EMTECH

Emerging Technologies Institute

Proyecto final de fundamentos de programación con Python

ESTUDIANTE: GUSTAVO XAVIER HERNÁNDEZ ARCE

UNIVERSIDAD DEL CARIBE, CANCÚN, Q.ROO

LINK DE GITHUB: |https://github.com/GustHer98/EmtechProyect

EMTECH
Emerging Technologies Institute

Índice

Contenido

EMTECH
Emerging Technologies Institute

Introducción

En ese proyecto se abordará la situación de la tienda lifestore, una tienda virtual que maneja una amplia gama de artículos. Con la ayuda de Python y los conocimientos adquiridos a lo largo del curso, se hará un análisis y clasificación de datos para tratar la acumulación de inventario, la reducción de búsquedas de productos y reducción de ventas en el último trimestre del año. Para al final, sugerir estrategias que ayuden a la empresa a mejorar su situación en la mayor cantidad de ámbitos posible, dejándonos también, un aprendizaje y reflexión sobre lo importante que es tomar cursos de este tipo para mejorar nuestro conocimientos y habilidades, y que podamos hacer aportaciones significativas para mejorar la situación de cualquier empresa que lo solicite.

Definición del código

1. Login

Para el Login se declararon dos listas, una para los usuarios administradores con sus respectivas contraseñas y otra para los usuarios normales, los que no tienen acceso a visualizar nada. Con la función input se pide un usuario y contraseña, que se evaluarán y se mostrará un mensaje de acuerdo con lo que se ingrese por teclado.

```
admins =[["admin1" , "root1"], ["admin2", "root2"], ["admin3", "root3"]]

#Lista con las credenciales de usuario normal
usuarios_normales = [["ventas1", "123"], ["ventas2", "456"], ["ventas3", "789"]]
usr = input("Ingrese usuario: ")
passw = input ("Ingrese contraseña: ")
```

Utilizo las variables “es_admin” y “es_user” como “banderas”, las cuales se inician en 0 y cambiarán a 1 en caso de que se detecte un inicio de sesión ya sea como admin o usuario normal, esto con el fin de poner el bloque de código correspondiente en otra sección y no se amontone todo en un solo lugar. Con el ciclo for en admins, recorreré la lista de usuarios admins buscando coincidencias con lo que se ingresó por teclado, tanto en usuario como en contraseña. Si las credenciales que se ingresaron son de admin, se mostrará el menú del cual el admin podrá elegir que quiere ver.

```
#Se usa es_admin como bandera en la comparación
es_admin = 0
es_user = 0
for admin in admins:
    if admin[0] == usr and admin[1] == passw:
        es_admin = 1
        break
    else:
        continue

#Si identificamos que es admin, le mostramos el menú de administrador
if es_admin == 1:
    flag = 0
    print("\nIngreso como administrador\n")
    print("1. Visualizar productos más vendidos y productos rezagados")
    print("2. Visualizar productos por reseña de servicio")
    print("3. Visualizar total de ingresos, ventas promedio mensuales, total anual y ventas
    promedio en el año")
```

Si las credenciales ingresadas no son de admin, averiguaré si son de usuario normal, en caso afirmativo, es_user será 1 y se mostrará el mensaje de que ingresó como usuario normal pero no puede visualizar nada porque solo pueden visualizar los administradores y el programa se acabará. En caso de que no sea ni admin ni usuario normal, se mostrará un mensaje de que esas credenciales no se encuentran registradas como ninguno de los dos, y se terminará el programa.

```
else:
    #Si no es admin, averiguamos si es un usuario normal
    for user in usuarios_normales:
        if user[0] == usr and user[1] == passw:
            es_user=1
            break
        else:
            continue
    if es_user==1:
        print("Ingreso como usuario normal")
        print("Ingrese como admin para visualizar información")
    else:
        #Si no es ni admin ni usuario normal, le decimos que ingrese como uno de ellos dos
        print("Credenciales no registradas")
        print("Ingrese un usuario normal o admin registrados")
```

