Ciencia de Datos | EmTech – FUNED

Proyecto Final Fundamentos de Programación con Python

DataSapiens

Marco Polo Bravo Montiel

5 de diciembre 2021

Contenido

Introducción

La tienda virtual *LifeStore* maneja una amplia gama de artículos. Se nos ha contratado porque han notado que tienen una acumulación importante de inventario y consideran que las búsquedas de sus productos han disminuido. De esta manera, nos han entregado registros con su información: *lifestore_products, lifestore_sales, lifestore_searches*. El primer registro contiene información sobre los productos (id, nombre, precio, categoría y *stock*); el segundo, sobre sus ventas (id, producto vendido, reseña, fecha y si hubo devolución o no); el tercero, sobre las búsquedas que se han realizado de esos productos (id y producto buscado).

Así, se nos ha solicitado analizar la información con tres objetivos principales:

- Productos más vendidos y productos rezagados
- Productos por reseña en el servicio
- Sugerir una estrategia

De igual forma, analizaremos los ingresos y ventas, tanto anuales, como mensuales. Así, se atenderán las inquietudes de la empresa, se analizará su situación actual y se dará una sugerencia para atender su problema.

El código desarrollado para este proyecto se encuentra en el siguiente repositorio: https://github.com/Marcony1/Proyecto-Final-Fundamentos

Definición del Código

La primera parte del código está orientada a responder la preguntas “¿qué productos tienen mayores/menores ventas?” y “¿qué productos tienen mayores/menores búsquedas?”. Sin embargo, antes se le solicita al usuario que introduzca su ID. De colocar “admin”, se le dará acceso. En caso contrario, se le pedirá que intente de nuevo.

```
"""
This is the LifeStore_SalesList data:

lifestore_searches = [id_search, id product]
lifestore_sales = [id_sale, id_product, score (from 1 to 5), date, refund (1 for true or 0 to f
lifestore_products = [id_product, name, price, category, stock]
"""

> lifestore_products = [ ···
]

> lifestore_sales = [ ···
]

> lifestore_searches = [ ···
]


from operator import itemgetter

######## INICIO DE LOGIN #########

entrada = input("Ingrese su nombre de usuario ")
usuario_correcto = False

while usuario_correcto == False:
    if entrada == "admin":
        print("Bienvenido")
        usuario_correcto = True
    else:
        entrada = input("Lo siento, usuario incorrecto. Por favor, intente de nuevo ")
        usuario_correcto = False

######### FIN DE LOGIN ###########
```

En la captura anterior, se aprecian las primeras líneas del código. En la parte superior, contamos con la descripción de los registros que la tienda *LifeStore* nos ha proporcionado. Posteriormente, se encuentran los registros (colapsados) y, finalmente, comienza el código con el que trabajaremos. Primero, hemos importado “*itemgetter*” de la biblioteca “*operator*”. Posteriormente, necesitaremos indicarle al programa de qué manera ordenar los elementos dentro de listas (en este caso se tratará de listas de listas). Después, le solicitamos el ID al usuario y declaramos como “*False*” una variable *booleana* que nos ayudará a verificar la

identidad del usuario. Una vez que el usuario haya introducido el ID, entraremos a un *while* del que solo se podrá salir cuando la variable *booleana* sea verdadera (cuando el usuario haya dado el ID correcto). En este caso, el ID correcto será "admin". En caso de introducir el incorrecto, se le solicitará de forma indefinida su ID. Una vez que haya introducido el correcto, se le dará la bienvenida al usuario.

```
1447
1448  print("\nBienvenido al reporte de LifeStore.")
1449  input("\nA continuación, se mostrarán los 5 productos más vendidos. Presione ENTER para continu
1450
1451
1452  # Creamos tabla para contar ventas de los productos #
1453  lista_ventas = []
1454  for i in range(1, len(lifestore_products) + 1):
1455      lista_ventas.append([i, 0])
1456
1457
1458  cantidad_registros_ventas = len(lifestore_sales)
1459
1460  for i in range(0, cantidad_registros_ventas):
1461      producto = lifestore_sales[i][1]
1462      lista_ventas[producto - 1][1] = lista_ventas[producto - 1][1] + 1
1463
1464  ######### Productos Más Vendidos ########
1465  from operator import itemgetter
1466  lista_ordenada = sorted(lista_ventas, key = itemgetter(1), reverse = True)
1467
1468  print("\n********************")
1469  print("Los productos más vendidos son: \n")
1470  for i in range(0, 5):
1471      print(lifestore_products[lista_ordenada[i][0] - 1][1])
1472      print("(" + str(lista_ordenada[i][1]) + " unidades vendidas) \n")
1473
1474  input("Presione ENTER para ver los 5 productos menos vendidos ")
1475
```

