MARTES 14 DE SEPTIEMBRE DEL 2021

EMTECH
Emerging Technologies Institute

REPORTE 01

ANÁLISIS DE DATOS PARA LIFESTORE

ALEXA MERCADO SÁNCHEZ

TUTOR: JAIME SAÚL ALONSO SÁNCHEZ
EmTech

Índice

El programa en GitHub se puede encontrar aquí: https://github.com/coolaxtro/EmTech.git

Introducción

La finalidad del análisis de datos tiene que ver con facilitar el manejo y representación de grandes cantidades de información, encontrando patrones y promoviendo un análisis que permita deducir comportamientos y entender el por qué de fenómenos sociales, económicos, biológicos, etc.

En este caso, el proyecto plantea una problemática de la empresa LifeStore. Esta es una tienda en línea que, a partir de listas de datos generados, necesita que le facilitemos conocer algunas características de su rendimiento en el mercado y su relación con sus clientes. Espera que los auxiliemos con los siguientes puntos:

- Productos más vendidos y productos rezagados a partir del análisis de las categorías con menores ventas y categorías con menores búsquedas.
    - Generar un listado de los 50 productos con mayores ventas y uno con los 100 productos con mayores búsquedas.
    - Por categoría, generar un listado con los 50 productos con menores ventas y uno con los 100 productos con menores búsquedas.
- Productos por reseña en el servicio a partir del análisis de categorías con mayores ventas y categorías con mayores búsquedas.
    - Mostrar dos listados de 20 productos cada una, un listado para productos con las mejores reseñas y otro para las peores, considerando los productos con devolución.
- Sugerir una estrategia de productos a retirar del mercado, así como sugerencia de cómo reducir la acumulación de inventario considerando los datos de ingresos y ventas mensuales.
    - Total, de ingresos y ventas promedio mensuales, total anual y meses con más ventas al año

Definición del código

Importación de información

A partir del archivo lifestore_file.py nos es posible importar las listas que contienen la información a analizar. Esto se realiza para no tener tantas líneas de código en un solo documento y que sea más fácil para el programador visualizar su progreso.

```
#Aquí se importan desde el archivo lifestore_file.py las 3 listas
que usaremos
from lifestore_file import lifestore_products as LP
from lifestore_file import lifestore_sales as LSa
from lifestore_file import lifestore_searches as LSe
```

Aquí, entonces, se importan las listas de productos, ventas y búsquedas como LP, LSa y LSe respectivamente.

Es importante recordar que estas listas tienen la forma

- LP = [id_product, name, price, category, stock]
- LSa = [id_sale, id_product, score (from 1 to 5), date, refund (1 for true or 0 to false)]
- LSe = [id_search, id product]

Definición de variables

Se definen algunas listas que nos permitirán almacenar información.

```
#Definición de variables
ventas=[]
busca=[]
bajos=[]
altos=[]
ratingdual=[]
reembolso=[]
categoriaelegida=[]
resultadocat=[]
```

```
#Definición de variables de ventas por mes
ventas1=[]
ventas2=[]
ventas3=[]
ventas4=[]
ventas5=[]
ventas6=[]
ventas7=[]
ventas8=[]
ventas9=[]
ventas10=[]
ventas11=[]
ventas12=[]
#Variable que hace append de todos los meses
Ventapormeses=[]
```

Donde:

- **ventas** guardará información de LSa como [id, categoría] en forma nested list.
- **busca** almacenará datos de LSe como [id] en forma nested list.
- **bajos y altos** almacenarán información de las reseñas de LSa bajas y altas respectivamente como [id, reembolso] en forma de nested list.
- **ratingdual** tendrá información de productos que hayan sido calificados tanto bien como mal en forma [id, reembolso] en forma de nested list.
- **reembolso** guardará datos de elementos que hayan sido reembolsados de forma [id, reembolso] en forma de nested list.
- **categoriaelegida** almacenará todos los elementos vendidos de la categoría elegida de la forma [#ventas, [id, categoría]]
- **resultadocat** desplegará el número de ítems que el usuario elija de su categoría seleccionada de forma [#ventas, [id, categoría]]
- **ventas1-12** guardarán información desde LSa de las ventas respectivas por mes.
- **Ventapormeses** hará un append de todas las ventas por meses de la forma [mes,[venta, promedio de ventas]]