2. Consigna 1

Una vez que se ingresa como administrador, se pueden escoger tres opciones, que representan las 3 consignas. La opción se pide por entrada de teclado y se guardará en la variable “seleccion”. Se evalúa con ifs que opción se seleccionó y dependiendo de la opción, cambiará la variable selección, esto con el fin de no amontonar código en esa zona.

```
print("1. Visualizar productos más vendidos y productos rezagados")
print("2. Visualizar productos por reseña de servicio")
print("3. Visualizar total de ingresos, ventas promedio mensuales, total anual y ventas
promedio en el año")
opcion=int(input("\nSeleccione una opción: "))
seleccion = 0
while flag!=1:
    if opcion == 1:
        #Se usan otras dos banderas para tener el codigo en una sección diferente y no se
        amontone todo aqui
        seleccion = 1
        flag = 1
```

Si se selecciona la primera consigna, se usará una lista llamada total_ventas, se comparará cada id de producto en lifestore_products con los ids de lifestore_sales, para contar cuantas veces se repite cada uno, cada cuenta se guardará en la variable contador, cuando pase al siguiente id, el contador se reinicia. En una lista aparte se agrega el id de producto, descripción, veces que se vendió dicho producto y categoría (se hará uso de ella más adelante)

```
# 50 PRODUCTOS MAS VENDIDOS
#Comparo cada id de producto en lifestore products con los ids de lifestore sales para
contar cuantas veces se repite cada uno
contador = 0
total_ventas = []
for producto in lifestore_products:
  for venta in lifestore_sales:
    if producto[0] == venta[1]:
      contador += 1
```

Para ordenar todo lo que sea requerido, se utilizará Bubble sort adaptado a las listas que sean necesarias ordenar. En este caso, se ordenan la cantidad de ventas por producto para obtener las 50 con más ventas e imprimir sus datos.

```
long = len(total_ventas)
for i in range(long-1):
  for j in range(0,long-i-1):
    if total_ventas[j][2] < total_ventas[j+1][2]:
      total_ventas[j], total_ventas[j+1] = total_ventas[j+1], total_ventas[j]
print("\nSe mostrarán los 50 productos con mayores ventas\n")
#Imprimo los 50 productos con mayores ventas
for idx in range(0,50):
  print("El producto: ", total_ventas[idx][1], "Se vendió: ", total_ventas[idx][2])
```

Para sacar los productos con mayores búsquedas, se hará el mismo procedimiento, pero orientado a la lista lifestore_searches

```
contador = 0
total_busq = []
for producto in lifestore_products:
  for busqueda in lifestore_searches:
    if producto[0] == busqueda[1]:
      contador += 1
  formato = [producto[0], producto[1], contador]
  total_busq.append(formato)
  contador = 0
#Ordenamiento
long = len(total_busq)
for i in range(long-1):
  for j in range(0,long-i-1):
    if total_busq[j][2] < total_busq[j+1][2]:
      total_busq[j], total_busq[j+1] = total_busq[j+1], total_busq[j]
print("\nSe mostrarán los 50 productos con mayores busquedas\n")
for idx in range(0,50):
  print("El producto: ", total_busq[idx][1], "Se buscó: ", total_busq[idx][2])
```

Para sacar los productos con más ventas por categoría, se reciclará la lista total_ventas, que ya tiene la cantidad de ventas por producto ordenados. Solamente se accederá a la categoría de cada producto en la misma lista de total_ventas y se separará cada uno a una lista diferente.

```
procesadores = []
tarjetas_video = []
tarjetas_madre = []
dd = []
usb = []
pantallas = []
bocinas = []
audifonos  = []
#Separo cada categoría en una lista diferente cada categoría
for cat in total_ventas:
  if cat[3] == "procesadores":
    procesadores.append(cat)
  elif cat[3] == "tarjetas de video":
    tarjetas_video.append(cat)
  elif cat[3] == "tarjetas madre":
    tarjetas_madre.append(cat)
  elif cat[3] == "discos duros":
    dd.append(cat)
  elif cat[3] == "memorias usb":
    usb.append(cat)
  elif cat[3] == "pantallas":
    pantallas.append(cat)
```