Una vez aceptado, se le dará la bienvenida al usuario y se le irá mostrando un reporte con datos de la tienda. El usuario deberá presionar la tecla "ENTER" para ir avanzando por el reporte. Posteriormente, el programa creará una lista vacía llamada "lista_ventas". Los elementos con los que la llenaremos también serán listas. "lista_ventas" tendrá la misma cantidad de elementos (listas) que la cantidad de elementos que tiene el registro "lifestore_products". Cada una de las sublistas a agregar, tendrá dos elementos: un id y un 0 (que será actualizado posteriormente). De esta forma, estamos creando una clase de "tabla" en la que tenemos una columna con todos los id's de los productos y una columna (temporalmente en ceros) en la que contaremos cuántas veces se vendió cada producto.

Posteriormente, se obtiene la cantidad de elementos del registro "*lifestore_sales*" y se guarda en la variable "cantidad_registros_ventas". De este modo, se utiliza un *for* para revisar cada uno de dichos registros y extraer el producto que se vendió en cada uno de ellos. Una vez extraído eso, se actualiza la tabla "lista_ventas" y se le suma "1" al producto que se extrajo del registro. En pocas palabras, "lista_ventas" funciona como un contador de la frecuencia de cada uno de los productos.

Más adelante, ordenaremos esta lista de listas. Ordenaremos las sublistas tomando en cuenta su segundo elemento (aquel que lleva la frecuencia del producto). Por lo que importamos "*itemgetter*". Guardaremos la lista ordenada en "lista_ordenada". Al usar la función *sorted*, le indicamos a la función que se fije en el índice 1 (el segundo elemento de las sublistas), que es la frecuencia de cada producto y establecemos *reverse = True*, es decir, que no ordene de menor a mayor, sino que haga lo contrario: ordenar de mayor a menor.

Ya ordenada la información, ésta se presenta. Por medio de un *for*, imprimimos los productos de los 5 primeros elementos de la lista ordenada. Extraemos el id del producto de la "lista_ordenada" y lo introducimos como índice en el registro "*lifestore_products*" para poder extraer el nombre del producto. Después, se imprime la frecuencia de nuestra "lista_ordenada". Una vez hecho esto, se le dice al usuario que presione ENTER para revisar, ahora, los productos menos vendidos.

```
1474  input("Presione ENTER para ver los 5 productos menos vendidos ")
1475
1476  ######### Productos Menos Vendidos ########
1477  from operator import itemgetter
1478  lista_ordenada = sorted(lista_ventas, key = itemgetter(1), reverse = False)
1479
1480  print("\n*******************")
1481  print("Los productos menos vendidos son: \n")
1482  for i in range(0, 5):
1483      print(lifestore_products[lista_ordenada[i][0] - 1][1])
1484      print("(" + str(lista_ordenada[i][1]) + " unidades vendidas) \n")
1485
1486  input("Presione ENTER para continuar ")
1487
1488  input("\nA continuación, se mostrarán los 10 productos más buscados. Presione ENTER para contir
1489
```

En este caso, el proceso es exactamente el mismo. Lo que se hace es que, en los parámetros de "*sorted*", se establece *reverse = False*, por lo que ahora los elementos se ordenarán de

manera inversa. Posteriormente, el código es prácticamente idéntico. Una vez más, se le pide al usuario dar *click* a "ENTER" para continuar.

```
######### Productos Más Buscados ########

# Creamos lista de búsquedas
lista_busquedas = []
for i in range(1, len(lifestore_products) + 1):
    lista_busquedas.append([i, 0])


cantidad_registros_busquedas = len(lifestore_searches)

for i in range(0, cantidad_registros_busquedas):
    producto = lifestore_searches[i][1]
    lista_busquedas[producto - 1][1] = lista_busquedas[producto - 1][1] + 1


lista_ordenada2 = sorted(lista_busquedas, key = itemgetter(1), reverse = True)


print("\n********************")
print("Los productos más buscados son: \n")
for i in range(0, 10):
    print(lifestore_products[lista_ordenada2[i][0] - 1][1])
    print("(" + str(lista_ordenada2[i][1]) + " búsquedas realizadas) \n")

input("Presione ENTER para ver los 10 productos menos buscados ")
```

