EMTECH
Emerging Technologies Institute

Santander

LEARNING PATH

DATA SCIENCE CON PYTHON

PROYECTO-01

ANALISIS DE DATOS DE LIFESTORE

GRUPO 1

PYTHONISTAS

BECARIO:

GÓMEZ GAMBOA JHONATAN ALEXANDER

TUTOR:

ALONSO SÁNCHEZ JAIME SAÚL

PROYECTO 01

Gómez Gamboa Jhonatan Alexander

13/9/2021

Índice

Introducción

En el presente trabajo se plasmaron las metodologías que permitieron hacer el análisis de los datos proporcionados por la empresa Lifestore, siguiendo los planteamientos proporcionados en el documento del proyecto. se pondra especial atención a la creación del código para la obtención de la información necesaria, la implementación de una interfaz amigable e intuitiva para navegar por la información, los obstáculos encontrados y como se superaron a la hora de realizar el trabajo dentro de los lineamientos del mismo.

La interpretación de la información recolectada,sera presentada en un formato amigable y de fácil entendimiento, así como las herramientas visuales utilizadas para dar una explicación más acertada para la parte del reporte, pero no formando parte del código principal.

El planteamiento del código se presenta en cuatro grandes grupos, el primero de ellos es un bloque de código utilizado para obtener información importante de la base de datos como podría ser las categorías de los productos, los años en que se tienen registro de las compras, así como los meses de cada una. El segundo gran grupo fue utilizado para resolver el primer apartado, utilizando lo que me gusta llamar "listas complementarias". El tercer bloque de código tiene como finalidad responder a las preguntas del segundo enunciado, de reseñas y devoluciones. El cuarto gran bloque de código, tiene la finalidad de dar respuesta a las incógnitas y preguntas planteadas en el aparado tres como lo son los ingresos promedio mensuales, ingresos totales al año entre otros. El cuarto bloque de código se utilizó para crear una interfaz amigable con el usuario que permite consultar la información solicitada en el trabajo, pero de una forma más clara, más intuitiva y acertada.

Una vez el haber obtenido la información importante en forma de listas se procedió a realizar un análisis para poder contestar a los enunciados del proyecto, así como dar una explicación de comportamiento de los productos, así como soluciones para problemáticas que dificulten una correcta implementación de la empresa para la venta de productos electronicos y digitales.

Entender a los clientes y en base a eso poder anticiparse a su respuesta, con el objetivo de reducir gastos y maximizar ganancias. Se hará uso de herramientas visuales y de análisis exploratorio muy poderosas, como lo son el clustering, mapas de calor, y método del codo para encontrar los agrupamientos adecuados. Por ultimo daré mi conclusión sobre la información obtenida, así como las propuestas que se pueden hacer para tomar mejores decisiones que permitan a la empresa optimizar sus proceso de ventas con buenas estrategias así como buen conocimiento de los compartimentos de los clientes con ciertos artículos

Definición del codigo

En esta parte se define el código utilizado partiendo de los cuatro bloques de código que se definieron. Comenzando con el bloque que nos permite obtener información relevante del conjunto de datos, y que será el punto de partida. Para pode organizar y catalogar los datos necesitaremos conocer los años en que se realizaron las compras, las categorías de los productos. Una vez obtenidos ya podremos comenzar a estructurar nuestro código.

La explicación más a detalle de los procesos y de cómo se definieron las se encuentran plasmadas en el código en forma de comentarios.