Generando nested lists en las variables definidas

Ventas de productos: ventas

Para identificar cuáles productos habían sido los más o menos vendidos, se decidió guardarlo en forma [id, categoría] para poder generar fácilmente la relación de las ventas con su categoría mediante el siguiente código.

```
for venta in LSa: #itera en cada elemento de la lista
lifestore_sales
    for J in LP: #revisa cada elemento de la lista
    lifestore_products para revisar la categoría
        if venta[1]==J[0]: #revisa que los índices que contienen el
        id del producto sean iguales
            ventas.append([venta[1],J[3]]) #hace append del id del
            producto y su categoría
```

Búsquedas: busca

Como no existían requerimientos especiales para las búsquedas, como relacionarlas con algún otro parámetro, el código quedó como sigue:

```
for search in LSe: #itera en cada búsqueda de lifestore_searches
    busca.append(search[1]) #genera una lista con todos los IDs de
    búsquedas
```

Calificaciones y reembolso: bajo, alto, ratingdual y reembolso

Para ello se estableció que un rating alto correspondía de 4-5 estrellas y bajo de 1-3. Asimismo, se encontraron los productos que tuvieron un rating tanto alto como bajo y los productos con reembolso.

```
for rating in LSa: #por cada elemento en lifestore_sales
    if rating[2]<=3: #en caso de que el rating encontrado en el
    elemento de lifestore_sales sea menor o igual a tres se
    considera mala review
        bajos.append([rating[1],rating[4]]) #en la lista bajos se
        añade el [id,reembolso (1/0)]
    else:  #en caso de que el rating sea mayor a 3 estrellas
        altos.append([rating[1],rating[4]]) #en la lista altos se
        añade el [id,reembolso (1/0)]
for i in bajos:
    if i in altos: #en caso de que un elemento esté tanto en la
    lista de altos y bajos
        ratingdual.append(i) #se añade a la lista de rating dual
    if i[1]==1: #en caso de que se haya reembolsado
        reembolso.append(i) #se añade a la lista de los elementos con
        reembolso
```

Ventas y promedios de ventas: ventas(1-12), Ventapormeses

Se decidió generar distintas listas que contuvieran las ventas por mes, así como una que juntara la de todos los meses. Esto se realizó buscando que en el índice de la fecha de compra, apareciera el mes y le hiciera append a su lista designada.

```
#Filtra por mes los datos de ventas y los guarda en su lista
respectiva
for i in LSa:
  if "/01/" in i[3]:
    ventas1.append(i)
  elif "/02/" in i[3]:
    ventas2.append(i)
  elif "/03/" in i[3]:
    ventas3.append(i)
  elif "/04/" in i[3]:
    ventas4.append(i)
  elif "/05/" in i[3]:
    ventas5.append(i)
  elif "/06/" in i[3]:
    ventas6.append(i)
  elif "/07/" in i[3]:
    ventas7.append(i)
  elif "/08/" in i[3]:
    ventas8.append(i)
  elif "/09/" in i[3]:
    ventas9.append(i)
  elif "/10/" in i[3]:
    ventas10.append(i)
  elif "/11/" in i[3]:
    ventas11.append(i)
  elif "/12/" in i[3]:
    ventas12.append(i)
  else: print("¡No hay más de 12 meses!") #En caso de que llegue
  un dato extranio
```

Funciones utilizadas

Contador de recurrencias: counter

En todos los casos necesitamos obtener de alguna forma el número de veces que se compra un producto, se califica mal un producto, se busca un producto, etc. Por ello se generó una función encargada de contar en la lista introducida el número de recurrencias donde list1 es la lista para contabilizar y x define si buscamos anidar uno o dos parámetros.

```
def counter(list1,x): #Función para contabilizar la recurrencia
de un dato en una lista
    lista=[]
    lista2=[]
    if x==True: #Es true cuando se necesitan anidar dos parámetros
    por iteración a la lista, como el caso de la valoración que
    sería [id, reembolso] o reembolso que sería [id, 1/0]
        valores=[[lista[0],lista[1]] for lista in list1] #le asigna
        los valores de los índices 0 y 1 de la lista de entrada a la
        variable valores
    else: #En caso de que sólo se necesite 1 valor, como el caso de
    búsquedas con el id del producto buscado
        valores=[lista for lista in list1] #le asigna los valores de
        la lista de entrada a valores
    for i in valores:
        if i not in lista:
            lista.append(i) #hace append de los valores nuevos que
            encuentre en lista
    for i in lista: #de todos los valores nuevos y únicos que
    encontró
        A=valores.count(i) #cuenta cuántas veces se encuentran en la
        lista de valores
        lista2.append([A,i]) #genera una nueva lista que hace append
        de [veces que se encontró el valor, valor]
    return(lista2)

#en esta sección se hace la cuenta de los items de las listas
generadas de busquedas, ventas y ratings tanto altos como bajos
ventascontadas=counter(ventas,True)
busquedascontadas=counter(busca,False)
ratingbajocontado=counter(bajos,True)
ratingaltocontado=counter(altos,True)
```

En caso de que x==True, la salida será [#recurrencias, [id, dato]]. En caso de que x==False la salida será [#recurrencias, id].