La siguiente sección del reporte consiste en mostrar un análisis similar, pero para los productos más y menos buscados. Nuevamente, se crea una lista vacía "lista_busquedas" y se le agregan listas vacías. La estrategia va a volver a ser crear un contador para la cantidad de búsquedas de cada producto. Para esto, se obtiene la cantidad de registros de "*lifestore_searches*" y, por medio de un *for*, su busca a través de sus elementos extrayendo los id's de los productos y anotando la frecuencia en "lista_busquedas". Al igual que en la sección anterior, esta lista se ordenará y se reportarán los productos con mayores búsquedas (en este caso, se muestran 10). Otra vez, al mostrar los resultados, se le pide al usuario presionar la tecla "ENTER".

```
######### Productos Menos Buscados ########
from operator import itemgetter
lista_ordenada2 = sorted(lista_busquedas, key = itemgetter(1), reverse = False)

print("\n********************")
print("Los productos menos buscados son: \n")
for i in range(0, 10):
    print(lifestore_products[lista_ordenada2[i][0] - 1][1])
    print("(" + str(lista_ordenada2[i][1]) + " búsquedas realizadas) \n")

input("Presione enter para continuar ")

```

Para obtener los productos con menores búsquedas, el código es análogo. Solo se establece *reverse = False*. Después de esto, el usuario deberá presionar "ENTER" para pasar a la siguiente sección.

```
#### Productos por Reseñas ###

input("\nA continuación, se mostrarán los 5 productos con mejores reseñas. Presione ENTER para

# Creamos tabla para contar reseñas positivas de los productos #
lista_resenas = []
for i in range(1, len(lifestore_products) + 1):
    lista_resenas.append([i, 0, 0, 0, 0, 0]) # Contaremos reseñas de 1 a 5


cantidad_registros_resenas = len(lifestore_sales)

for i in range(0, cantidad_registros_resenas):
    producto = lifestore_sales[i][1]
    resena = lifestore_sales[i][2]
    lista_resenas[producto - 1][resena] = lista_resenas[producto - 1][resena] + 1

lista_ordenada3 = sorted(lista_resenas, key = itemgetter(5, 4, 3, 2, 1), reverse = True)


print("\n********************")
print("Los productos con mejores reseñas son: \n")
for i in range(0, 5):
    print(lifestore_products[lista_ordenada3[i][0] - 1][1])
    print("\n* " + str(lista_ordenada3[i][5]) + " reseñas de 5 estrellas")
    print("* " + str(lista_ordenada3[i][4]) + " reseñas de 4 estrellas")
    print("* " + str(lista_ordenada3[i][3]) + " reseñas de 3 estrellas")
    print("* " + str(lista_ordenada3[i][2]) + " reseñas de 2 estrellas")
    print("* " + str(lista_ordenada3[i][1]) + " reseñas de 1 estrella \n\n")

input("Presione ENTER para ver los 5 productos con peores reseñas ")
```

En la siguiente sección, se responde a la pregunta "¿qué productos tienen mejores/peores reseñas?" Para esto, se sigue una estrategia similar. Podría decirse que se recicla el código