GitHub : [https://github.com/AlexandeGomez/PROYECTO-01-DataScience-con-python.git]

obteniendo información del dataset

```
# Algunas aclaraciones importantes :
# La metodologia utilizada para abordar toda la obtención de la información es
# utilizar 2 listas que se complementen entre si, la lista 1 por ejemplo
# tendra etiquetas tipo strings como elementos los cuales representaran a la
# información de la lista 2
# ejemplo:
#        lista1 = ["Grupo1", "Grupo2"]
#      lista 2 = [[1,2,3,4],[5,6,7,8]]
# si utilizamos a la lista 1 para localizar la información asociada al grupo 1
# utilizando su posición podremos saber donde se encuentra su información en la
# otra lista. POSICÓN Ó INDICE palabras clave para el abordaje y entendimiento
# del codigo
#
# WARNING:
# estas listas agrupadas una vez creadas tendran un orden que debe conservarse
# solo los elementos de la jerarquia mas baja pueden ser aordenados y cambiados
# de posición

#                                  [     ]
# ---------------------------------[  1  ]---------------------------------------
#                                  [     ]
# Se obtuvo una lista con los años de las compras
# lifestore_sales = [id-compra,id-producto,score,date,refound]
# date nos es de interes, y tiene la fomra siguiente "dd/mm/aaaa"
# date esta en el indice [3] y tomando esa cadena el año esta en [6:10]
# ciclo for para iterar por todos los elemento.
#
# estructuras : in (nos permite conocer si un elemento, objeto se encuentra en
# una colección salida True/False)
#               not : permite negar una expresión logica

años = []
for i in range(len(lifestore_sales)):
    if lifestore_sales[i][3][6:10] not in años:
      años.append(lifestore_sales[i][3][6:10])
aos=años[:]
#                                  [     ]
# ---------------------------------[  2  ]---------------------------------------
#                                  [     ]
```

```
# De con el mismo planteamiento anterior obtendremos las categorias del conjunto
# lifestore_products = [id-producto, nombre,precio,categoria, stock]
# salida esperada :
#categorias=['procesadores','tarjetas de video','tarjetas madre','discos duros',
#'memorias usb','pantallas','bocinas','audifonos']

categorias = []
for i in range(len(lifestore_products)):
    if lifestore_products[i][3] not in categorias:
      categorias.append(lifestore_products[i][3])

#                                  [     ]
# ---------------------------------[  3  ]---------------------------------------
#                                  [     ]
# Usamos la sentencia siguiente para obtener en una lista los meses que hay en
#un año, con la finalidad de utilizar la metodologia de "listas complementarias"
# termino propio.
# salida esperada : ["01","02",..,"12"]
meses = []
for i in range(1,13):
    if len(str(i))==1:
        meses.append("0"+ str(i))
    else:
        meses.append(str(i))
```

```
[1] "2020" "2019" "2002"
```

```
 [1] "01" "02" "03" "04" "05" "06" "07" "08" "09" "10" "11" "12"
```

```
[1] "procesadores"      "tarjetas de video" "tarjetas madre"
[4] "discos duros"      "memorias usb"      "pantallas"
[7] "bocinas"           "audifonos"
```

lo que se muestra a continuación es la salida del segundo bloque del código para poder responder el primer planteamiento del problema el cual es obtener un concentrado de los productos con mayores ventas y los productos con mayores búsquedas.