Posteriormente solo se muestran las mayores ventas por categoría por categoría.

```
print("\nSe mostrarán los  productos MAS vendidos por Categoria")
#Imprimo los mas vendidos de cada categoría
print("Procesadores\n")
for idx in range(int(len(procesadores)/2)):
  print("El producto: ", procesadores[idx][1], "Se vendió: ", procesadores[idx][2])

print("\nTarjetas de video\n")
for idx in range(0,int(len(tarjetas_video)/2)):
  print("El producto: ", tarjetas_video[idx][1], "Se vendió: ", tarjetas_video[idx][2])

print("\nTarjetas madre\n")
for idx in range(0,int(len(tarjetas_madre)/2)):
  print("El producto: ", tarjetas_madre[idx][1], "Se vendió: ", tarjetas_madre[idx][2])
```

*Se hizo para cada categoría, la imagen solo es ilustrativa del código que se usó para cada una.

Ya teniendo también todos los productos en la lista total_ventas, esta vez se reúsa de nuevo, pero para ordenarlas de menor a mayor, se separan por categorías y se muestran los productos generales con menores ventas y después categoría por categoría.

```
#MENORES VENTAS GENERALES
#Reciclo lo que ya habia obtenido de contar la cantidad de ventas por producto
#Para ordenar esta vez de menor a mayor con Bubble sort
long = len(total_ventas)
for i in range(long-1):
  for j in range(0,long-i-1):
    if total_ventas[j][2] > total_ventas[j+1][2]:
      total_ventas[j], total_ventas[j+1] = total_ventas[j+1], total_ventas[j]
print("\nSe mostrarán los 50 productos con MENORES ventas\n")
for idx in range(0,50):
  print("El producto: ", total_ventas[idx][1], "Se vendió: ", total_ventas[idx][2])

#MENORES VENTAS POR CATEGORIA
#Separo de nuevo de la lista ya ordenada de menor a mayor de ventas por categoría
procesadores = []
tarjetas_video = []
tarjetas_madre = []
dd = []
usb = []
pantallas = []
bocinas = []
audifonos  = []
for cat in total_ventas:
  if cat[3] == "procesadores":
    procesadores.append(cat)
  elif cat[3] == "tarjetas de video":
```

Por último, para los productos con menores búsquedas, se reúsa también la lista que ya se tenía con las búsquedas totales, pero esta vez se ordena de menor a mayor y se imprimen los primeros 50. Posteriormente a esta última impresión, se termina el programa

```
#MENORES BUSQUEDAS GENERALES
#Reciclo lo que ya habia obtenido de contar la cantidad de busquedas por producto
#Para ordenar esta vez de menor a mayor con Bubble sort
long = len(total_busq)
for i in range(long-1):
  for j in range(0,long-i-1):
    if total_busq[j][2] > total_busq[j+1][2]:
      total_busq[j], total_busq[j+1] = total_busq[j+1], total_busq[j]
print("\nSe mostrarán los 50 productos con MENORES busquedas\n")
for idx in range(0,50):
  print("El producto: ", total_busq[idx][1], "Se buscó: ", total_busq[idx][2])
```