anterior. Se crea una lista vacía llamada “lista_resenas” y se le agregarán sublistas con 6 elementos. El primero corresponderá al id de cada producto y los posteriores 5 serán contadores de la cantidad de reseñas (de 1 a 5) que recibió el producto. Posteriormente, se analiza la dimensión del registro “*lifestore_sales*” para revisar cada una de sus entradas a través de un *for*. En esta revisión, se extraerán el id del producto y la calificación obtenida. Estos datos serán utilizados para actualizar la cuenta en “lista_resenas”. Finalmente, volveremos a ordenar dicha lista, pero, esta vez, indicaremos el criterio de desempates a *itemgetter*. Queremos ordenarlas en función de cuántas reseñas de 5 estrellas tienen, posteriormente, considerar las de 4 estrellas… y así hasta llegar a una estrella. Ya ordenada la lista, no hay más que imprimir la información relevante con la ayuda de un nuevo *for*. Al final, se vuelve a pedir al usuario que dé *click* a “ENTER”.

```
# Creamos tabla para contar reseñas negativas de los productos #
lista_resenas_malas = []
for i in range(1, len(lifestore_products) + 1):
    lista_resenas_malas.append([i, 0, 0, 0, 0, 0, 0]) # Contaremos reseñas de 1 a 5
    # Incorporamos columna para devoluciones

cantidad_registros_resenas_malas = len(lifestore_sales)

for i in range(0, cantidad_registros_resenas_malas):
    producto = lifestore_sales[i][1]
    resena = lifestore_sales[i][2]
    lista_resenas_malas[producto - 1][resena] = lista_resenas_malas[producto - 1][resena] + 1
    if lifestore_sales[i][4] == 1:
        lista_resenas_malas[producto - 1][6] = lista_resenas_malas[producto - 1][6] + 1

lista_ordenada4 = sorted(lista_resenas_malas, key = itemgetter(1, 6, 2), reverse = True)


print("\n********************")
print("Los productos con peores reseñas son: \n")
for i in range(0, 5):
    print(lifestore_products[lista_ordenada4[i][0] - 1][1])
    print("\n* " + str(lista_ordenada4[i][5]) + " reseñas de 5 estrellas")
    print("* " + str(lista_ordenada4[i][4]) + " reseñas de 4 estrellas")
    print("* " + str(lista_ordenada4[i][3]) + " reseñas de 3 estrellas")
    print("* " + str(lista_ordenada4[i][2]) + " reseñas de 2 estrellas")
    print("* " + str(lista_ordenada4[i][1]) + " reseñas de 1 estrella \n")

    print("(Devoluciones: " + str(lista_ordenada4[i][6]) + ") \n\n")

input("Presione enter para continuar ")
```

Para determinar los productos con peores reseñas, el proceso es muy similar. La única diferencia la encontraremos en la forma en que ordenamos con ayuda de *itemgetter*. En este caso, daremos prioridad a las reseñas de 1 estrella. El criterio de desempate serán las

devoluciones y, posteriormente, se tomarán en consideración las reseñas de 2 estrellas. La información es presentada y se le pide al usuario que dé “ENTER”.

```
#### Análisis de Ventas por Mes ###
input("\nA continuación, se mostrará información relativa a las ventas. Presione ENTER para cor


# Creamos tabla para los meses, las ventas y los ingresos #
lista_ventas_meses = []
for i in range(1, 13):
    if i < 10:
        lista_ventas_meses.append(["0" + str(i), 0, 0]) # Contaremos cantidad de ventas y ganar
    else:
        lista_ventas_meses.append([str(i), 0, 0])
cantidad_registros_ventas_meses = len(lifestore_sales)

for i in range(0, cantidad_registros_ventas_meses):
    producto = lifestore_sales[i][1]
    mes = lifestore_sales[i][3][3:5]
    if lifestore_sales[i][4] != 1: # No contamos las que tengan devolución
        for j in range(0, 12):
            if mes == lista_ventas_meses[j][0]:
                lista_ventas_meses[j][1] = lista_ventas_meses[j][1] + 1
                lista_ventas_meses[j][2] = lista_ventas_meses[j][2] + lifestore_products[produc

total_ventas = 0
total_ingreso = 0

for i in range(0, len(lista_ventas_meses)):
    total_ventas = total_ventas + lista_ventas_meses[i][1]
    total_ingreso = total_ingreso + lista_ventas_meses[i][2]

promedio_ventas_mes = total_ventas/12
promedio_ingreso_mes = total_ingreso/12

print("\n\nLa cantidad total de ventas en el año es de: ", total_ventas)
print("La cantidad total de ingresos en el año es de: $", total_ingreso)
print("La cantidad promedio de ventas por mes es: ", promedio_ventas_mes)
print("La cantidad promedio de ingresos por mes es de: $", promedio_ingreso_mes)
```

En la siguiente sección, se analizan las ventas por mes. Se crea una lista vacía y se introducirán listas con tres elementos: id del mes, contador de ventas y acumulado de ingresos. El id se agregará como tipo de dato “*string*”. Éste se construirá con los índices de los respectivos meses. Si el índice del *for* es menor a 10, se agregará un 0 antes del índice, y esto será el id del mes a agregar en las sublistas. Si el índice del *for* es mayor o igual a 10, ya no se agrega este 0.

