Proyecto 1: Introducción a Python

Gómez Bravo Brandon Saúl

3 de septiembre de 2020

Resumen

En este trabajo se muestra un análisis descriptivo de las ventas y búsquedas de los productos ofrecidos por la tienda *LifeStore*. Se muestra el código hecho para analizar los datos. Además, las posibles soluciones para generar una mayor cantidad de ventas, y por lo tanto de ingresos.

Índice

I. Introducción

LifeStore es una tienda virtual, la cual ha acumulado un basto inventario. Los datos que ha almacena son:

- Características del producto: el id del producto, el nombre, precio, categoría y productos en almacén.
- Ventas: id de la venta, id del producto, la reseña del producto hecha por el comprador, la fecha de venta y si el producto fue devuelto.
- Búsquedas: guarda el id de la búsqueda y el id del producto buscado.

Con estos datos se puede hacer un análisis de los productos clasificándolos por la categoría a la que pertenecen. Además, se puede ver los productos más y menos vendidos, así como los más y menos buscados. A partir de las reseñas hechas podemos obtener una noción de los productos que necesitan salir del mercado.

II. Detalles del código

Primero se introduce el login-usuario, para lo cual el usuario debe introducir el nombre de usuario y la contraseña. En este se tiene:
Nombre de usuario: Gerente
Contraseña: 1234
El programa no continua hasta que el nombre y la contraseña sean dados correctamente.

```
1 import lifestore_file as data #Primero importamos los datos con el nombre de data
2
3 boleano_usuario = True #Boleano hasta que el usuario haya sido el correcto
4 boleano_contrase a = True #Boleano hasta que la contrase a sea la correcta
5
```

```
print("Ingrese el nombre del usuario:")
while boleano_usuario :
    usuario = input()
    if usuario != "Gerente": #Si el nombre es distinto de gerente, tiene que volver a
    ingresar el nombre
        print("El usuario no est  registrado en el sistema")
    else:
        boleano_usuario = False

print("Ingrese la contrase a")
while boleano_contrase a :
    contrase a = input()
    if contrase a != "1234": #Si el nombre es distinto de gerente, tiene que volver a
    ingresar el nombre
        print("Contrase a incorrecta")
    else:
        boleano_contrase a = False
```

Después se define el código que ayuda a ordenar una lista de listas. Si se tiene una lista del tipo $[[a_1, b_1, c_1], [a_2, b_2, c_2], [a_3, b_3, c_3]]$. Supongamos que $b_3 < b_1 < b_2$, entonces si está lista se decide ordenar por el segundo elemento se va a obtener $[[a_3, b_3, c_3], [a_1, b_1, c_1], [a_2, b_2, c_2]]$. Esta función es de gran ayuda pues en el primer elemento podemos poner el id del producto y en el segundo pueden ser las ventas, busquedas, etc. Y así obtener una lista ordenada y con el id de los productos bien identificados.

```
#Debido a que se ocupa el mismo procedimiento para casi todos los pasos
# Es mejor definir una funci n en vez de escribir el codigo decenas de veces

def Sorteo(Data, x): #Data es el array y x el  ndice  por el que se quiere ordenar
    Data_copy = [i for i in Data] #Generamos una copia Data, para no modificar el array
    original
    num_data = len(Data_copy) #El n mero de datos que se tienen
    for i in range(0, num_data):
        for j in range(0, num_data-i-1):
            if (Data_copy[j][x] > Data_copy[j+1][x]): #Comparamos el j-esimo y j+1-esimo
     elementos, por su entrada "x"
                ref = Data_copy[j] #Guardamos la j-esima lista
                Data_copy[j] = Data_copy[j+1] #Se reordenan
                Data_copy[j+1] = ref #Lo que sea hace es intercambiar de posicion  entre
     j y j+1
    return Data_copy #Regresa la lista de listas ordenadas por el x- simo  elemento

num_productos = len(data.lifestore_products) #Vemos cu ntos productos hay
```

Posteriormente se procede a mostrar los productos con mayores y menores ventas. Aquí radica la importancia de la función anterior pues. En el primer elemento se coloca el id del producto y en la segunda entrada el número de ventas de dicho producto. Así se tiene un array ordenado por el número de ventas y el id del producto identificado.