3. Consigna 2

   Para esta consigna, se realiza lo siguiente:

```
if seleccion == 2:
  #MEJORES RESEÑAS
  #Buscaré en lifestore sales, los productos con mejores reseñas, los que encuentre, guardo
  su id en la lista "resena"
  resena = []
  productos= []
  prod=0
  #Busco en lifestore sales, los productos con reseña 5, los que encuentre, guardo su id en
  la lista "resena"
  for x in lifestore_sales:
    if x[2] == 5:
      resena.append(x[1])
  #Elimino los ids repetidos en la lista resena, quedandome con ids unicos en la lista
  productos
  for i in resena:
    if i not in productos:
      productos.append(i)
  #Con la variable prod, controlo cuantos se han impreso, para solo imprimir los 20 primeros
  print("Se mostrarán los 20 productos con Mayor valoración\n")
  for z in productos:
    print("El producto: ", lifestore_products[z][1], "Tuvo una valoración de:", 5)
    prod+=1
    if prod == 20:
      break
#Utilizando la lista "productos" que ya tiene todos los ids con valoración 5 no repetidos
 separaré los ids por categoria accediendo a las categoría en la lista de lifestore
products
procesadores=[]
video=[]
madre=[]
dd=[]
usb=[]
pantalla=[]
bocinas=[]
audifonos=[]
for z in productos:
  if lifestore_products[z][3] == "procesadores":
    procesadores.append(lifestore_products[z])
  elif lifestore_products[z][3] == "tarjetas de video":
    video.append(lifestore_products[z])
  elif lifestore_products[z][3] == "tarjetas madre":
    madre.append(lifestore_products[z])
  elif lifestore_products[z][3] == "discos duros":
    dd.append(lifestore_products[z])
  elif lifestore_products[z][3] == "memorias usb":
    usb.append(lifestore_products[z])
  elif lifestore_products[z][3] == "pantallas":
    pantalla.append(lifestore_products[z])
  elif lifestore_products[z][3] == "bocinas":
    bocinas.append(lifestore_products[z])
  elif lifestore_products[z][3] == "audifonos":
```

Posteriormente, se imprimen las listas de cada categoría para mostrar los mejor valorados.

Para buscar los productos con menor valoración, se hará el mismo procedimiento que para los de mejor valoración, pero en esta ocasión se buscará los que tengan score de 1 y 2. En esta ocasión, no existieron productos con score=2. Una vez que se imprime los productos con peor valoración, se termina el programa.

```
#Productos generales con las peores reseñas
#Busco los productos con valoración de 1 y guardo lps ids en la lista resena
resena = []
productos= []
for x in lifestore_sales:
  if x[2] == 1 :
    resena.append(x[1])
#Elimino los ids repetidos
for i in resena:
  if i not in productos:
    productos.append(i)
print("\nSe mostrarán los  productos con PEOR valoración\n")
for z in productos:
  print("El producto: ", lifestore_products[z][1], "Tuvo una valoración de:", 1)
print("\nSe encontraron: ", len(productos), "Con valoración: ", 1)
print("Se encontraron: 0 productos con valoración 2")
```

4. Consigna 3

Creo una lista para cada mes del año, y ahí poder guardar los ids de los productos que se vendieron en esos meses. Después:

```
#Itero la lista lifestore sales para acceder al mes de cada venta
#Con el índice 3:5 me quedo solamente con los dos caracteres del mes. Pregunto si esa
venta pertenece a alguno de los 12 meses y si no tuvo devolución, los que cumplan, guardo
los ids en una lista aparte
#Una lista por cada mes
for x in lifestore_sales:
  if x[3][3:5] == "01" and x[4]==0:
    en.append(x[1])
  elif x[3][3:5] == "02" and x[4]==0:
    feb.append(x[1])
  elif x[3][3:5] == "03" and x[4]==0:
    mar.append(x[1])
  elif x[3][3:5] == "04" and x[4]==0:
    ab.append(x[1])
  elif x[3][3:5] == "05" and x[4]==0:
    may.append(x[1])
  elif x[3][3:5] == "06" and x[4]==0:
    jun.append(x[1])
  elif x[3][3:5] == "07" and x[4]==0:
    jul.append(x[1])
  elif x[3][3:5] == "08" and x[4]==0:
    ag.append(x[1])
  elif x[3][3:5] == "09" and x[4]==0:
    sep.append(x[1])
  elif x[3][3:5] == "10" and x[4]==0:
    octu.append(x[1])
  elif x[3][3:5] == "11" and x[4]==0:
```

Para sacar la cantidad neta vendida en productos por mes, sin contar devoluciones, se hará lo mismo que se hizo para el mes de enero, con todos los meses del año.

```
#En la lista "ingreso" se meterá el total($) de ventas por cada mes
ingreso=[]
#En esta variable se irá haciendo la suma por cada producto vendido
suma=0
#Para cada mes, verifico si tan siquiera tiene algo la lista, es decir, si hubo ventas
ese mes. Si no hubo, en la lista de ingreso, para ese mes, lo pongo en 0
if not en:
  ingreso.append(0)
else:
  #Teniendo los ids de los productos vendidos, accedo a la lista de products por id, para
  pedir el costo de cada producto e irlo sumando
  for x in en:
    #En flag almaceno el costo del producto en curso
    flag=0
    #Accedo al costo unitario de ese producto
    flag=lifestore_products[x][2]
    #Lo sumo al valor actual de la suma que se lleva en ese mes
    suma=suma+flag
  #Guardo el total de ese mes en la lista de totales por mes
  ingreso.append(suma)
  #Reinicio la variable suma para que esté limpia para el siguiente mes
  suma=0
```

