EMTECH
Emerging Technologies Institute

Santander

Introducción a Python
Caso práctico 1

Data science

Estudiante: Jimenez Schulz Leslie Mariana
Grupo 3

Código de acceso al proyecto:
https://github.com/jschulz22/proyecto1.git

Índice:

Introducción:

Gracias a que Python es un lenguaje de alto nivel con un uso cada vez mayor en distintos rubros, tanto tecnológicos como financieros, su dominio resulta fundamental para cualquier científico de datos o ingeniero de datos, pues debido a su versatilidad y hasta cierto punto, grado de intuitividad, facilita la eficiencia en la realización de códigos y ejecución de proyectos. Por lo tanto, de acuerdo con los objetivos planteados en este primer curso de *introducción a Python,* para la realización de este proyecto se ocuparon los principios básicos de programación en Python, sin incurrir en la utilización de apis, pandas o librerías que pudieran facilitar el mismo, ya que si bien tales herramientas constituyen una gran ayuda para cualquier programador, muchos procesos se pueden realizar con tan solo conocer los fundamentos básicos de la programación en Python, como se pudo comprobar durante la realización de este código, cuyo objetivo fue realizar de forma más eficiente el inventario dentro de un almacén para conocer las ventas del mismo y poder obtener un presupuesto y una propuesta que optimice y mejore las ventas del negocio.

Descripción del proyecto:

Para la realización del mismo se solicitó satisfacer 3 condiciones:

Método:

Para encontrar los 50 productos con mayores ventas se requirió del uso de ciclos for para poder romper con las estructuras empaquetadas en la lista de lifestore_sale y lifestore_products, esto con el propósito de averiguar cuántas veces se repetía el Id del producto buscado en la lista de ventas. Los resultados obtenidos se muestran en la imagen siguiente:

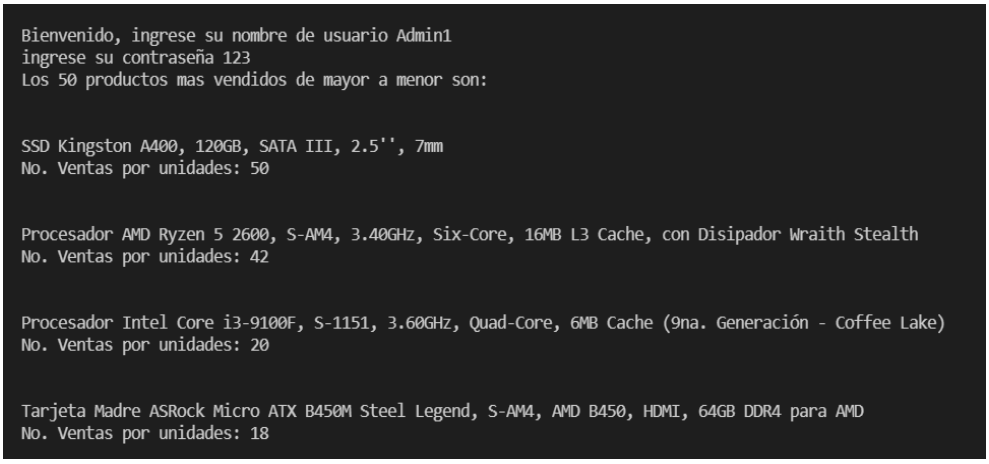


Fig1- 50 Fragmento de la lista que muestra los productos más vendidos de mayor a menor

En el caso de los productos más buscados se procedió de la misma manera que en el caso anterior, para los productos por categoría también se utilizaron ciclos for y el operador (=), generándose listas individuales para cada una de las diferentes categorías. Un procedimiento similar se siguió también para los productos menos buscados por categoría, obteniéndose los resultados que se pueden apreciar en las imágenes siguientes