Además se introduce un input(), el problema original consistía de mostrar los 50 productos más y menos vendidos, pero puede ser que al usuario no quiera ver tantos elemento o quiera ver más elementos. Es por ello que el código le pregunta al usuario cuántos elementos desea ver.

```
print('A continuaci n se muestra la lista de los productos m s vendidos y los menos vendidos \n   Cuntos   elementos desea ver? (Lo recomendado es 50)')
print(f'** Debe ser un n mero entero menor a {num_productos}**') #Preguntamos al usuario cuantos elementos desea ver
ventas_num = int(input())
print()
print(f'Se muestran el n mero de ventas y los detalles de los {ventas_num} m s vendidos')
print()
ventas_por_producto = [[i+1,0] for i in range(0, num_productos)] #Generamos una lista de lista
#La primera entrada es el id del producto y el segundo va a ser el n mero de ventas de dicho producto

for i in data.lifestore_sales: #Revisamos todas las ventas
    index = i[1]-1 #El  ndice  correspondiente al id_producto. El menos uno es debido a que Python empieza en 0
    ventas_por_producto[index][1] += 1 #Aumentamos el valor del n mero de ventas de ese producto

ventas_por_producto_ord = Sorteo(ventas_por_producto, 1)   #Lo ordenamos por el indice 1, es decir, el n mero de ventas

for i in range(num_productos, num_productos-ventas_num, -1):#Recordemos la lista al reves, para empezar por los mas vendidos
    producto_id = ventas_por_producto_ord[i-1][0]-1 #Imprimimos el n mero de ventas y los detalles del producto
    print(ventas_por_producto_ord[i-1][1], data.lifestore_products[producto_id])

print()
print(f'Se muestran el n mero de ventas y los detalles de los {ventas_num} menos vendidos')
print()

for i in range(0, ventas_num, 1): #Lo mismo pero ahora empezamos la lista de menor a mayor
    producto_id = ventas_por_producto_ord[i][0]-1
    print(ventas_por_producto_ord[i][1],data.lifestore_products[producto_id])
```

Para los más y menos buscados, es similar al paso anterior. De igual manera le preguntamos al usuario cuántos elementos desea ver. Y genera una lista de listas, la cual es ordenada de acuerdo al número de búsquedas para tener como resultado una lista ordenada con el número de búsquedas y el id del producto.

```
busquedas_por_producto = [[i+1,0] for i in range(0, num_productos)] #lista de listas, el primero es el id del producto, el segundo el numero de busqeudas
for i in data.lifestore_searches: #Revisamos todas las busquedas
    index = i[1]-1 #El  ndice  correspondiente al id_producto
    busquedas_por_producto[index][1] += 1 #Aumentamos el valor del n mero de ventas de ese producto

busquedas_por_producto_ord = Sorteo(busquedas_por_producto, 1) #Ordenamos los elementos por el n mero de busquedas

print()
print()
print('A continuaci n se muestra la lista de los productos m s buscados y los menos buscados \n   Cuntos   elementos desea ver? (Lo recomendado es 50)')
print(f'** Debe ser un n mero entero menor a {num_productos}**')
busquedas_num = int(input())    #PEddimos un numero entero de elementos que el usuario desee ver
print()
print(f'Se muestran el n mero de busquedas y los detalles de los {busquedas_num} m s buscados')
print()
for i in range(num_productos, num_productos-busquedas_num, -1): #Empezamos al rev s para ir del m s buscado al menos
    producto_id = busquedas_por_producto_ord[i-1][0]-1
    print(busquedas_por_producto_ord[i-1][1],data.lifestore_products[producto_id])
print()
print()
print(f'Se muestran el n mero de busquedas y los detalles de los {busquedas_num} menos buscados')
print()
for i in range(0, busquedas_num, 1):
    producto_id = busquedas_por_producto_ord[i][0]-1 #Imprimimos el n mero de busquedas y los detalles del producto
    print(busquedas_por_producto_ord[i][1],data.lifestore_products[producto_id])
```