Por cada uno de los meses, se toman las mismas acciones

```
#Lo mismo para cada mes, solo lo voy guarando en la  lista ingreso, por posición
if not feb:
  ingreso.append(0)
else:
  for x in feb:
    flag=0
    flag=lifestore_products[x][2]
    suma=suma+flag
  ingreso.append(suma)
  suma=0

if not mar:
  ingreso.append(0)
else:
  for x in mar:
    flag=0
    flag=lifestore_products[x][2]
    suma=suma+flag
  ingreso.append(suma)
  suma=0
```

Posteriormente se imprime el total de ventas netas, y la cantidad de productos vendidos en cada mes

```
#Imprimo el total de ventas netas por cada mes, y la cantidad de productos vendidos en
ese mes
print("\nSe mostrará el total de las ventas mensuales descontando las devoluciones\n")
print("Enero\n")
print("Total mensual de ventas: $", ingreso[0])
print("Cantidad de productos vendidos: ", len(en))

print("\nFebrero\n")
print("Total mensual de ventas: $", ingreso[1])
print("Cantidad de productos vendidos: ", len(feb))

print("\nMarzo\n")
print("Total mensual de ventas: $", ingreso[2])
print("Cantidad de productos vendidos: ", len(mar))

print("\nAbril\n")
print("Total mensual de ventas: $", ingreso[3])
print("Cantidad de productos vendidos: ", len(ab))

print("\nMayo\n")
print("Total mensual de ventas: $", ingreso[4])
print("Cantidad de productos vendidos: ", len(may))
```

Se sumas las ventas netas de cada mes para sacar el total anual y se imprime, después de divide ese mismo total entre 12 para sacar el promedio mensual. Terminado esto, se guarda el número de mes junto con la cantidad de ventas por cada mes en una lista y se ordenan los meses por mayor cantidad de ventas con bubble sort.

```
suma=0
#Sumo las ventas netas por cada mes para sacar la venta neta total del año
for f in ingreso:
  suma=f+suma
print("\nEl ingreso total anual fue de : $", suma)
#Divido entre 12 para sacar el promedio mensual
print("\nEl ingreso promedio mensual fue de : $", suma/12)

#Guardo el numero de mes y la cantidad de ventas por cada mes, en una lista, para despues
ordenar los meses de mayor a menor
ventas_mensuales=[[1,len(en)],[2,len(feb)],[3,len(mar)],[4,len(ab)],[5,len(may)],[6,len
(jun)],[7,len(jul)],[8,len(ag)], [9,len(sep)],[10,len(octu)], [11,len(nov)],[12,len(dec)]]
#Ordeno los meses de mayor a menor ventas
long = len(ventas_mensuales)
for i in range(long-1):
  for j in range(0,long-i-1):
    if ventas_mensuales[j][1] < ventas_mensuales[j+1][1]:
      ventas_mensuales[j], ventas_mensuales[j+1] = ventas_mensuales[j+1], ventas_mensuales
      [j]
```