Fig.2 Categoría: Procesadores con menores ventas

Fig.3 Categoría: Procesadores con menores búsquedas

Para obtener los datos que ilustrasen los productos con mejor reseña de acuerdo con el número de ventas y número de buscadas, se procedio a promediar ambos datos y a anexar en una lista el nombre del producto, junto con su ID y el valor del promedio obtenido, de está forma, aplicando la función sort() se pudo ordenar la lista, ya fuera de mayor a menor o de menor a mayor. De estas listas se seleccionaron los primeros 20 productos, con lo cual obtuvimos los resultados que se ilustran en la imagen siguiente:

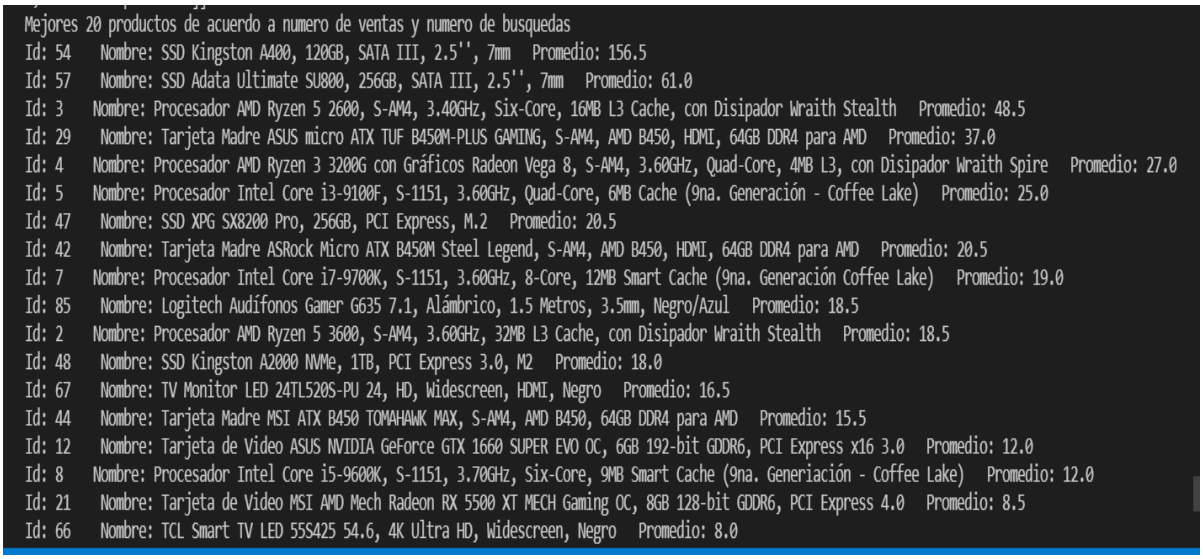


Fig.4 Mejores 20 productos de acuerdo con número de búsquedas y ventas

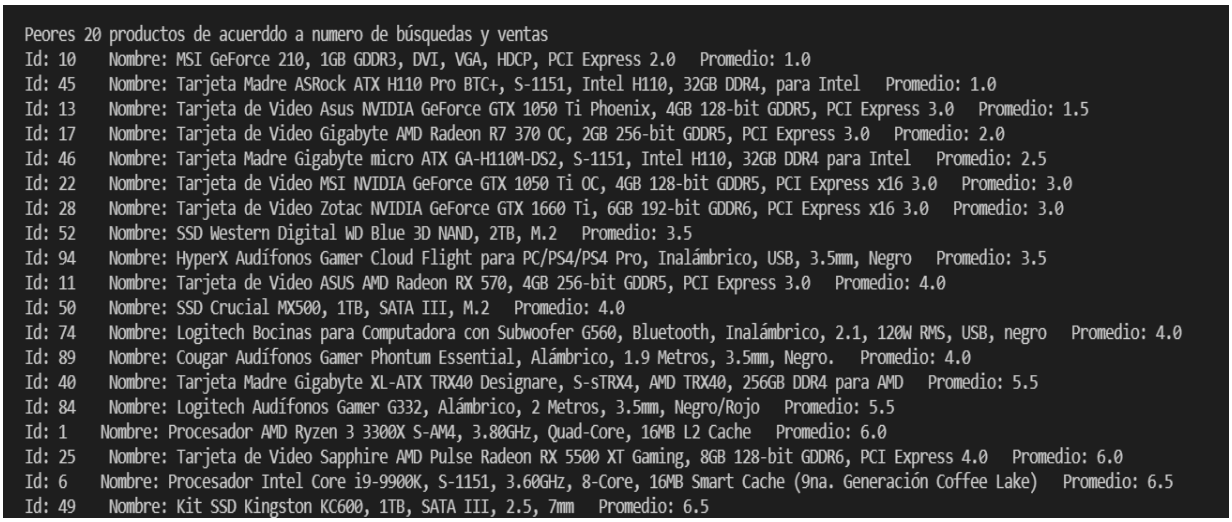


Fig.5 Peores 20 productos de acuerdo a número de búsquedas y ventas

Por otro lado también se obtuvieron los productos que no se vendieron, para lo cual fue necesario realizar una comparación entre los elementos 1 y 0 de las listas lifestore_sales

y lifestores_products respectivamente, con lo que se obtuvieron los resultados que se muestran a continuación:

```
los productos que no se vendieron son
ID: 1   Nombre: Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Quad-Core, 16MB L2 Cache
ID: 2   Nombre: Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 32MB L3 Cache, con Disipador Wraith Stealth
ID: 3   Nombre: Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth
ID: 4   Nombre: Procesador AMD Ryzen 3 3200G con Gráficos Radeon Vega 8, S-AM4, 3.60GHz, Quad-Core, 4MB L3, con Disipador Wraith Spire