Sorteo de datos: sorteo

Por otro lado, tenemos que acomodar los datos ya cuantificados por orden ascendente o descendente dependiendo de lo que se busque. Aparte, se estableció que el usuario puede elegir cuántos productos quiere que se muestren de este sorteo. Aquí, b es el número de productos que se desplegarán, lista es la lista de entrada y x es el parámetro para definir si queremos un sorteo ascendente o descendente.

```
def sorteo(b,lista,x):
    sortedlist=[] #se genera lista vacía
    if x=='high':
        lista.sort(reverse=True, key=lambda x: x[0]) #permite filtrar
        los datos de mayor a menor, tomando como dato crítico el que
        está en el índice 0
    elif x=='low':
        lista.sort(key=lambda x: x[0]) #permite filtrar los datos de
        menor a mayor, tomando como dato crítico el que está en el
        índice 0
    else:
        print("Error, choose high o low")
        exit(1) #sale del programa en caso de que haya error

    for i in range (b): #Lo itera el número de veces que elija el
    usuario
        if i>=len(lista):
            sortedlist.append('Out of range') #en caso de que sea mayor
            al índice de la lista, hace append de 'out of range'
        else:
            sortedlist.append(lista[i]) #en caso de que el índice esté
            dentro, hace append del valor en el índice i de nuestra
            lista de entrada
    return(sortedlist) #regresa la lista sorteada
```

Cálculo de ventas: ventas

Otro requerimiento es que ayudemos a LifeStore a encontrar detalles de sus ventas totales, así como por mes. Con esta función es posible encontrar el dinero ganado, así como el promedio de ventas sin contabilizar los elementos que hayan tenido devolución.

```
def ventastot(lista):#se introduce la lista de la que se desea
encontrar su suma y promedio
    ventaZ=0
    count=0
    promedio=0
    #se generan 3 variables, ventaZ que guarda la suma de las
    ventas, count que permite contar cuántas veces se realiza el
    loop para despues poder sacar el promedio guardado en la
    variable promedio
    for i in lista: #itera en cada elemento de la lista introducida
        if i[4]==0: #sólo productos sin devolución a considerar
            for J in LP: #revisa cada elemento de la lista
            lifestore_products para revisar los precios
                if i[1]==J[0]: #revisa que los índices que contienen el
                id del producto sean iguales
                    ventaZ+=float(J[2]) #suma el precio del producto
                    encontrado a ventas
                    count+=1 #cuenta las veces que se itera
            if count>0: #en caso de sí tener ventas
                promedio=ventaZ/count #saca el promedio
    return([ventaZ,promedio]) #regresa en forma de lista ventas y
    promedio
```

```
Ventatotal=ventastot(LSa) #Encuentra las ventas totales y
promedio total de ventas

#Hace append de cada mes en Ventaspormeses[]
Ventapormeses.append([1,ventastot(ventas1)])
Ventapormeses.append([2,ventastot(ventas2)])
Ventapormeses.append([3,ventastot(ventas3)])
Ventapormeses.append([4,ventastot(ventas4)])
Ventapormeses.append([5,ventastot(ventas5)])
Ventapormeses.append([6,ventastot(ventas6)])
Ventapormeses.append([7,ventastot(ventas7)])
Ventapormeses.append([8,ventastot(ventas8)])
Ventapormeses.append([9,ventastot(ventas9)])
Ventapormeses.append([10,ventastot(ventas10)])
Ventapormeses.append([11,ventastot(ventas11)])
Ventapormeses.append([12,ventastot(ventas12)])
```

La salida de Ventapormeses es entonces [mes, [ventas, promedio de ventas]]

Interfaz de usuario

Pantalla de Inicio y Login

En la pantalla de inicio se imprime el nombre de las instituciones involucradas, da la bienvenida y pide el usuario y contraseña. Asimismo, se define una variable de intentos para limitar el acceso si no se conoce la contraseña. El programa permitirá que el usuario introduzca la contraseña incorrectamente un máximo de 3 veces, en caso de que exceda este número de intentos, se cierra el programa.