En este caso, se analizará el registro de las ventas “*lifestore_sales*”. Para cada elemento de dicha lista (ese elemento es una lista también), se extraerá el id del producto, posteriormente, el mes de la venta y, al final, se verificará si no hubo una devolución. En

caso de no haberla, esta información se agregará en nuestra lista contador que, en este caso, llamaremos “lista_ventas_meses”. Ahí, agregaremos la cantidad de ventas por mes y el acumulado por mes.

Posteriormente, con nuestra “lista_venta_meses” llena, se hará la suma de los segundos y terceros elementos de las sublistas. De esta manera, se obtendrán las ventas totales y los ingresos acumulados totales. Bastará dividirlos entre 12 para obtener los promedios mensuales. Todo esto se reportará al usuario y, nuevamente, se le pedirá que dé “ENTER”.

```
1634
1635  print("\n\nA continuación se mostrará información relativa a los meses en los que se realizaron
1636  input("\nPresione ENTER para continuar ")
1637
1638  meses = ["Enero", "Febrero", "Marzo", "Abril", "Mayo",
1639   "Junio", "Julio", "Agosto", "Septiembre", "Octubre", "Noviembre", "Diciembre"]
1640
1641  # Ordenamos por cantidad de ventas
1642  lista_ordenada5 = sorted(lista_ventas_meses, key = itemgetter(1), reverse = True)
1643
1644
1645  print("\n\nDe mayor a menor cantidad de ventas, los meses quedan ordenados de la siguiente form
1646
1647  for i in range(0, 12):
1648      print(meses[int(lista_ordenada5[i][0]) - 1], "con " + str(lista_ordenada5[i][1]) + " ventas
1649
1650
1651  # Ordenamos por cantidad de ingreso
1652  lista_ordenada6 = sorted(lista_ventas_meses, key = itemgetter(2), reverse = True)
1653
1654
1655  print("\n\nDe mayor a menor cantidad de ingreso, los meses quedan ordenados de la siguiente for
1656
1657  for i in range(0, 12):
1658      print(meses[int(lista_ordenada6[i][0]) - 1], "con $" + str(lista_ordenada6[i][2]) + " de in
1659
1660
1661
1662  print("\nGracias por revisar el reporte de LifeStore. ¡Hasta Pronto!\n\n\n")
1663
1664
1665
```

Finalmente, en la última parte, creamos una lista con los nombres de los meses. Vincularemos a estos nombres a los números del mes (ej. Enero = 01). Sin embargo, como “Enero” tiene el índice “0” en la lista “meses”, al extraer el id del mes de la “lista_ventas_meses”, tendremos que convertirla en *int* y, posteriormente, restarle 1 para que “01” se convierta en “0” y extraiga de forma correcta el nombre “Enero” de la lista “meses”.

En fin, trabajaremos con la misma “lista_ventas_meses” y la ordenaremos de mayor a menor tomando como referencia la cantidad de ventas, y también haremos lo mismo para los ingresos acumulados. Después de esto, con ayuda de un *for*, reportaremos esto al usuario. Es decir, imprimiremos (de mayor a menor) los meses junto con la cantidad correspondiente a cada uno. Después de esto, el programa agradece al usuario por haberlo usado.

Solución al Problema

Las preguntas planteadas son respondidas por el programa al ejecutarlo. A continuación, mostraremos los resultados arrojados por la terminal y, posteriormente, plantearemos las sugerencias necesarias para dar solución a la situación en la que se encuentra *LifeStore*.

```
Ingrese su nombre de usuario admin
Bienvenido

Bienvenido al reporte de LifeStore.

A continuación, se mostrarán los 5 productos más vendidos. Presione ENTER para continuar

********************
Los productos más vendidos son:

SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm
(50 unidades vendidas)

Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth
(42 unidades vendidas)

Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake)
(20 unidades vendidas)

Tarjeta Madre ASRock Micro ATX B450M Steel Legend, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD
(18 unidades vendidas)

SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm
(15 unidades vendidas)

Presione ENTER para ver los 5 productos menos vendidos
```

```
********************
Los productos menos vendidos son:

Procesador Intel Core i3-8100, S-1151, 3.60GHz, Quad-Core, 6MB Smart Cache (8va. Generación - Coffee Lake)
(0 unidades vendidas)

Tarjeta de Video EVGA NVIDIA GeForce GT 710, 2GB 64-bit GDDR3, PCI Express 2.0
(0 unidades vendidas)

Tarjeta de Video EVGA NVIDIA GeForce GTX 1660 Ti SC Ultra Gaming, 6GB 192-bit GDDR6, PCI 3.0
(0 unidades vendidas)

Tarjeta de Video EVGA NVIDIA GeForce RTX 2060 SC ULTRA Gaming, 6GB 192-bit GDDR6, PCI Express 3.0
(0 unidades vendidas)

Tarjeta de Video Gigabyte NVIDIA GeForce GTX 1650 OC Low Profile, 4GB 128-bit GDDR5, PCI Express 3.0 x16
(0 unidades vendidas)

Presione ENTER para continuar

A continuación, se mostrarán los 10 productos más buscados. Presione ENTER para continuar
```