ID: 5   Nombre: Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake)
ID: 6   Nombre: Procesador Intel Core i9-9900K, S-1151, 3.60GHz, 8-Core, 16MB Smart Cache (9na. Generación Coffee Lake)
ID: 7   Nombre: Procesador Intel Core i7-9700K, S-1151, 3.60GHz, 8-Core, 12MB Smart Cache (9na. Generación Coffee Lake)
ID: 8   Nombre: Procesador Intel Core i5-9600K, S-1151, 3.70GHz, Six-Core, 9MB Smart Cache (9na. Generiación - Coffee Lake)
ID: 9   Nombre: Procesador Intel Core i3-8100, S-1151, 3.60GHz, Quad-Core, 6MB Smart Cache (8va. Generación - Coffee Lake)
ID: 10   Nombre: MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PCI Express 2.0
ID: 11   Nombre: Tarjeta de Video ASUS AMD Radeon RX 570, 4GB 256-bit GDDR5, PCI Express 3.0
ID: 12   Nombre: Tarjeta de Video ASUS NVIDIA GeForce GTX 1660 SUPER EVO OC, 6GB 192-bit GDDR6, PCI Express x16 3.0
ID: 13   Nombre: Tarjeta de Video Asus NVIDIA GeForce GTX 1050 Ti Phoenix, 4GB 128-bit GDDR5, PCI Express 3.0
ID: 14   Nombre: Tarjeta de Video EVGA NVIDIA GeForce GT 710, 2GB 64-bit GDDR3, PCI Express 2.0
ID: 15   Nombre: Tarjeta de Video EVGA NVIDIA GeForce GTX 1660 Ti SC Ultra Gaming, 6GB 192-bit GDDR6, PCI 3.0
ID: 16   Nombre: Tarjeta de Video EVGA NVIDIA GeForce RTX 2060 SC ULTRA Gaming, 6GB 192-bit GDDR6, PCI Express 3.0
ID: 17   Nombre: Tarjeta de Video Gigabyte AMD Radeon R7 370 OC, 2GB 256-bit GDDR5, PCI Express 3.0
ID: 18   Nombre: Tarjeta de Video Gigabyte NVIDIA GeForce GT 1030, 2GB 64-bit GDDR5, PCI Express x16 3.0
ID: 19   Nombre: Tarjeta de Video Gigabyte NVIDIA GeForce GTX 1650 OC Low Profile, 4GB 128-bit GDDR5, PCI Express 3.0 x16
```