- La contraseña es **Pass123**

```
print("""
 ______       _______       _
|  ____|     |__   __|     | |
| |__   _ __ ___ | | ___  ___| |__
|  __| | '_ ` _ \| |/ _ \/ __| '_ \
| |____| | | | | | |  __/ (__| | | |
|______|_| |_| |_|_|\___|\___|_| |_|

      |  |  |  |  |  |  |and

 LINESTORE

""") #Para que se vea pro

print("Bienvenido a este software para consultar datos de tu
tienda\n\n")
hola=input("Usuario: ")
cont=input("Contraseña: ")
intento=1
```

```
 ______       _______       _
|  ____|     |__   __|     | |
| |__   _ __ ___ | | ___  ___| |__
|  __| | '_ ` _ \| |/ _ \/ __| '_ \
| |____| | | | | | |  __/ (__| | | |
|______|_| |_| |_|_|\___|\___|_| |_|

                and

    LINESTORE


Bienvenido a este software para consultar datos de tu tienda


Usuario: Alex
Contraseña: Pass123
```

```
#Esta sección espera que el usuario digite correctamente la
contraseña en al menos 3 intentos, si no, finaliza el programa
while(cont!='Pass123'):
  if intento<=2:
    print("Contraseña incorrecta, intenta de nuevo")
    cont=input("Contraseña: ")
    intento+=1
  else:
    print("Contacte a su administrador")
    exit(1)
```

```
Usuario: Alex
Contraseña: 1
Contraseña incorrecta, intenta de nuevo
Contraseña: 2
Contraseña incorrecta, intenta de nuevo
Contraseña: 3
Contacte a su administrador
repl process died unexpectedly: exit status 1
```

Bucle infinito

Este bucle nos permitirá preguntarle al usuario qué opción desea consultar desde el menú generado.

```
while(True): #Bucle infinito
  print(100*"\n") #Para limpiar pantalla
  print("¿Qué deseas hacer?")
  decision=input("""
   _____________________________
  |1.- Mayores ventas           |
  |2.- Mayores búsquedas        |
  |3.- Menores ventas           |
  |4.- Menores búsquedas        |
  |_____________________________|
  |5.- Mejores reseñas          |
  |6.- Peores reseñas           |
  |_____________________________|
  |7.- Total de ingresos        |
  |8.- Ventas promedio mensuales|
  |9.- Meses con más ventas     |
  |_____________________________|
  """)

  decision=int(decision) #Haciendo cast de la variable a int
  print(100*"\n") # Para limpiar pantalla

  index=0 #Nos permitirá enumerar las listas de salida como
  counter

  #Generando los headers de las distintas tablas posibles
  tablav="|  Lugar  |  ID  |  Ventas   |"
  tablab="|  Lugar  |  ID  | Búsquedas |"
  tablar="|  Lugar  |  ID  |  #Ratings |"
  tablabr="|  Lugar  |  ID  |  #Ratings |  Reembolso  |"
  tablam="|  Mes  |  Ventas  |  Promedio |"
```

```
¿Qué deseas hacer?

   _____________________________
  |1.- Mayores ventas           |
  |2.- Mayores búsquedas        |
  |3.- Menores ventas           |
  |4.- Menores búsquedas        |
  |_____________________________|
  |5.- Mejores reseñas          |
  |6.- Peores reseñas           |
  |_____________________________|
  |7.- Total de ingresos        |
  |8.- Ventas promedio mensuales|
  |9.- Meses con más ventas     |
  |_____________________________|
```

Y posteriormente, dependiendo de la opción que elija, será el proceso por realizar.