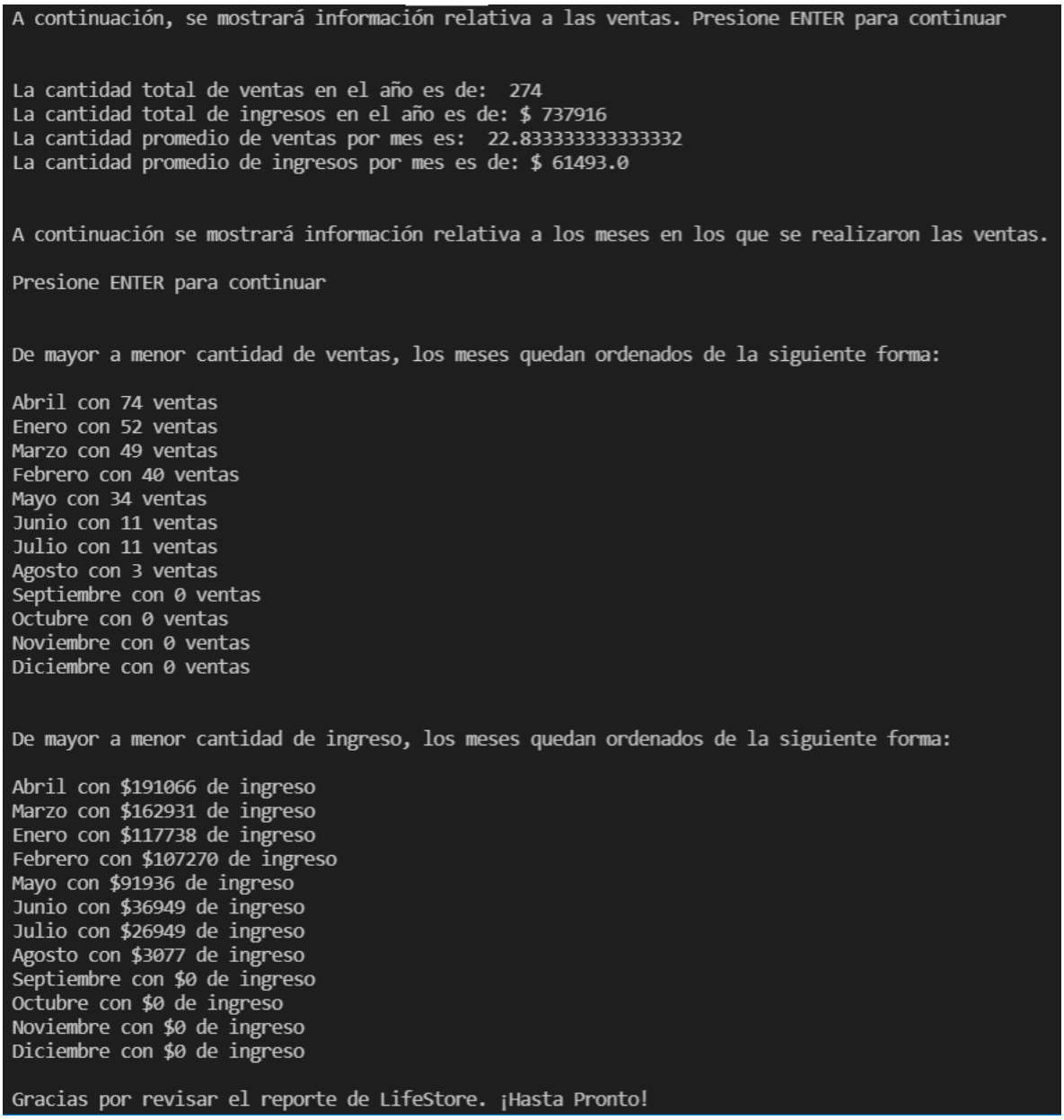
```
********************
Los productos más buscados son:

SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm
(263 búsquedas realizadas)

SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm
(107 búsquedas realizadas)

Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD
(60 búsquedas realizadas)

Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth
(55 búsquedas realizadas)

Procesador AMD Ryzen 3 3200G con Gráficos Radeon Vega 8, S-AM4, 3.60GHz, Quad-Core, 4MB L3, con Disipador Wraith Spire
(41 búsquedas realizadas)

Logitech Audífonos Gamer G635 7.1, Alámbrico, 1.5 Metros, 3.5mm, Negro/Azul
(35 búsquedas realizadas)

TV Monitor LED 24TL520S-PU 24, HD, Widescreen, HDMI, Negro
(32 búsquedas realizadas)

Procesador Intel Core i7-9700K, S-1151, 3.60GHz, 8-Core, 12MB Smart Cache (9na. Generación Coffee Lake)
(31 búsquedas realizadas)

Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake)
(30 búsquedas realizadas)

SSD XPG SX8200 Pro, 256GB, PCI Express, M.2
(30 búsquedas realizadas)

Presione ENTER para ver los 10 productos menos buscados
```

```
******************
Los productos menos buscados son:

Tarjeta de Video EVGA NVIDIA GeForce GT 710, 2GB 64-bit GDDR3, PCI Express 2.0
(0 búsquedas realizadas)

Tarjeta de Video EVGA NVIDIA GeForce RTX 2060 SC ULTRA Gaming, 6GB 192-bit GDDR6, PCI Express 3.0
(0 búsquedas realizadas)

Tarjeta de Video Gigabyte NVIDIA GeForce GTX 1650 OC Low Profile, 4GB 128-bit GDDR5, PCI Express 3.0 x16
(0 búsquedas realizadas)

Tarjeta de Video Gigabyte NVIDIA GeForce RTX 2060 SUPER WINDFORCE OC, 8 GB 256 bit GDDR6, PCI Express x16 3.0
(0 búsquedas realizadas)

Tarjeta de Video MSI Radeon X1550, 128MB 64 bit GDDR2, PCI Express x16
(0 búsquedas realizadas)

Tarjeta de Video PNY NVIDIA GeForce RTX 2080, 8GB 256-bit GDDR6, PCI Express 3.0
(0 búsquedas realizadas)

Tarjeta Madre AORUS ATX Z390 ELITE, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel
(0 búsquedas realizadas)

Tarjeta Madre ASRock Z390 Phantom Gaming 4, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel
(0 búsquedas realizadas)

Tarjeta Madre ASUS ATX PRIME Z390-A, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel
(0 búsquedas realizadas)

Tarjeta Madre ASUS ATX ROG STRIX B550-F GAMING WI-FI, S-AM4, AMD B550, HDMI, max. 128GB DDR4 para AMD
(0 búsquedas realizadas)

Presione enter para continuar
```

```
********************
Los productos con mejores reseñas son:

SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm

* 38 reseñas de 5 estrellas