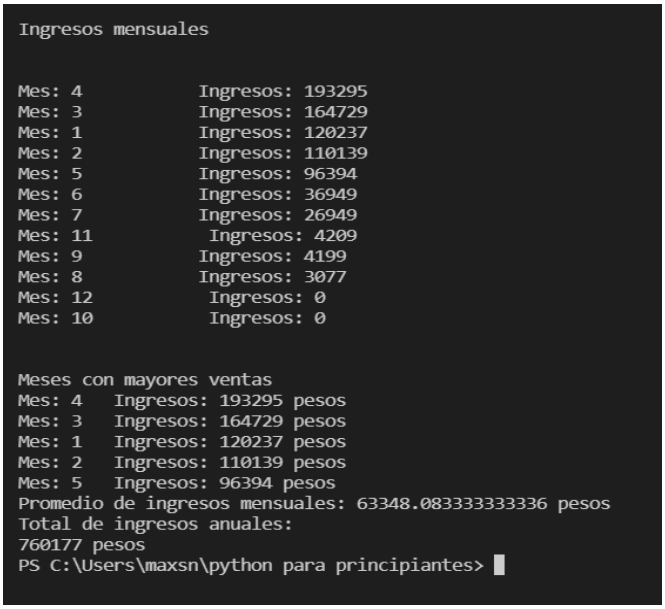
Fig.6 Productos que no tuvieron ventas

Como se puede apreciar en la imagen anterior fueron varios los productos que no reportaron ventas, alrededor de 46, por lo que sería recomendable no seguir surtiéndolos y enfocarse únicamente en los productos que si reportaron ventas.

Para el tercer punto que se pidió satisfacer en este proyecto, que fue obtener el promedio de ingresos mensuales, el total de ingresos anuales y el promedio de ventas por mes fue necesario iterar con ciclos *for* y utilizar contadores, de tal suerte que de la lista obtenida para los ingresos mensuales, a la cual se llamó *ingresos_mensuales,* mediante la función sort() se consideraron los 5 meses con mayores ventas.

Note particularmente que en la tabla obtenida para los ingresos mensuales hubo meses en los que no se reportaron ventas, por lo que si ese es el comportamiento que siguen las ventas durante tales ventas, convendría cerrar la tienda durante esos meses o intentar realizar una mejor estrategia publicitaria para aumentarlas.

Las tablas ya mencionadas se muestran a continuación, en donde al final se pueden apreciar los datos correspondientes al ingreso total de ventas anuales, en donde todas las ventas corresponden al año 2020 y el promedio de ingresos obtenidos por mes.



El código que permitió realizar todo anterior se encuentra en el siguiente vínculo: https://github.com/jschulz22/proyecto1.git
Para acceder a el debe escribir como nombre de usuario “Admin1” o “Admin2” y poner la contraseña “123” o “abc”.

Definición de código:

```
from lifestore_file import  lifestore_searches, lifestore_sales, lifestore_products
import sys

##Solicitud al usuario de contraseña y nombre
usuario_01=input("Bienvenido, ingrese su nombre de usuario ")
contraseña_01=input("ingrese su contraseña ")

usuario= ["Admin1", "Admin2"]
contraseña=["123", "abc"]

if usuario_01  not in usuario and contraseña_01  not in  contraseña:
    sys.exit("Usted no tiene permisos para ejecutar este programa, gracias por su visita")
##fin de nombre y contraseña del usuario

##Productos vendidos del menos vendido al más vendido
```

Para observar cuantas veces es nombrada la variable en la lista se utiliza el contador
Los ciclos for son necesarios para poder entrar dentro de la estructura de la lista

```
ventas=[]
contador=0
for producto in lifestore_products:
    for venta in lifestore_sales:
        if producto[0]==venta[1]:
            contador= contador+1
    if contador != 0:
        ventas.append ([contador, producto[0]])
        contador=0
ventas.sort(reverse=True)
```