Opción 1

En caso de elegir “Mayores ventas” el código y output es el siguiente:

```
if decision==1: #Mayores ventas
  dec=input("¿De cuántos items deseas mayores ventas? \n")
  highventas=sorteo(int(dec),ventascontadas,'high') #Sort
  descendente de "dec" ventas cuantificadas
  print(tablav) #Imprimiendo los títulos de la tabla

  for i in highventas: #Imprime elemento por elemento: index,
  ID y número de ventas
    index+=1
    print( "| ",index, "      ", i[1][0],"       ",i[0])

  #La siguiente parte del código es igual para todas las
  opciones, intenté hacer una función pero no me fue posible
  posicionar en la función un continue, por lo que es
  repetitivo a través del código
  inout=input("¿Deseas hacer otra consulta? (y/n) \n")
  if inout=='y':
    continue
  else:
    print("Hasta la vista, viejo")
    exit(1)
```

```
¿De cuántos items deseas mayores ventas?
12
|  Lugar  |  ID  |  Ventas   |
|  1        54         50
|  2        3         42
|  3        5         20
|  4        42         18
|  5        57         15
|  6        29         14
|  7        2         13
|  8        4         13
|  9        47         11
|  10        12         9
|  11        48         9
|  12        7         7
¿Deseas hacer otra consulta? (y/n)
```

Opción 2

En caso de elegir “Mayores búsquedas” el código y output es el siguiente:

```
elif decision==2: #Búsquedas máximas
  dec=input("¿De cuántos items deseas mayores búsquedas? \n")
  highbusq=sorteo(int(dec),busquedascontadas,'high') #Sort
  descendente de "dec" búsquedas cuantificadas

  print(tablab)
  for i in highbusq:
    index+=1
    print( "| ",index, "      ", i[1],"       ",i[0])

  inout=input("¿Deseas hacer otra consulta? (y/n) \n") #Espera
  que el usuario le diga si quiere hacer otra cosa
  if inout=='y':
    continue
  else:
    print("Hasta la vista, baby")
    exit(1)
```

```
¿De cuántos items deseas mayores búsquedas?
8
|  Lugar  |  ID  | Búsquedas |
|  1        54         263
|  2        57         107
|  3        29         60
|  4        3         55
|  5        4         41
|  6        85         35
|  7        67         32
|  8        7         31
¿Deseas hacer otra consulta? (y/n)
```

Opción 3

En caso de elegir “Menores ventas” el código y output es el siguiente:

```
elif decision==3: #Ventas menores
  dcateg=[] #Se genera una variable que almacenará las
  categorías
  dec=input("¿De cuántos items deseas menores ventas? \n")
  lowventas=sorteo(int(dec),ventascontadas,'low') #Sort
  ascendente de "dec" búsquedas cuantificadas
  bajventas=sorteo(len(LSa),ventascontadas,'low') #La peor
  situación es que tuviéramos una categoría por venta, por lo
  que se genera esta lista que considera esa opción

  index=0
  print(tablav)
  for i in lowventas:
    index+=1
    print( "| ",index, "      ", i[1][0],"       ",i[0])

  index=0
  inout=input("¿Deseas consultar por categoría? (y/n) \n")
  if inout=='y':
    print("\nCategorías disponibles:")

    #El loop siguiente itera por todos los índices de
    lifestore_Products y numera las diferentes categorías
    encontradas
    for i in LP:
      if i[3] not in dcateg:
        index+=1
        print(index,".- ",i[3])
        dcateg.append(i[3])

    ca=input("Elige tu categoría: \n")
    ca=int(ca)
    cat=dcateg[ca-1] #La categoría elegida sería de n-1 ya que
    nuestras listas empiezan con índice 0
    num=input("Elige cuántos items quieres de esa categoría:
    \n")
    num=int(num)

    print("Categoría elegida: ",cat)
    print(tablav)

    for i in bajventas:
      if i != 'Out of range':
        if i[1][1]==cat:
          categoriaelegida.append(i) #Va introduciendo
          dependiendo la categoría que eligió el usuario los
          elementos correspondientes a la misma

    for i in range(num): #Itera el número de veces que el
    usuario haya elegido
      if i<len(categoriaelegida):
        resultadocat.append(categoriaelegida[i])
      else: resultadocat.append('Out of range') #Siempre
      teniendo en cuenta si excede los datos existentes en la
      lista

    index=0
    for i in resultadocat:
      index+=1
      print( "| ",index, "      ", i[1][0],"       ",i[0])

    inout=input("¿Deseas hacer otra consulta? (y/n) \n")
    if inout=='y':
      continue
    else:
      print("Hasta la vista, ser etéreo")
      exit(1)
```

```
¿De cuántos items deseas menores ventas?
4
|  Lugar  |  ID  |  Ventas   |
|  1        10        1
|  2        13        1
|  3        17        1
|  4        22        1
¿Deseas consultar por categoría? (y/n)
y

Categorías disponibles:
1 .-  procesadores
2 .-  tarjetas de video
3 .-  tarjetas madre
4 .-  discos duros
5 .-  memorias usb
6 .-  pantallas
7 .-  bocinas
8 .-  audifonos
Elige tu categoría:
1
Elige cuántos items quieres de esa categoría:
4
Categoría elegida:  procesadores
|  Lugar  |  ID  |  Ventas   |
|  1        1        2
|  2        6        3
|  3        8        4
|  4        7        7
¿Deseas hacer otra consulta? (y/n)
```