Por último:

```
#Accedo a los 3 primeros elementos de la lista ventas_mensuales, pero como no se que mes
es el que está en qué lugar, pregunto que mes es, por el numero de mes con el que se
guardó la longitud de cada uno
print("\nSe mostrarán los 3 meses con mayor cantidad de ventas:\n")
if ventas_mensuales[0][0] == 1:
  print("El mes con mayor ventas fue: Enero", "Con: ", ventas_mensuales[0][1], "ventas")
elif ventas_mensuales[0][0] == 2:
  print("El mes con mayor ventas fue: Febrero", "Con: ", ventas_mensuales[0][1], "ventas")
elif ventas_mensuales[0][0] == 3:
  print("El mes con mayor ventas fue: Marzo", "Con: ", ventas_mensuales[0][1], "ventas")
elif ventas_mensuales[0][0] == 4:
  print("El mes con mayor ventas fue: Abril", "Con: ", ventas_mensuales[0][1], "ventas")
elif ventas_mensuales[0][0] == 5:
  print("El mes con mayor ventas fue: Mayo", "Con: ", ventas_mensuales[0][1], "ventas")
elif ventas_mensuales[0][0] == 6:
  print("El mes con mayor ventas fue: Junio", "Con: ", ventas_mensuales[0][1], "ventas")
elif ventas_mensuales[0][0] == 7:
  print("El mes con mayor ventas fue: Julio", "Con: ", ventas_mensuales[0][1], "ventas")
elif ventas_mensuales[0][0] == 8:
  print("El mes con mayor ventas fue: Agosto", "Con: ", ventas_mensuales[0][1], "ventas")
elif ventas_mensuales[0][0] == 9:
  print("El mes con mayor ventas fue: Septiembre", "Con: ", ventas_mensuales[0][1],
  "ventas")
elif ventas_mensuales[0][0] == 10:
  print("El mes con mayor ventas fue: Octubre", "Con: ", ventas_mensuales[0][1], "ventas")
elif ventas_mensuales[0][0] == 11:
  print("El mes con mayor ventas fue: Noviembre", "Con: ", ventas_mensuales[0][1],
```

Este proceso se realiza para las 3 primeras posiciones de la lista. Se imprimen los 3 meses con más ventas, se termina el programa y se le agrega 1 a es_admin para que se rompa el ciclo

Solución al problema

Uno de los problemas de la empresa, es la acumulación de inventario, que se presenta en diversas categorías. Según los resultados obtenidos, en todo el año solo se vendieron 2 pantallas de 2 tipos diferentes y 2 bocinas de un mismo tipo.

```
Pantallas

El producto:  TCL Smart TV LED 55S425 54.6, 4K Ultra HD, Widescreen, Negro Se vendió:  1
El producto:  TV Monitor LED 24TL520S-PU 24, HD, Widescreen, HDMI, Negro Se vendió:  1
El producto:  Makena Smart TV LED 32S2 32'', HD, Widescreen, Gris Se vendió:  0
El producto:  Seiki TV LED SC-39HS950N 38.5, HD, Widescreen, Negro Se vendió:  0
El producto:  Samsung TV LED LH43QMREBGCXGO 43, 4K Ultra HD, Widescreen, Negro Se vendió:  0
El producto:  Samsung Smart TV LED UN70RU7100FXZX 70, 4K Ultra HD, Widescreen, Negro Se vendió:  0

Bocinas

El producto:  Logitech Bocinas para Computadora con Subwoofer G560, Bluetooth, Inalámbrico, 2.1, 120W RMS, USB, negro Se vendió:
2
El producto:  Lenovo Barra de Sonido, Alámbrico, 2.5W, USB, Negro Se vendió:  0
El producto:  Acteck Bocina con Subwoofer AXF-290, Bluetooth, Inalámbrico, 2.1, 18W RMS, 180W PMPO, USB, Negro Se vendió:  0
El producto:  Verbatim Bocina Portátil Mini, Bluetooth, Inalámbrico, 3W RMS, USB, Blanco Se vendió:  0
El producto:  Ghia Bocina Portátil BX300, Bluetooth, Inalámbrico, 40W RMS, USB, Rojo - Resistente al Agua Se vendió:  0
```