0 productos más vendidos del listado
Se busca el nombre del producto en la lista life_store mediante la variable nombre_del_producto. Las ventas por unidades se obtienen del valor[0] de la lista ventas

```
contador=0
print("Los 50 productos mas vendidos de mayor a menor son:")
print("\n")
for id in ventas:
    if contador!=50:
        nombre_del_producto = lifestore_products[id[1]-1][1]
        print(f"{nombre_del_producto} ")
        print(f"No. Ventas por unidades: {id[0]} ")
    contador += 1
    print("\n")
```

En la siguiente parte del código seleccionamos los productos menos vendidos por categoría de entre los 50 productos menos vendidos. Se empleó la función sort() para tener la lista ordenada de menor a mayor

```
ventas.sort()
productos_menos_vendidos = ventas[:49]
print(productos_menos_vendidos)

print("Productos por categoria con menores ventas")
print("\n")
```

Las ventas que se obtuvieron de los productos de acuerdo con sus categorías se obtuvieron comparando el nombre de la categoría deseada con el valor del nombre del producto en la lista lifestore_products.

```
print(" Procesadores con menores ventas")
```

```
for venta in productos_menos_vendidos:
    procesador= lifestore_products[venta[1]-1][3]
    if procesador == "procesadores":
        print(f"Nombre: {lifestore_products[venta[1]-1][1]}")
        print(f"Unidades vendidas: {venta[0]} ")
print("\n")
print("Tarjetas de video")
for venta in productos_menos_vendidos:
    tarjeta_video= lifestore_products[venta[1]-1][3]
    if tarjeta_video == "tarjetas de video":
        print(f"La  {lifestore_products[venta[1]-1][1]} vendio {venta[0]} unidad")
print("\n")

print("Tarjetas madre")
for venta in productos_menos_vendidos:
    tarjeta_madre= lifestore_products[venta[1]-1][3]
    if tarjeta_madre == "tarjetas madre":
        print(f"La  {lifestore_products[venta[1]-1][1]} vendio {venta[0]} unidad")
print("\n")

print("discos duros")
for venta in productos_menos_vendidos:
    discos= lifestore_products[venta[1]-1][3]
    if discos == "discos duros":
        print(f"La  {lifestore_products[venta[1]-1][1]} vendio {venta[0]} unidad")
print("\n")

print("Audifonos")
for venta in productos_menos_vendidos:
    audifono= lifestore_products[venta[1]-1][3]
    if audifono == "audifonos":
        print(f" {lifestore_products[venta[1]-1][1]} vendio {venta[0]} unidad")
print("\n")

print("Memorias USB")
for venta in productos_menos_vendidos:
    memorias= lifestore_products[venta[1]-1][3]
    if memorias == "memorias usb":
        print(f"La  {lifestore_products[venta[1]-1][1]} vendio {venta[0]} unidad")
print("\n")

print("Pantallas")
for venta in productos_menos_vendidos:
```

```
    pantalla= lifestore_products[venta[1]-1][3]
    if pantalla == "pantallas":
        print(f"La  {lifestore_products[venta[1]-1][1]} vendio {venta[0]} unidad")
print("\n")

print("bocinas")
for venta in productos_menos_vendidos:
    bocina= lifestore_products[venta[1]-1][3]
    if bocina == "bocinas":
        print(f"Las  {lifestore_products[venta[1]-1][1]} vendio {venta[0]} unidad")
print("\n")
##fin del código para los productos menos vendidos por categoría
```

100 productos con menores busquedas

En esta lista se almacenaron los indices de los productos que habían sido buscados por lo menos una vez

```
lista_busquedas=[]
for busqueda in lifestore_searches:
    numero_busquedas = busqueda[1]
    lista_busquedas.append( numero_busquedas )

##Se creo una lista llamada conteo en la que se almacenaron los valores de los productos en función de su número de búsquedas
conteo=[]
for producto_buscado in lista_busquedas:
    if [lista_busquedas.count(producto_buscado),producto_buscado] not in conteo:
        conteo.append([lista_busquedas.count(producto_buscado),producto_buscado])
conteo.sort(reverse=True)
##Fin del código para productos con menores busqueda
```