Enunciado 1

```
# La siguiente linea de codigo tienen como objetivo contar el numero de ventas
# por articulo cuardando la información en una lista con el siguiente formato
#             [id, numero de ventas, refound(1 ó 0)]
# refound tiene dos valores posibles 1 = True (si hubó un reembolso)
#                                    0 = False (no hubó reembolso)
conteo_ventas_totales = []

for p in lifestore_products:
    cont=0
    for s in lifestore_sales:
        if p[0]==s[1] and s[4]==0:
```

```
            cont+=1
    conteo_ventas_totales.append([p[0],cont,s[4]])

# Las siguientes lineas de codigo tienen como objetivo contar y guardar el
# numero de busquedas y su id
#    [id, numero-busquedas]
conteo_busquedas_totales = []

for p in lifestore_products:
    cont=0
    for s in lifestore_searches:
        if p[0]==s[1]:
            cont+=1
    conteo_busquedas_totales.append([p[0],cont])

#                                  [     ]
# ---------------------------------[  4  ]-------------------------------------
#                                  [     ]
# La siguientes lineas de codigo tienen como objetivo el tomar una lista de
# de elemenentos y ordenarla usando el metodo de burbuja
# a = es un parametro para en el caso de quere ordenar listas como elementos
# cuyo caracteristica de ordenamiento se encuentra en el indice "a"
# un ciclo while permite aplicar el ordenamiento repetidas veces hasta que
# se cumple una condión.
#
# La forma de ordenar : se compara un elemento en la posición i-esima con un
# elemento inmediatamente siguiente i+1-esima.
#
# condición : si el elemento en la posición i-esimo es menor que el siguiente
# elemento, si se cumple la condición se hara un intercambio entre elementos
# esto provocara que se ordenen de mayor a menor los elementos
#
# la condición para detener el bucle while es cuando comparan elementos pares
# contiguos.
#
# Se cumple que "los elementos estan ordenados si al iterar entre elementos
# pares de la colección, y sumando 1 a un contador cada que se cumple una
# desigualdad, si al iterar por toda la colección el contador es igual a
# la longitud de la colección menos 1 la coleccion esta hecha"
# Se ordena la colección que guarda las ventas totales
# ------------------------------------------------------------------------ordenamiento
a = 1
condicion=True
while condicion:
    for i in range(len(conteo_ventas_totales)-1):
        if conteo_ventas_totales[i][a]<conteo_ventas_totales[i+1][a]:
            conteo_ventas_totales[i],conteo_ventas_totales[i+1] =\
            conteo_ventas_totales[i+1],conteo_ventas_totales[i]
    cont=0
    for j in range(len(conteo_ventas_totales)-1):
        if conteo_ventas_totales[j][a]>=conteo_ventas_totales[j+1][a]:
            cont+=1
    if cont>=len(conteo_ventas_totales)-1:
```

```
        condicion=False
# ------------------------------------------------------------------ordenamiento

# Se aplica la misma estructura vista en el bloque B[4] consultarlo para mas
# información
# se ordena la lista que contiene las busquedas totales de mayores busquedas a
#menores
# utilizando "listas complementarias"
# ------------------------------------------------------------------ordenamiento
a = 1

condicion=True
while condicion:
    for i in range(len(conteo_busquedas_totales)-1):
        if conteo_busquedas_totales[i][a]<conteo_busquedas_totales[i+1][a]:
            conteo_busquedas_totales[i],conteo_busquedas_totales[i+1] =\
            conteo_busquedas_totales[i+1],conteo_busquedas_totales[i]
    cont=0
    for j in range(len(conteo_busquedas_totales)-1):
        if conteo_busquedas_totales[j][a]>=conteo_busquedas_totales[j+1][a]:
            cont+=1
    if cont>=len(conteo_busquedas_totales)-1:
        condicion=False
# ------------------------------------------------------------------ordenamiento

# variable que contiene las listas de 15 mejores ventas y 20 mejores busquedas
# obteniendo una rebana de 15 elementos y una de 20 elementos
conteo_mejores_ventas = conteo_ventas_totales[:15]
conteo_mejores_busquedas = conteo_busquedas_totales[:20]

#----------- Termina 1.1 : listas con mayores ventas y busquedas----------------


# se crea una nueva lista, la cual sera una colección de las ventas totales de
# lifestore, dividida en años y cada año en subcategoria
#
# clasificacion_ventas_ac = [[2020],                          [2019],[2002]]
#               ¨[[procesadores, tarjetas video,...,bocinas],[...], [...]]
#    [[[id,año,categoria,refound],[id,año,categoria,refound],[...]],[...],...]
#
# para poder colocar un elemento en cada categoria debe de cumplir una condición
# ya sea que la venta sea la misma que el año en el tiempo presente del codigo
# y que el producto asociado a la compra pertenezca a la categoria de productos
# utilizando "listas complementarias"
clasificacion_ventas_ac = []