Mayor cantidad de ventas en pantallas y bocinas

Otra categoría que presenta una acumulación por falta de ventas, son las tarjetas de video. Si bien el volumen de ventas es mayor que el de las categorías anteriormente mencionadas, también el catálogo de tarjetas de video es mucho mayor que el de pantallas y bocinas.

```
Tarjetas de video

El producto:  Tarjeta de Video ASUS NVIDIA GeForce GTX 1660 SUPER EVO OC, 6GB 192-bit GDDR6, PCI Express x16 3.0 Se vendió:  9
El producto:  Tarjeta de Video Gigabyte NVIDIA GeForce GT 1030, 2GB 64-bit GDDR5, PCI Express x16 3.0 Se vendió:  5
El producto:  Tarjeta de Video ASUS AMD Radeon RX 570, 4GB 256-bit GDDR5, PCI Express 3.0 Se vendió:  3
El producto:  Tarjeta de Video MSI AMD Mech Radeon RX 5500 XT MECH Gaming OC, 8GB 128-bit GDDR6, PCI Express 4.0 Se vendió:  2
El producto:  Tarjeta de Video Sapphire AMD Pulse Radeon RX 5500 XT Gaming, 8GB 128-bit GDDR6, PCI Express 4.0 Se vendió:  2
El producto:  MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PCI Express 2.0 Se vendió:  1
El producto:  Tarjeta de Video Asus NVIDIA GeForce GTX 1050 Ti Phoenix, 4GB 128-bit GDDR5, PCI Express 3.0 Se vendió:  1
El producto:  Tarjeta de Video Gigabyte AMD Radeon R7 370 OC, 2GB 256-bit GDDR5, PCI Express 3.0 Se vendió:  1
El producto:  Tarjeta de Video MSI NVIDIA GeForce GTX 1050 Ti OC, 4GB 128-bit GDDR5, PCI Express x16 3.0 Se vendió:  1
```

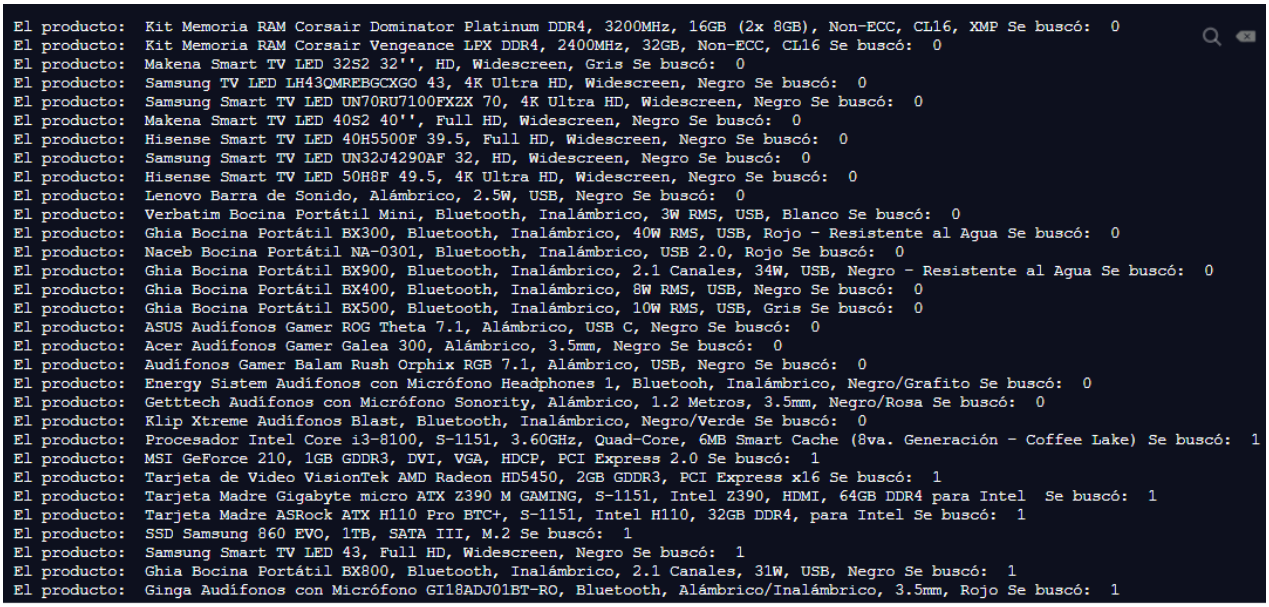
Productos con mayores ventas de tarjetas de video

```
Tarjetas de video

El producto:  Tarjeta de Video EVGA NVIDIA GeForce GT 710, 2GB 64-bit GDDR3, PCI Express 2.0 Se vendió:  0
El producto:  Tarjeta de Video EVGA NVIDIA GeForce GTX 1660 Ti SC Ultra Gaming, 6GB 192-bit GDDR6, PCI 3.0 Se vendió:  0
El producto:  Tarjeta de Video EVGA NVIDIA GeForce RTX 2060 SC ULTRA Gaming, 6GB 192-bit GDDR6, PCI Express 3.0 Se vendió:  0
El producto:  Tarjeta de Video Gigabyte NVIDIA GeForce GTX 1650 OC Low Profile, 4GB 128-bit GDDR5, PCI Express 3.0 x16 Se vendió:  0
El producto:  Tarjeta de Video Gigabyte NVIDIA GeForce RTX 2060 SUPER WINDFORCE OC, 8 GB 256 bit GDDR6, PCI Express x16 3.0 Se vendió:
  0
El producto:  Tarjeta de Video MSI Radeon X1550, 128MB 64 bit GDDR2, PCI Express x16 Se vendió:  0
El producto:  Tarjeta de Video PNY NVIDIA GeForce RTX 2080, 8GB 256-bit GDDR6, PCI Express 3.0  Se vendió:  0
El producto:  Tarjeta de Video VisionTek AMD Radeon HD 5450, 1GB DDR3, PCI Express x16 2.1 Se vendió:  0
El producto:  Tarjeta de Video VisionTek AMD Radeon HD5450, 2GB GDDR3, PCI Express x16 Se vendió:  0
```