* 11 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 1 reseñas de 2 estrellas
* 0 reseñas de 1 estrella


Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth

* 35 reseñas de 5 estrellas
* 6 reseñas de 4 estrellas
* 1 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
* 0 reseñas de 1 estrella


Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake)

* 14 reseñas de 5 estrellas
* 6 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
* 0 reseñas de 1 estrella


SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm

* 13 reseñas de 5 estrellas
* 2 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
* 0 reseñas de 1 estrella


Tarjeta Madre ASRock Micro ATX B450M Steel Legend, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD

* 10 reseñas de 5 estrellas
* 8 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
```

```
* 10 reseñas de 5 estrellas
* 8 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
* 0 reseñas de 1 estrella


Presione ENTER para ver los 5 productos con peores reseñas

********************
Los productos con peores reseñas son:

Tarjeta Madre AORUS micro ATX B450 AORUS M (rev. 1.0), S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD

* 0 reseñas de 5 estrellas
* 1 reseñas de 4 estrellas
* 1 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
* 4 reseñas de 1 estrella

(Devoluciones: 3)


Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD

* 8 reseñas de 5 estrellas
* 4 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
* 2 reseñas de 1 estrella

(Devoluciones: 1)


Tarjeta de Video Gigabyte AMD Radeon R7 370 OC, 2GB 256-bit GDDR5, PCI Express 3.0

* 0 reseñas de 5 estrellas
* 0 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
* 1 reseñas de 1 estrella

(Devoluciones: 1)
```

```
Tarjeta de Video Gigabyte AMD Radeon R7 370 OC, 2GB 256-bit GDDR5, PCI Express 3.0

* 0 reseñas de 5 estrellas
* 0 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
* 1 reseñas de 1 estrella

(Devoluciones: 1)


Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151, Intel H110, 32GB DDR4, para Intel

* 0 reseñas de 5 estrellas
* 0 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 0 reseñas de 2 estrellas
* 1 reseñas de 1 estrella

(Devoluciones: 1)


Tarjeta Madre Gigabyte micro ATX GA-H110M-DS2, S-1151, Intel H110, 32GB DDR4 para Intel

* 0 reseñas de 5 estrellas
* 0 reseñas de 4 estrellas
* 0 reseñas de 3 estrellas
* 1 reseñas de 2 estrellas
* 0 reseñas de 1 estrella

(Devoluciones: 1)


Presione enter para continuar

A continuación, se mostrará información relativa a las ventas. Presione ENTER para continuar
```