for año in años:
    ls_c = []
    for categoria in categorias:
        ls = []
        for p in lifestore_products:
            for s in lifestore_sales:
                if s[1]==p[0] and año==s[3][6:10] and categoria==p[3]\
                and s[4]==0:
```

```
                    ls.append([s[1],s[3],p[3], s[4]])
        ls_c.append(ls)
    clasificacion_ventas_ac.append(ls_c)

# Se crea una nueva lista, coleccion de las busquedas totales separadas por
# categorias de productos.
# clasificacion_busquedas_c = [procesadores, tarjeras madre,...,bocinas]
#       [[id,categoria],[id,categoria],...],[[id,categoria],[id,categoria],...]
# utilizando "listas complementarias"
# para colocarse en cada lista deben de cumplir las condiciones de que la
# busqueda asociada a ese producto pertenece a la categoria correcta
clasificacion_busquedas_c = []
for categoria in categorias:
    ls = []
    for p in lifestore_products:
        for s in lifestore_searches:
            if s[1]==p[0] and categoria==p[3]:
                ls.append([s[1],p[3]])
    clasificacion_busquedas_c.append(ls)

# Se crearа una lista que contendra la cantidad de ventas asociada a cada
# producto de cada categoria de cada año
conteo_ventas_ac = []

for año in años:
    lc=[]
    for categoria in categorias:
        l = []
        for p in lifestore_products:
            cont=0

#                                 [     ]
# --------------------------------[  5  ]---------------------------------------
#                                 [     ]
# Las siguientes lineas de codigo son muy importantes, ya que permite utilizar
# las "listas complementarias" ya que permite acceder a la posición de la etiqueta
# la cual esta asociada a los datos de esa etiquetda dentro de la jerarquia
# el bloque de codigo recibe una lista (contiene todas las etiquetas) y recibe
# una etiqueta para poder buscarla en la lista 1 asociada y obtener su posición
# se podria decir que es un analogo al metodo lista.index(elemento) que devulve
# su posición, solo que esta forma fue hecha con las estructuras basicas.

# nos devuelve la posicón de lo que el elemento año se encuentre en la lista
# años, a demas de ser una forma mas intuitiva.
# si el elemento existe dentro de lista devolvera la posición, de lo contrario
# devolvera un -1
#-------------------------------------------------------------------------------index
            if len(años)==0:
                indx_año = -1
            else:
                for x in range(len(años)):
                    if años[x]==año:
                        indx_año = x
```

```
                        break
                    else:
                        indx_año = -1
#-------------------------------------------------------------------------------index
# misma estuctura aplicada en B[5] pero con la lista categorias
            if len(categorias)==0:
                indx_cat = -1
            else:
                for x in range(len(categorias)):
                    if categorias[x]==categoria:
                        indx_cat = x
                        break
                    else:
                        indx_cat = -1
#-------------------------------------------------------------------------------index
            for v in clasificacion_ventas_ac[indx_año][indx_cat]:
                if p[0]==v[0] and p[3]==categoria:
                    cont+=1
            if p[3]==categoria:
                l.append([p[0],año,categoria,cont])
        lc.append(l)
    conteo_ventas_ac.append(lc)

# Se creara una lista que contendra la ide del producto asociada a su numero de
#busquedas utilizando un contador que cada que encuentre en elemento a contar
#en la lista de todas las busquedas contara cuantaas veces aparece
conteo_busquedas_c = []

for categoria in categorias:
    l = []
    for p in lifestore_products:
        cont=0
# lineas de codigo para encontrar posiciones consultar bloque B[5] para mas info
#-------------------------------------------------------------------------------index
        if len(categorias)==0:
            indx_cat = -1
        else:
            for x in range(len(categorias)):
                if categorias[x]==categoria:
                    indx_cat = x
                    break
                else:
                    indx_cat = -1
#-------------------------------------------------------------------------------index
        for b in clasificacion_busquedas_c[indx_cat]:
            if p[0]==b[0] and p[3]==categoria:
                cont+=1
        if p[3]==categoria:
            l.append([p[0],categoria,cont])
    conteo_busquedas_c.append(l)


# Se utiliza el bloque de ordenamiento definido y explicado en B[4]
```

```
# ordenara las compras de menor a mayor, con la sentencia de ordenamiento de B[4]
# basta con modificar los dos operadores relacionales en la estructa e invertirlos
# para cambiar el orde.
#
# Se itera utilizando como referencia la lista de años y categorias para poder
# acceder a los elementos de la lista que contiene la ventas divididas por
# año y categorias
for año in años:
    for categoria in categorias:
#-------------------------------------------------------------------------------index
        if len(años)==0:
            indx_año = -1
        else:
            for x in range(len(años)):
                if años[x]==año:
                    indx_año = x
                    break
                else:
                    indx_año = -1
#-------------------------------------------------------------------------------index
        if len(categorias)==0:
            indx_cat = -1
        else:
            for x in range(len(categorias)):
                if categorias[x]==categoria:
                    indx_cat = x
                    break
                else:
                    indx_cat = -1
#-------------------------------------------------------------------------------index
# como se mencionó en el B[4] el valor de "a" representa la posición del elemento
# dentro de la listas a ordenras.
# ejemplo
#        lista1=["pez", M, 8] lista2=["araña", H, 4]
# si colocamos a = 2 entonces las listas se ordenaran tomando unicamente este
# criterio

#-----------------------------------------------------------------------ordenamiento
        a = 3
        condicion=True
        while condicion:
            for i in range(len(conteo_ventas_ac[indx_año][indx_cat])-1):
                if conteo_ventas_ac[indx_año][indx_cat][i][a]>\
                conteo_ventas_ac[indx_año][indx_cat][i+1][a]:
                    conteo_ventas_ac[indx_año][indx_cat][i],\
                    conteo_ventas_ac[indx_año][indx_cat][i+1] =\
                    conteo_ventas_ac[indx_año][indx_cat][i+1],\
                    conteo_ventas_ac[indx_año][indx_cat][i]
            cont=0
            for j in range(len(conteo_ventas_ac[indx_año][indx_cat])-1):
                if conteo_ventas_ac[indx_año][indx_cat][j][a]<=\
                conteo_ventas_ac[indx_año][indx_cat][j+1][a]:
                    cont+=1
```

```
            if cont>=len(conteo_ventas_ac[indx_año][indx_cat])-1:
                condicion=False
#-------------------------------------------------------------------------ordenamiento

# ordenamiento de la lista de conteo de bisquedas agrupadas por categorias
# consultar el inicio del codigo para saber como funcionan las
# "listas complementarias"
for categoria in categorias:

# se utiliza la estructura explicada en B[5]
#-----------------------------------------------------------------------------index
    if len(categorias)==0:
        indx_cat = -1
    else:
        for x in range(len(categorias)):
            if categorias[x]==categoria:
                indx_cat = x
                break
            else:
                indx_cat = -1
#-----------------------------------------------------------------------------index
# de menor a mayor, utilizando la estuctura mostada en B[4]

#-------------------------------------------------------------------ordenamiento
    a = 2
    condicion=True
    while condicion:
        for i in range(len(conteo_busquedas_c[indx_cat])-1):
            if conteo_busquedas_c[indx_cat][i][a]>\
            conteo_busquedas_c[indx_cat][i+1][a]:
                conteo_busquedas_c[indx_cat][i],\
                conteo_busquedas_c[indx_cat][i+1] =\
                conteo_busquedas_c[indx_cat][i+1],\
                conteo_busquedas_c[indx_cat][i]
        cont=0
        for j in range(len(conteo_busquedas_c[indx_cat])-1):
            if conteo_busquedas_c[indx_cat][j][a]<=\
            conteo_busquedas_c[indx_cat][j+1][a]:
                cont+=1
        if cont>=len(conteo_busquedas_c[indx_cat])-1:
            condicion=False
#-------------------------------------------------------------------ordenamiento

# ahora tenemos las ventas y busquedas ordenadas y catalogadas
```