Productos con menores ventas de tarjetas de video

Reflejo de esto, lo podemos ver en las búsquedas de productos, los productos que más aparecen en la lista de los 50 menos buscados, son las pantallas



Productos menos buscados

1. Una posible solución es hacer un recorte del catálogo de pantallas. Eliminar las pantallas con menos búsquedas y con menos ventas a la vez. Esto con el fin de reducir la inversión que hace la empresa en adquirirlas y que los clientes no se están interesando ni en buscarlas.
2. A pesar de tener una mediana venta en las tarjetas de video, se tiene registro de un buen volumen de búsquedas en las tarjetas de video, por lo cual, una posible solución podría ser evaluar la posibilidad de lanzar ofertas de estos productos o bien, reducir un poco el costo sin que la tienda pierda ganancias. Podríamos suponer que los clientes están buscando estos productos, pero no se están decidiendo a comprarlos por algún motivo.
3. Según los resultados, en el mes de septiembre y noviembre se registró una venta, pero se devolvió, en octubre y diciembre no se registró ninguna venta. Por lo cual, considero que la tienda podría invertir un poco más en marketing en este último trimestre del año. Una de las principales razones para creer en esto, es que es usual que las personas reciban aguinaldo en noviembre-diciembre, y busquen que regalar a fin de año a sus familiares. Si se logra orientar el marketing a un público enfocado en el uso de este tipo de productos tecnológicos, se puede obtener un incremento sustancial en las ventas de ese último trimestre.

EMTECH
Emerging Technologies Institute

Conclusión

Definitivamente la ciencia de datos está presente en nuestra vida cotidiana, y este proyecto es una clara muestra de que las tiendas-empresas, tienden a buscar personal que realice estas labores para optimizar sus registros. Por esto la importancia de estar al día en las tecnologías emergentes y tomar cursos de este tipo, que nos ayuden a ampliar nuestro panorama de herramientas que podemos utilizar en nuestro día a día, y ayuden a las empresas a tomar mejorar decisiones.

En este caso, la tienda lifestore necesita hacer ajustes en su catálogo de ventas para aligerar su inventario, eliminar del catálogo los productos que los clientes ni siquiera se interesan por buscar, además de hacer un análisis más profundo de porque los clientes están buscando las tarjetas de video, pero no se están interesando tanto por hacer la compra. Finalmente, una mejora en la estrategia de marketing en el último tramo del año, considero que puede mejorar las ventas de la tienda y por consecuencia, reducir el inventario acumulado.