Para los elementos ordenados por la reseña es igual. Se le pregunta al usuario cuántos elementos desea ver y se generar una lista de listas. La diferencia es que para los productos los que tienen score = 0, no son tomados en cuenta como las "peores reseñas" simplemente no han sido calificados, por lo tanto son ignorados. Esto es porque score=0 significa que no han sido calificados, mas no que tienen "malas reseñas".

```
promedio_score = [[i+1, 0, 0, 0] for i in range(0, num_productos)]
#Lista de listas, el primero es el id del producto, el segundo es la suma de las rese as
#la tercera es el n mero de rese as y la cuarta es el promedio, es decir,, la segunda entre la primera
for i in data.lifestore_sales:
    index = i[1]-1
    promedio_score[index][1] += i[2]
    promedio_score[index][2] += 1
for i in promedio_score: #Para obtener el promedio nos fijamos que no se divida entre 0
    if i[2] != 0:
        i[3] = i[1]/i[2]

print()
print()
print('A continuaci n se muestra la lista de los productos con mejores y peores rese as \n   Cuntos   elementos desea ver? (Lo recomendado es 20)')
print(f'** Debe ser un n mero entero menor a 42**') #Debe ser 42 porque si su score es 0, significa que no ha tenido ventas
num_rese as = int(input())
promedio_score_ord = Sorteo(promedio_score, 3) #Ordenamos la lista de listas respecto al promedio
print(f'Se muestran el score y los detalles de los {num_rese as} con mejores rese as')
print()
for i in range(num_productos,num_productos-num_rese as, -1): #Empezamos de mayor  a menor
    producto_id = promedio_score_ord[i-1][0]-1 #id_product
    print(promedio_score_ord[i-1][3],data.lifestore_products[producto_id])
print()
print()
print(f'Se muestran el score y los detalles de los {num_rese as} con las peores rese as')
print()
flag = 0
indice = 0
while (flag < num_rese as): #Aqui es con un while porque si su score 0 no signifca que haya tenido la peor rese a, sino que no ha tenido rese as
    if promedio_score_ord[indice][3] != 0:
        producto_id = promedio_score_ord[indice][0]-1
        print(promedio_score_ord[indice][3],data.lifestore_products[producto_id])
        flag += 1
    indice += 1
```

Para las categorías, primero revisamos qué categorías hay y las guardamos en un array. Después se revisa los productos vendidos y la categoría a la que pertenecen de manera que se tenga un conteo promedio de ventas por categoría, así como el ingreso por categoría. De igual manera, se obtiene el número de búsquedas por categoría.

```
categorias = [] #Array donde vamos a guardar el nombre de las categorias
for i in data.lifestore_products: #recorremos todos los productos
    categoria_pro = i[3] #Obtenemos la categoria del producto
    if categoria_pro not in categorias: #Si no ha sido agregada a categorias
        categorias.append(categoria_pro) #La agregamos

num_categorias = len(categorias) #Guardamos el n mero de categorias hechas

#Lista de listas, la primera la categoria y la segunda el n mero de ventas de dicha
    categoria
categorias_ventas = [[categorias[i],0] for i in range(0, num_categorias)]
#Lista de listas, la primera la categoria y la segunda el n mero de busquedas de dicha
    categoria
categorias_busquedas = [[categorias[i],0] for i in range(0, num_categorias)]

for i in data.lifestore_searches: #Revisamos todas las busqeudas hechas
    id_producto = i[1]-1 #Id del producto
    tipo_de_produto = data.lifestore_products[id_producto][3] #Vemos la categoria del
    producto
    for j in range(0, num_categorias):
        checar_categoria = categorias[j] #Tenemos que ver que categoria es
        if tipo_de_produto == checar_categoria:
            categorias_busquedas[j][1] += 1 #Agregamos +1 a la categoria que pertence
            break

for i in data.lifestore_sales: #Lo mismo que la anteior pero ahora con las ventas
    id_producto = i[1]-1
    tipo_de_produto = data.lifestore_products[id_producto][3]
    for j in range(0, num_categorias):
        checar_categoria = categorias[j]
        if tipo_de_produto == checar_categoria:
            categorias_ventas[j][1] += 1
            break

categorias_ventas_ord = Sorteo(categorias_ventas, 1) #Ordenamos las categorias por el
    numero de ventas
categorias_busquedas_ord = Sorteo(categorias_busquedas, 1) #Ordenamos las categorias por
     el numero de busquedas

print()
print()
print("A continuaci n se muestra el n mero de busquedas por categoria:")
print()
for i in range(num_categorias-1, -1, -1): #Recorremos las categorias de mayores
    busquedas a menores
    cate = categorias_busquedas_ord[i]
    print(cate[1], cate[0]) #Imprimos el numero de busquedas y la categoria

print()
print()
print("A continuaci n se muestra el n mero de ventas por categoria:")
print()
for i in range(num_categorias-1, -1, -1): #Lo mismo pero para ventas
    cate = categorias_ventas_ord[i]
    print(cate[1], cate[0])
```

