percentil 95 de la demanda acumulada en cualquier ventana de 5 días en múltiples simulaciones ]

Esta política garantiza un nivel de servicio alto y toma en cuenta la incertidumbre natural de la demanda, por lo tanto, es:

* Robusta frente a variabilidad,
* Sostenible en el tiempo,
* Ajustable si cambian tus condiciones logísticas.

Puedes simular su desempeño y ajustar el percentil si notas exceso o insuficiencia de inventario.

**Ruteo:**

Solo Clarke-Wright, es en sí una heurística Greedy

Clarke-Wright después 2-opt (se puede)

Greedy despues 2 opt

**CLARKE & WRIGHT**

En términos generales va seleccionando rutas para añadir, y lo hace en base a cuanto más se gana por hacer las rutas juntas, que por hacerlas por separado.

Su paso a paso es ordenar los pares de rutas de mayor a menor beneficio, sin importar como afectará la solución global después, y así va caso a caso hasta llegar una solución localmente óptima.

### **Entrada:**

* D: ubicación del depósito (corresponde a la tienda asignada).
* N: conjunto de órdenes del día (cada cliente con ubicación y volumen total del pedido).
* d(i,j): matriz de distancias entre clientes y tienda.
* cap: capacidad máxima del vehículo (volumen total que puede cargar).
* volumen\_total(i): volumen total del pedido del cliente i (calculado previamente sumando los productos comprados por el cliente i).

### **Paso 1: Calcular ahorros**

Para cada par (i, j) en N, con i ≠ j:

→ ahorro(i,j) ← d(D,i) + d(D,j) - d(i,j)

Guardar los pares (i,j) con su ahorro en una lista "savings".

### **Paso 2: Ordenar ahorros en orden descendente**

Ordenar "savings" por ahorro(i,j) de mayor a menor.

### **Paso 3: Inicializar rutas vacías**

* rutas ← []
* nodos\_asignados ← ∅

### **Paso 4: Construir rutas usando la lista de ahorros**

Para cada par (i,j) en "savings":

* Si i y j no están en ninguna ruta:
  + Si volumen\_total(i) + volumen\_total(j) ≤ cap:
    - Crear nueva ruta [i, j]
    - Agregar a rutas
    - Agregar i y j a nodos\_asignados
* Si solo uno de i o j está en una ruta y es extremo:
  + Si se puede agregar el otro sin exceder cap:
    - Agregar el nodo al inicio o fin de la ruta correspondiente
    - Agregar el nuevo nodo a nodos\_asignados
* Si ambos i y j están en rutas diferentes y ambos son extremos:
  + Si se puede fusionar sin exceder cap:
    - Unir las dos rutas
    - Eliminar las originales de rutas
    - Agregar la nueva ruta fusionada

### **Paso 5: Asignar nodos no cubiertos**

Para cada nodo k ∈ N que no está en nodos\_asignados:

* Crear ruta [k]
* Agregar a rutas

### **Paso 6: Agregar depósito**

Para cada ruta r en rutas:

* r ← [D] + r + [D]

### **Salida:**

* rutas: lista de rutas completas desde y hacia el depósito D (tienda).

<https://www.researchgate.net/publication/265778994_A_savings_based_method_for_real-life_vehicle_routing_problems/link/6565c169b86a1d521b162382/download?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6Il9kaXJlY3QiLCJwYWdlIjoicHVibGljYXRpb24iLCJwcmV2aW91c1BhZ2UiOiJxdWVzdGlvbiJ9fQ>