La salida es la siguiente cuando solicitamos la lista de los 15 productos con mayores ventas:

```
    id  ventas                                            nombres
0   54      49     SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm
1    3      42  Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, S...
2    5      20  Procesador Intel Core i3-9100F, S-1151, 3.60GH...
3   42      18  Tarjeta Madre ASRock Micro ATX B450M Steel Leg...
4   57      15  SSD Adata Ultimate SU800, 256GB, SATA III, 2.5...
```

```
5    4      13  Procesador AMD Ryzen 3 3200G con Gráficos Rade...
6   29      13  Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GA...
7    2      12  Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 3...
8   47      11        SSD XPG SX8200 Pro, 256GB, PCI Express, M.2
9   12       9  Tarjeta de Video ASUS NVIDIA GeForce GTX 1660 ...
10  48       9  SSD Kingston A2000 NVMe, 1TB, PCI Express 3.0, M2
11   7       7  Procesador Intel Core i7-9700K, S-1151, 3.60GH...
12  44       6  Tarjeta Madre MSI ATX B450 TOMAHAWK MAX, S-AM4...
13  18       5  Tarjeta de Video Gigabyte NVIDIA GeForce GT 10...
14   8       4  Procesador Intel Core i5-9600K, S-1151, 3.70GH...
```

La salida es la siguiente cuando solicitamos la lista de los 20 productos con mayores busquedas:

```
    id  busquedas                                          nombre
0   54        263     SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm
1   57        107  SSD Adata Ultimate SU800, 256GB, SATA III, 2.5...
2   29         60  Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GA...
3    3         55  Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, S...
4    4         41  Procesador AMD Ryzen 3 3200G con Gráficos Rade...
5   85         35  Logitech Audífonos Gamer G635 7.1, Alámbrico, ...
6   67         32  TV Monitor LED 24TL520S-PU 24, HD, Widescreen,...
7    7         31  Procesador Intel Core i7-9700K, S-1151, 3.60GH...
8    5         30  Procesador Intel Core i3-9100F, S-1151, 3.60GH...
9   47         30        SSD XPG SX8200 Pro, 256GB, PCI Express, M.2
10  48         27  SSD Kingston A2000 NVMe, 1TB, PCI Express 3.0, M2
11  44         25  Tarjeta Madre MSI ATX B450 TOMAHAWK MAX, S-AM4...
12   2         24  Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 3...
13  42         23  Tarjeta Madre ASRock Micro ATX B450M Steel Leg...
14   8         20  Procesador Intel Core i5-9600K, S-1151, 3.70GH...
15  12         15  Tarjeta de Video ASUS NVIDIA GeForce GTX 1660 ...
16  21         15  Tarjeta de Video MSI AMD Mech Radeon RX 5500 X...
17  66         15  TCL Smart TV LED 55S425 54.6, 4K Ultra HD, Wid...
18  18         11  Tarjeta de Video Gigabyte NVIDIA GeForce GT 10...
19  51         11         SSD Kingston UV500, 480GB, SATA III, mSATA
```