Por último se imprime las número de ventas por mes y el número de búsquedas por mes. Además, del ingreso total anual.

```
#Lista de listas, el primero es el nombre del mes
#el segundo el mes en numero, el tercero el numero de ventas en ese mes
#El cuarto el numero de ingresos en dicho mes
ventas_meses = [['Enero',1,0,0],['Febrero',2,0,0],['Marzo',3,0,0],
                ['Abril',4,0,0],['Mayo',5,0,0],['Junio',6,0,0],
                ['Julio',7,0,0],['Agosto',8,0,0],['Septiembre',9,0,0],
                ['Octubre',10,0,0],['Noviembre',11,0,0],['Diciembre',12,0,0]]

for i in data.lifestore_sales: #Recorremos las ventas
    mes_index = int(i[3][3:5])-1 #obtenemos el indice del mes
    prod_index = i[1]-1 #El producto del indice
    if i[4] == 0: #La cuarta entrada del producto es si fue regresado
        #Si fue regresado no lo contamos como venta y no contamos los ingresos
        precio = data.lifestore_products[prod_index][2] #Obtenemos el precio
        ventas_meses[mes_index][2] += 1 #Aumentamos una venta
        ventas_meses[mes_index][3] += precio #El ingreso de dicha venta

#Ordenamos por el segundo indice, es decir, por el numero de ventas
mes_numero_ventas_ord = Sorteo(ventas_meses, 2)
#Ordenamos por el tercer indice, es decir, por el ingreso obtenido
mes_ingresos_ord = Sorteo(ventas_meses, 3)

print()
print()
print('A continuaci n se muestra el mes y el n mero de ventas:')
print()
#Lo recorremos la rev s para mostrar primero los meses con mayores ventas
for i in range(len(mes_numero_ventas_ord)-1,-1,-1):
    print(mes_numero_ventas_ord[i][0],':', mes_numero_ventas_ord[i][2])
print()
print()
print('A continuaci n se muestra el mes y el n mero de ingresos:')
print()
#Lo recorremos la rev s para mostrar primero los meses con mayores ingresos
for i in range(len(mes_ingresos_ord)-1,-1,-1):
    print(mes_ingresos_ord[i][0],':', mes_ingresos_ord[i][3])

ingreso_total = 0
print()
print()
for i in ventas_meses:
    ingreso_total += i[3] #Aumentamos el numero de ingresos por mes para obtener el
    total
print('Total de ingreso anual:', ingreso_total) #Imprimos el ingreso total del a o
```

Esta última parte del código, no le imprime nada al usuario, pero nos sirve para obtener el número de diferentes productos de una categoría y el total de productos guardados en stock de dicha categoría. La cual va a servir para el análisis.

```
#Lista de lista, el primero es el nombre de la categoria
#El segundo es el n mero de distintos producots
#El tercero es el total de productos, de esta categoria, guardados en stock
categorias_stock = [[categorias[i],0,0] for i in range(0, num_categorias)]
for i in data.lifestore_products: #Revisamos los productos
    tipo_de_produto = i[3]
    for j in range(0, num_categorias):
        checar_categoria = categorias[j]
        if tipo_de_produto == checar_categoria:
            categorias_stock[j][1] += 1 #Agregamos mas un producto encontrado
            categorias_stock[j][2] += i[4] #Agregamos cuantos hay en stock del mismo
    producto
            break

#Lista de liista, el primero es el nombre de la categoria
#el segundo los ingresos de los productos vendidos
categorias_ingresos = [[categorias[i],0] for i in range(0, num_categorias)]
for i in data.lifestore_sales: #Revisamos en las ventas
    id_producto = i[1]-1
    tipo_de_produto = data.lifestore_products[id_producto][3]
    costo = data.lifestore_products[id_producto][2]
    for j in range(0, num_categorias):
        checar_categoria = categorias[j]
        if tipo_de_produto == checar_categoria:
            categorias_ingresos[j][1] += costo #Guardamos el ingreso por categoria
            break
```