Opción 4

En caso de elegir “Menores búsquedas” el código y output es el siguiente:

```
elif decision==4: #Búsquedas menores
  dec=input("¿De cuántos items deseas menores búsquedas? \n")
  lowbusq=sorteo(int(dec),busquedascontadas,'low')  #Sort
  ascendente de "dec" búsquedas cuantificadas

  print(tablab)

  for i in lowbusq:
    index+=1
    print( "| ",index, "      ", i[1],"       ",i[0])

  inout=input("¿Deseas hacer otra consulta? (y/n) \n")
  if inout=='y':
    continue
  else:
    print("Hasta la vista, corazón")
    exit(1)
```

```
¿De cuántos items deseas menores búsquedas?
6
|  Lugar  |  ID  | Búsquedas |
| 1       9         1
| 2       10         1
| 3       27         1
| 4       35         1
| 5       45         1
| 6       59         1
¿Deseas hacer otra consulta? (y/n)
```

Opción 5

En caso de elegir “Mejores reseñas” el código y output es el siguiente:

```
elif decision==5: #Mejores reseñas
  print("Se considera que los artículos con buenas reseñas son
  los que tienen de 4 a 5 estrellas\n")
  dec=input("¿De cuántos items deseas mejores reseñas? \n")
  bestrev=sorteo(int(dec),ratingaltocontado,'high')  #Sort
  descendente de "dec" ratings buenos cuantificados

  print(tablar)
  for i in bestrev:
    index+=1
    print( "| ",index, "      ", i[1][0],"       ",i[0])

  print("\nLos IDs de productos que recibieron reviews tanto
  buenas como malas son ")
  for i in ratingdual:
    print(i[0])

  inout=input("¿Deseas hacer otra consulta? (y/n) \n")
  if inout=='y':
    continue
  else:
    print("Hasta la vista, chulísimo")
    exit(1)
```

```
Se considera que los artículos con buenas reseñas son los que tienen de
 4 a 5 estrellas

¿De cuántos items deseas mejores reseñas?
6
|  Lugar  |  ID  |  #Ratings |
| 1       54        49
| 2       3        41
| 3       5        20
| 4       42        18
| 5       57        15
| 6       2        12

Los IDs de productos que recibieron reviews tanto buenas como malas son

3
4
29
31
31
47
48
¿Deseas hacer otra consulta? (y/n)
```

Opción 6

En caso de elegir “Peores reseñas” el código y output es el siguiente:

```
elif decision==6: #Peores reseñas
  print("Se considera que los artículos con malas reseñas son
  los que tienen de 1 a 3 estrellas y se documenta si hubo
  reembolso\n")
  dec=input("¿De cuántos items deseas peores reseñas? \n")
  worstrev=sorteo(int(dec),ratingbajocontado,'high')  #Sort
  descendente de "dec" ratings malos cuantificados

  print(tablabr)
  for i in worstrev:
    index+=1
    if i!='Out of range':
      if i[1][1]==1: #En esta sección se revisa si tuvieron
      devolución y se cambia el 1/0 por Sí/No
        i[1][1]='Sí'
      else:
        i[1][1]='No'
      print( "| ",index, "       ", i[1][0],"       ",i[0], "
            ",i[1][1])
    else: print( "| ",index, "       ", "---","      ","---", "
         ","---") #En caso de que la petición esté fuera del
    rango, imprime esto

  print("\nLos IDs de productos que recibieron reviews tanto
  buenas como malas son")
  for i in ratingdual:
    print(i[0])

  inout=input("¿Deseas hacer otra consulta? (y/n) \n")
  if inout=='y':
    continue
  else:
    print("Hasta la vista, pythonista")
    exit(1)
```

```
Se considera que los artículos con malas reseñas son los que tienen de
1 a 3 estrellas y se documenta si hubo reembolso

¿De cuántos items deseas peores reseñas?
7
|  Lugar  |  ID  |  #Ratings |  Reembolso  |
|  1      31        3           Sí
|  2      31        2           No
|  3      2         1           Sí
|  4      3         1           No
|  5      4         1           No
|  6      17        1           Sí
|  7      29        1           Sí

Los IDs de productos que recibieron reviews tanto buenas como malas son
3
4
29
31
31
47
48
¿Deseas hacer otra consulta? (y/n)
```