Productos mas buscados

Para obtener cuales fueron los productos más buscados se utilizo un contador que iniciamos en 0. Para localizar el nombre del producto fue necesario navegar la lista lifetore_products.

```
print("Productos MÁs buscados")
contador=0
print("Los 100 productos con más busquedas son:")
print("\n")
for numero in conteo:
    if contador!=100:
        nombre_del_producto = lifestore_products[numero[1]-1][1]
```

```
    print(f"Nombe: {nombre_del_producto}        Número de búsquedas: {numero[0]} ")
    contador += 1
```

Búsquedas de productos:

Para obtener la lista que nos diera los productos que habían sido buscados se localizó ese producto en la lista productos_menos_buscados, si el producto no estaba en tal lista, el valor del número de veces que tal producto aparecía junto con el valor del Id de este producto eran almacenados en esa lista.

```
productos_menos_buscados=[]
for producto_buscado in lista_busquedas:
  if [lista_busquedas.count(producto_buscado),producto_buscado] not in productos_menos_buscados:
    productos_menos_buscados.append([lista_busquedas.count(producto_buscado),producto_buscado])
productos_menos_buscados.sort()

print("/"*100)
print("Productos por categoria con menores busquedas")
```

Los productos con menores búsquedas se buscaron en la lista de productos_menos_buscados. El nombre del producto se buscó en la lista lifestore_producs, en cada caso fue necesario hacer coincidir el nombre con la categoría del producto que estábamos buscando

```
print(" Procesadores con menores busquedas")
for producto_no_buscado in productos_menos_buscados:
  procesador= lifestore_products[producto_no_buscado[1]-1][3]
  if procesador == "procesadores":
    print(f"El  {lifestore_products[producto_no_buscado[1]-1][1]} fue buscado {producto_no_buscado[0]} vez")
print("\n")

print(" discos duros")
for producto_no_buscado in productos_menos_buscados:
  disco_duro_01= lifestore_products[producto_no_buscado[1]-1][3]
  if disco_duro_01== "dicos duros":
    print(f"El  {lifestore_products[producto_no_buscado[1]-1][1]} fue buscado {producto_no_buscado[0]} vez")
print("\n")

print("Audifonos")
for producto_no_buscado in productos_menos_buscados:
  audifono= lifestore_products[producto_no_buscado[1]-1][3]
  if audifono == "audifonos":
    print(f" {lifestore_products[producto_no_buscado[1]-1][1]}      No. Busquedas: {producto_no_buscado[0]} ")
```

```
print("\n")

print("Tarjetas de video")
for producto_no_buscado in productos_menos_buscados:
    tarjeta_video= lifestore_products[producto_no_buscado[1]-1][3]
    if tarjeta_video == "tarjetas de video":
        print(f"La  {lifestore_products[producto_no_buscado[1]-1][1]} fue buscado {producto_no_buscado[0]} vez")
print("\n")

print("Tarjetas madre")
for producto_no_buscado in productos_menos_buscados:
    tarjeta_madre= lifestore_products[producto_no_buscado[1]-1][3]
    if tarjeta_madre == "tarjetas madre":
        print(f"La  {lifestore_products[producto_no_buscado[1]-1][1]} fue buscado {producto_no_buscado[0]} veces")
print("\n")

print("Memorias USB")
for producto_no_buscado in productos_menos_buscados:
    memorias= lifestore_products[producto_no_buscado[1]-1][3]
    if memorias == "memorias usb":
        print(f"La  {lifestore_products[producto_no_buscado[1]-1][1]} fue buscado {producto_no_buscado[0]} veces")
print("\n")

print("Pantallas")
for producto_no_buscado in productos_menos_buscados:
    pantalla= lifestore_products[producto_no_buscado[1]-1][3]
    if pantalla == "pantallas":
        print(f"La  {lifestore_products[producto_no_buscado[1]-1][1]} fue buscado {producto_no_buscado[0]} veces")
print("\n")

print("bocinas")
for producto_no_buscado in productos_menos_buscados:
    bocina= lifestore_products[producto_no_buscado[1]-1][3]
    if bocina == "bocinas":
        print(f"Las  {lifestore_products[producto_no_buscado[1]-1][1]} fue buscado {producto_no_buscado[0]} veces")
print("\n")

##fin del codigo para productos por catregoria con menores busquedas

##Productos por reseña y categoría

print("productos por reseña y por servicio")
```