Como se puede apreciar, ciertos procesadores y discos de estados sólidos contribuyen a la mayor cantidad de ventas y reseñas positivas. Por el contrario, parece que las tarjetas de video no se están vendiendo bien. La gente parece estar muy interesada en discos de estado sólido, tarjetas madre, televisores, audífonos y ciertos procesadores. Por el contrario, no parecen interesarse por las tarjetas de video y ciertas tarjetas madre. Esto no solo se puede apreciar a través de las búsquedas, sino que también a través de las reseñas dadas. Por lo que parece que el negocio debería enfocarse en aquellos productos que contribuyen a la mayor parte de

sus ingresos y poner en liquidación, así como también dejar de adquirir, en la misma cantidad en la que lo viene haciendo, aquellos productos que no son populares y reciben quejas.

Se puede apreciar que la tienda vende, en promedio 29 productos al mes y percibe cerca de $60,000 en ganancias al mes. Sin embargo, al analizar las ventas por meses, nos podemos dar cuenta de que estos promedios ocultan la tendencia descendente (en ambas, la cantidad vendida y el ingreso percibido) de la venta de los productos de la tienda, mes con mes. Esta situación se torna crítica en los últimos 4 meses del año, en los que no vende ni percibe ningún tipo de ingreso (o, en su defecto, vende, pero le devuelven). Esto explica el motivo por el cuál la cantidad en *stock* ha ido en aumento. Los últimos 4 meses sin ventas son un fuerte incentivo para dejar de surtirse y enfocarse en vender o liquidar el producto con el que cuentan. Tendrán que asumir una pérdida ya que, esta tendencia descendente muestra la depreciación que sufren los productos tecnológicos al acercarse un nuevo año y nuevos productos más modernos. Por lo que *LifeStore* tendrá que invertir en publicidad y liquidar su *stock* acumulado al mismo tiempo que tendrá que surtirse de productos más recientes. De cualquier manera, el que los últimos meses no se haya vendido prácticamente nada es algo alarmante y se tendrá que revisar a fondo si se trata de una cuestión tecnológica que está impidiendo que los usuarios puedan acceder al sitio *web*.

Conclusión

LifeStore, tienda dedicada a la venta de productos tecnológicos, está pasando por un periodo de tiempo en el que sus ventas han mermado de forma crítica. El motivo por el cual las existencias se han acumulado en los inventarios es el continuar surtiéndose sin lograr vender nada en los últimos 4 meses. Sin duda, hay productos más populares que otros. Sin embargo, es posible que el dejar de vender los productos menos populares sea solo una solución parcial ya que tampoco se han vendido los productos populares en los últimos años. Por este motivo, sí se recomienda recortar la venta de los productos impopulares e, incluso, hacer ventas de liquidación. Aunque también se recomienda fuerte mente crear una campaña publicitaria y monitorear constantemente el tráfico de usuarios en la plataforma de ventas para poder detectar cualquier anomalía que evite que ésta sea hallada por los buscadores. Independientemente de la situación de la empresa, podría ser una buena opción el tomar un

crédito blando para financiar estas campañas de publicidad e investigación y, así, perseguir una recuperación a corto-mediano plazo.