Opción 7

En caso de elegir “Total de ingresos” el código y output es el siguiente:

```
elif decision==7: #Total de ingresos
  print("Tu venta total es de: ",Ventatotal[0], "\n")
  print("Siendo el promedio de: ",Ventatotal[1], "\n")

  inout=input("¿Deseas hacer otra consulta? (y/n) \n")
  if inout=='y':
    continue
  else:
    print("Hasta la vista, mi estimado")
    exit(1)
```

```
Tu venta total es de:  737916.0

Siendo el promedio de:  2693.124087591241

¿Deseas hacer otra consulta? (y/n)
```

Opción 8

En caso de elegir “Ventas promedio mensuales” el código es el siguiente:

```
elif decision==8: #Consulta de venta por mes
  mes=input("Elige el mes a consultar: (1-12) \n")
  mes=int(mes)
  print(tablam)
  print( "| ",Ventapormeses[mes-1][0], "     ", Ventapormeses
  [mes-1][1][0],"    ",Ventapormeses[mes-1][1][1])
  inout=input("¿Deseas hacer otra consulta? (y/n) \n")

  if inout=='y':
    continue
  else:
    print("Hasta la vista, bonito")
    exit(1)
```

```
Elige el mes a consultar: (1-12)
6
|  Mes  |  Ventas  |  Promedio |
|  6       36949.0      3359.0
¿Deseas hacer otra consulta? (y/n)
```

Opción 9

En caso de elegir “Meses con más ventas” el código es el siguiente:

```
elif decision==9: #Meses con más ventas
  mesalto=[] #Variable que guarda los valores de n meses más
  altos
  mesnum=input("Elige de cuántos meses quieres las mayores
  ventas: \n")
  mesnum=int(mesnum)

  ventasmayores=Ventapormeses
  ventasmayores.sort(reverse=True, key=lambda x: x[1][0])
  #Sorteo descendente tomando como referencia el valor que está
  en n[1][0]

  for i in range(mesnum): #Añade los valores de los n meses que
  el usuario elija
    mesalto.append(ventasmayores[i])

  print(tablam)

  for i in mesalto:
    print( "| ",i[0], "    ", i[1][0],"   ",i[1][1])

  inout=input("¿Deseas hacer otra consulta? (y/n) \n")
  if inout=='y':
    continue
  else:
    print("Hasta la vista, EmTecher")
    exit(1)
```

```
Elige de cuántos meses quieres las mayores ventas:
7
|  Mes  |  Ventas  |  Promedio |
|  4       191066.0     2581.972972972973
|  3       162931.0     3325.122448979592
|  1       117738.0     2264.1923076923076
|  2       107270.0     2681.75
|  5       91936.0     2704.0
|  6       36949.0     3359.0
|  7       26949.0     2449.909090909091
¿Deseas hacer otra consulta? (y/n)
```

Solución al problema

Resultados

A partir del análisis de la información, se llega a lo siguiente:

Los 15 productos **más vendidos** son:

ID	Categoría	Ventas
54	Discos duros	50
3	Procesadores	42
5	Procesadores	20
42	Tarjetas madre	18
57	Discos duros	15
29	Tarjetas madre	14
2	Procesadores	13
4	Procesadores	13
47	Discos duros	11
12	Tarjetas de video	9
48	Discos duros	9
7	Procesadores	7
31	Tarjetas madre	6
44	Tarjetas madre	6
18	Tarjetas de video	5

Los 20 productos **más buscados** son:

ID	Categoría	Búsquedas
54	Discos duros	263
57	Discos duros	107
29	Tarjetas madre	60
3	Procesadores	55
4	Procesadores	41
85	Audífonos	35
67	Pantallas	32
7	Procesadores	31
5	Procesadores	30
47	Discos duros	30
48	Discos duros	27
44	Tarjetas madre	25
2	Procesadores	24
42	Tarjetas madre	23
8	Procesadores	20
12	Tarjetas de video	15
21	Tarjetas de video	15
66	Pantallas	15
18	Tarjetas de video	11
51	Discos duros	11

Los 5 productos **menos vendidos** por categoría son:

Categoría	IDs	Ventas
Procesadores	1, 6, 8, 7, 2	2, 3, 4, 7, 13
Tarjetas de video	10, 13, 17, 22, 28	1, 1, 1, 1, 1
Tarjetas madre	40, 45, 46, 33, 31	1, 1, 1, 2, 6
Discos duros	50, 52, 49, 51, 48	1, 2, 3, 3, 9
Memorias USB	60	1
Pantallas	66, 67	1, 1
Bocinas	74	2
Audífonos	84, 89, 94, 85	1, 1, 1, 2

Los 20 productos **menos buscados** son:

ID	Categoría	Búsquedas
9	Procesadores	1
10	Tarjetas de video	1
27	Tarjetas de video	1
35	Tarjeta madre	1
45	Tarjeta madre	1
59	Disco duro	1
70	Pantallas	1
80	Bocinas	1
93	Audífonos	1
13	Tarjeta de video	2
56	Disco duro	2
76	Bocina	2
91	Audífonos	2
17	Tarjetas de video	3
39	Tarjetas madre	3
95	Audífonos	3
15	Tarjetas de video	4
46	Tarjetas madre	4
63	Pantallas	4
73	Pantallas	4

Los 10 productos con **mejores reseñas** son:

ID	# Reseñas
54	49
3	41
5	20
42	18
57	15
2	12
4	12

29	12
47	10
12	9

Los 10 productos con **peores reseñas** agrupados por devolución son:

ID	# Reseñas	¿Devolución?
31	3	Sí
31	2	No
2	1	Sí
3	1	No
4	1	No
17	1	Sí
29	1	Sí
29	1	No
45	1	Sí
46	1	Sí

El total de ingresos es: $**737916**

Con promedio de: $**2693.124**

Las **ventas por mes** son las siguientes

Mes	Venta	Promedio
1	$117738	$2264.2
2	$107270	$2681.75
3	$162931	$3325.12
4	$191066	$2581.97
5	$91936	$2704
6	$36949	$3359
7	$26949	$2450
8	$3077	$1025.6
9	0	N/A
10	0	N/A
11	0	N/A
12	0	N/A

Mientras que los 3 **meses con más ventas** son:

Mes	Venta	Promedio
4	$191066	$2581.97
3	$162931	$3325.12
1	$117738	$2264.2

Análisis de resultados

El producto estrella de la tienda es sin pensarlo el ID 54: SSD Kingston 120GB siendo el más vendido, buscado y con mejores reseñas. Los productos más vendidos guardan una estrecha relación con los productos más buscados. Las memorias USB, pantallas, audífonos y bocinas son de los productos menos vendidos y comprados, por lo que la verdadera fortaleza de la tienda son los procesadores, tarjetas madre, tarjetas de video y memorias SSD. Así mismo, con mucha lógica se mantiene el hecho de que los productos más vendidos son los que mejores reseñas tienen. El producto con ID 31 se mantiene en la tabla de mayores ventas y al mismo tiempo es el que tuvo más devoluciones y malas calificaciones. El declive de ventas se dio en marzo llegando a niveles catastróficos de ventas desde agosto a diciembre.

A partir de la relación de los ítems con menos ventas por categoría, sería factible retirar ese hardware de la tienda. También, sería bueno considerar focalizar el mercado de la tienda hacia una mayor variedad de procesadores y discos duros, erradicando los accesorios externos de la computadora como memorias USB, pantallas, audífonos y bocinas. Es importante retirar el producto con ID 31 ya que está generando una mala reputación en la tienda. Para el control de inventario, una sugerencia sería que el número de productos por producto no supere los 50, ya que observando la tendencia de ventas un número máximo de 50 productos sería suficiente para surtir a los clientes por un año entero.

Conclusión

A partir del análisis de datos fuimos capaces de encontrar la relación de productos y sus ventas, búsquedas y calificaciones. Tuvimos la oportunidad de crear una interfaz real basada en análisis de datos y orientamos al cliente basando nuestras sugerencias en datos generados de su tienda. También pudimos analizar el nivel de ingresos por mes de la tienda, identificando cuándo empezó su declive y cuáles fueron los meses con más ventas. Por otro lado, fuimos capaces de aplicar los conocimientos que aprendimos en todo el bloque 1 de nuestro curso. Fue gratificante demostrar los conocimientos adquiridos de esta forma.

El programa en GitHub se puede encontrar aquí: https://github.com/coolaxtro/EmTech.git