Para obtner los productos que no se vendieron se realizó una comparación entre los valores almacenados para los Id en las listas lifestore_products y lifestore_sales. Los productos que no podían localizarse en ambas listas se almacenaron en una tercera lista que concentró los productos no vendidos

```
print("los productos que no se vendieron son")
for producto in lifestore_products:
    for venta in lifestore_sales:
        if producto[0] != venta[1]:
            producto_no_vendido = True
    if producto_no_vendido == True:
        print(f"ID: {producto[0]}   Nombre: {producto[1]} ")
print("\n")

conteo.sort(reverse=True)
ventas.sort(reverse=True)
```

Código para productos más buscados:
En este caso se tuvo que realizar una identación de tres niveles y obtener una lista en la que se pudieran almacenar los valores que se necesitaban, en este caso, el Id del producto, su nombre y su número de búsquedas

```
productos_mas_buscados=[]
for producto in lifestore_products:
    for producto_buscado in conteo:
        if producto_buscado[1]==producto[0]:
            productos_mas_buscados.append([producto[0], producto[1], producto_buscado[0]])
```

El promedio que se obtuvo de la forma que a continuación se ilustra, fue un valor necesario para posteriormente obtener los mejores y peores productos en función de su número de búsquedas y ventas

```
lista=[]
for busquedas in productos_mas_buscados:
    b=int(busquedas[2])
    for id in ventas:
        a=int(id[0])
        if id[1]==busquedas[0]:
         promedio=(a+b)/2
         lista.append([promedio, busquedas[0], busquedas[1]])

print("Los 56 mejores productos de acuerdo con el número de venta y numero de búsquedas son:")
for elemento in productos_mas_buscados:
```

```
    for id in ventas:
        if elemento[0]==id[1]:
            print(f"ID: {id[1]}     Nombre: {elemento[1]}")
            print(f"No.busquedadas: {elemento[2]}   ventas {id[0]}")
            print("\n")
print("\n")
```

Para calcular los mejores 20 productos por "reseña" considerando el análisis realizado para obtener los productos más vendidos y los productos con más búsquedas, sin eliminar los productos devueltos, tenemos el código siguiente:

20 mejores productos

Se empleó la función sort() inverse para poder obtener los valores acomodados del más grande al más pequeño. Se consideró la lista de productos_mas_buscados para poder obtener una lista que nos diera los productos mejor ranqueados

```
lista.sort(reverse=True)
productos_mejores=lista[0:19]

print("Mejores 20 productos de acuerdo a numero de ventas y numero de busquedas")
for elemento in productos_mejores:
    print(f"Id: {elemento[1]}    Nombre: {elemento[2]}   Promedio: {elemento[0]}")
print("\n")
```

20 Peores productos

Se empleó la función sort() para poder obtener los valores ordenados de menor a mayor. Se realizó una nueva lista que contenía a los productos ranqueados con el puntaje más bajo de acuerdo con el promedio.

```
lista.sort()
productos_peores=lista[0:19]
print("Peores 20 productos de acuerddo a numero de búsquedas y ventas")
for elemento in productos_peores:
    print(f"Id: {elemento[1]}    Nombre: {elemento[2]}   Promedio: {elemento[0]}")
print("\n")

##fin del código para mejores y peores productos de acuerdo con reseña
```