III. Análisis

A continuación se presenta el análisis de los datos obtenidos por categoría.

Análisis por categoría					
Categoría	**Productos**	**Total stock**	**Busquedas**	**Ventas**	**Ingresos**
Procesadores	9	1821	222	104	371,726
Tarjetas de video	19	349	82	26	136,224
Tarjetas madre	18	814	137	49	127,321
Discos duros	13	433	463	94	93,496
Memorias usb	2	15	0	1	2,519
Pantallas	12	1190	56	2	11,278
Bocinas	10	146	9	2	8,478
Audifonos	13	573	64	5	9,135

Cuadro 1: Análisis de las búsquedas y ventas por categoría.

- Procesadores: se tiene los mayores ingresos y ventas. En stock hay mucha mercancía, pero si revisamos las reseñas de los procesadores, todas están arriba de 4. Debido a esto, los procesadores es la categoría más importante en todo nuestro inventario. En este caso, se recomienda mantener

los mismos precios y la misma cantidad en stock porque es claro que hay grandes ventas en esta categoría.

- Tarjetas de vídeo: es la segunda categoría con mayores ingresos. A pesar de tener una gran cantidad en stock (349), las tarjetas de vídeo con mejor reseña (id: 12, 21, 22) no hay en existencia, por lo que se recomienda adquirir más de estos productos.

- Tarjetas madre: la tercera categoría con mayores ingresos. No hay en existencias los productos con mejores reseñas (id: 40, 42, 44) y hay una gran cantidad para los productos con peores reseñas (id: 31, 45, 46). Es por ello que se recomienda retirar del mercado los productor id : 31, 45 y 46. Y adquirir más de id: 40, 42 y 44. Así se tendría mayores ventas e ingresos en la categoría de Tarjetas madre.

- Discos duros: la categoría más buscadas. En general, los productos vendidos no han tenido malas referencias, sin embargo, los discos duros con id = 53, 55 58 no han tendido busquedas ni ventas. Por ello se recomienda hacer más publicidad y/o promociones a estos productos para que generen atracción a los clientes.

- Memorias USB: Las memorias USB son las que menos ingresos ha generado, pero es de esperarse debido al costo general de este tipo de productos. En general hay pocos en stock, pero está bien porque no hay muchas busquedas ni ventas. La única recomendación es hacer más publicidad a estos productos.

- Pantallas: esta categoría ha tenido muy pocas ventas si se compara con la cantidad de productos que hay en stcok. Es por ello que urge la necesidad de hacer esta categoría más visible al público. O bien quitar del mercado, al menos, la mitad de los productos en stock.

- Bocinas: no se han obtenido grandes ventas, sin embargo, las reseñas del producto vendido no son malas. Por lo que se recomienda hacer más publicidad y no adquirir más producto hasta que se hayan vendido al menos la mitad de los productos que hay en stock.

- Audífonos: se han tenido pocas ventas, pero todas las ventas tienen excelentes reseñas, salvo el porducto id: 89. Por lo que se recomienda retirar este producto del mercado.

IV. Conclusiones

En general, se observó que no es factible quitar toda una categoría del mercado, pues sólo sería pérdida de ingresos. Lo más conveniente es ver, por categoría, los productos con mejores y peores reseñas, y en caso de que las ventas del producto con mejor reseña superen al de peor reseñas, quitar del mercado el producto con peores reseñas.

Además revisar, las categorías que tengan mucho producto en stock y pocas ventas, a éstas enfatizar la promoción de sus productos a través de publicidad más llamativa y/o descuentos que no generen pérdidas para la empresa.