Cálculos de ingresos

En este caso se empleó la función round para obtener el valor redondeado. Fue necesario contar el número de elementos en la lista para poder obtener el promedio de las ventas

```
print("Promedio de Ventas Mensuales")
ventas_anuales=len(lifestore_sales)
ventas_promedio_mensuales=(ventas_anuales)/12
Numero_redondeado = round(ventas_promedio_mensuales, 2)
print(Numero_redondeado)
print("\n")

##Ingresos mensuales
print("Ingresos mensuales")
ingresos_mensuales = []
```

Ciclo for que permite validar los valores encontrados para una lista de datos

```
for i in range(1,13):
    total = 0
    for ventas in lifestore_sales:
        producto_vendido = ventas[1]
        ganancia = lifestore_products[ventas[1]-1][2]
        fecha_de_venta = ventas[3]
        mes_de_venta = fecha_de_venta[3:5]
        if i == int(mes_de_venta):
            total += ganancia
    ingresos_mensuales.append([total,i])
    ingresos_mensuales.sort(reverse=True)
print("\n")

for ingreso in ingresos_mensuales:
    print(f"Mes: {ingreso[1]}          Ingresos: {ingreso[0]}")
print("\n")

##meses con mayores ventas

meses_con_mas_ventas=ingresos_mensuales[0:5]

print("Meses con mayores ventas")
for ingresos in meses_con_mas_ventas:
    print(f"Mes: {ingresos[1]}   Ingresos: {ingresos[0]} pesos")

valores=[]
```

```
inicio=0
for ingreso in ingresos_mensuales:
    m=int(ingreso[0])
    inicio=inicio+m
    valores.append(inicio)
no_meses=len(ingresos_mensuales)
promedio_l=inicio/(no_meses)
print(f"Promedio de ingresos mensuales: {promedio_l} pesos")
print("Total de ingresos anuales:")
print(f"{valores[11]} pesos")
```

Solución al problema:

De acuerdo con lo que se pudo observar al ejecutar el código hay un total de 46 productos que no reportaron ventas, por lo que la sugerencia sería retirarlos dentro de los productos que la tienda tiene a su disposición, por otro lado hay una drástica disminución de las ventas en los meses de octubre y diciembre, pues en tales meses el ingreso de ventas decae a 0, por lo que el almacén debería permanecer cerrado durante ese periodo o bien, establecer una mejor estrategia de publicidad para aumentar las ventas.

Por otra parte, los productos que se categorizaron en la lista de peores productos por ser los productos con menor número de búsquedas y ventas, deberían disminuir su existencia en almacén, pues realmente no aportan muchos ingresos a la tienda y solo se acumulan en el inventario, restando espacio para almacenar aquellos productos que se venden con mucha más frecuencia.

Importante también sería reducir de acuerdo con el número de ventas reportado, los productos que menos se venden por categoría, pues al igual que en el caso anterior, los productos que se venden poco solo ocupan espacio en el inventario que podría ser aprovechado para almacenar aquellos que si reportan ventas, lo cual ayudaría a hacer más eficiente el proceso de compra venta y probablemente a mejorar los ingresos de la tienda.

Conclusión:

Como se pudo observar durante la realización de este proyecto, el lenguaje de programación *Python* resulta útil para tratar grandes volúmenes de datos y poder manejar esa información, de acuerdo con las necesidades que se tengan, con mucha más facilidad, esto gracias a los distintos arreglos que se pueden hace r para tratar con tales datos, como las tuplas y las listas, en los cuales se pueden almacenar valores que si bien son de carácter distinto, ya que se puede tratar de valores numéricos o de cadenas de texto,

se puede trabajar con ellos con facilidad cuando se navega a través de tales listas y se itera para separarlos según las necesidades que tengamos. Es por esta razón que al mismo tiempo *Python* nos facilita la realización de procesos matemáticos que de otra forma resultarían largos y tedioso y representa, por tanto, una muy útil herramienta en diversos ramos de la industria.