Mostraremos a continuación la salida de los 5 productos con peores ventas por categorias, tomando como muestra los productos de la categoria de procesadores de las compras realizadas en 2020.

```
  id     categoria ventas                                    nombres
0  9  procesadores      0  Procesador Intel Core i3-8100, S-1151, 3.60GHz...
1  1  procesadores      2  Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Q...
2  6  procesadores      3  Procesador Intel Core i9-9900K, S-1151, 3.60GH...
3  8  procesadores      4  Procesador Intel Core i5-9600K, S-1151, 3.70GH...
4  7  procesadores      7  Procesador Intel Core i7-9700K, S-1151, 3.60GH...
```

Mostraremos a continuación la salida de los 20 productos con menos busquedas por categorias, tomando como muestra los productos de la categoria de procesadores.

```
  id     categoria busquedas                                      nombre
0  9  procesadores         1  Procesador Intel Core i3-8100, S-1151, 3.60GHz...
1  1  procesadores        10  Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Q...
2  6  procesadores        10  Procesador Intel Core i9-9900K, S-1151, 3.60GH...
```

```
3  8  procesadores        20  Procesador Intel Core i5-9600K, S-1151, 3.70GH...
4  2  procesadores        24  Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 3...
5  5  procesadores        30  Procesador Intel Core i3-9100F, S-1151, 3.60GH...
6  7  procesadores        31  Procesador Intel Core i7-9700K, S-1151, 3.60GH...
7  4  procesadores        41  Procesador AMD Ryzen 3 3200G con Gráficos Rade...
8  3  procesadores        55  Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, S...
```

Enunciado 2

```
# Se crea una lista que funciona como concentrado para las reseñas y devoluciones
# en este caso estaremos utilizando la siguiente estructura para guardar la
# infomación  : [id,[r,r,r,..],[d,d,d,...]] una lista cuyo primer elemento sera el id
# del prodiucto, el segundo elemento sera una lista que contendra todas las reseñas
#asociadas a ese producto, el tercer elemento sera una lista que contendra la
# el valor logico si esa compra asociada se dio un reembolso
# a esa estructura la llamaremos :
# "unidad de guardado" = [id,[r,r,r,..],[d,d,d,...]] = unit
# concentrado_reseñas_devoluciones_a tendra la siguiente forma:
#
# concentrado_reseñas_devoluciones_a = ["año1",año2,...,añon]
#                           [[unit1,unit2,unit3,...],[],...,[]]

concentrado_reseñas_devoluciones_a = []
# itera por el conjunto de años y de las ventas
for año in años:
    ls=[]
    for p in lifestore_products:
        ls.append([p[0],[],[]])
        for s in lifestore_sales:
            if p[0]==s[1] and año==s[3][6:10]:
                ls[len(ls)-1][1].append(s[2])
                ls[len(ls)-1][2].append(s[4])
# pasa una situación y es que si un producto no fue comprado, no tendra las dos
# valoraciones que estamo buscando y ese articlo su "unidad de guardad" estara
# vacia ejemplo : [ide,[],[]], para nuestro analisis necesitaremos que al menos
# este un cero, haciendo referencia que el no fue comprado.
# por eso la condición siguiente : [id,[r,r,r,..],[d,d,d,...]] si la lista en el
#indice 1 esta vacia y la del indice 2 tambien esta vacia, osea longitud cero
# agregar un cero en su interior, claro esto de ultimo de haber interado por la
# lista completa, si nos aseguramos de que en efecto el articulo no fue comprado
        if len(ls[len(ls)-1][1])==0:
            ls[len(ls)-1][1].append(0)

        if len(ls[len(ls)-1][2])==0:
            ls[len(ls)-1][2].append(0)

    concentrado_reseñas_devoluciones_a.append(ls)


# se crea concentrado que tiene los promedios de las reseñas y las devoluciones
# Las siguientes lineas de codigo tienen la finalidad de utilizar la lista creada
# anteriormenete llamada "concentrado_reseñas_devoluciones_a" y calcular la media
```

```
# tanto de las reseñas como de las devoluciones y guardarlas en una nueva lista
# que tendra la siguiente forma y estara dividad en años:
# - - - [año1,año2,...,añon]
# - - [[id1, promedio_reseñas1, promedio_reembolsos1],[[id2, promedio_reseñas2,
# ----------------------------------promedio_reembolsos2]],["año2"],...,[año3]]

promedio_reseñas_devoluciones_a = []

for año in años:
    ls=[]
# se utilizan las siguiente lineas para encontar el indice del elemento año en
# la lista años, puede consultar mas de la estructura en B[5]
#------------------------------------------------------------------------------- index
    if len(años)==0:
        indx_año = -1
    else:
        for x in range(len(años)):
            if años[x]==año:
                indx_año = x
                break
            else:
                indx_año = -1
#------------------------------------------------------------------------------- index
# condirendo  [id,[r,r,r,..],[d,d,d,...]] como elementos de guardado, se itera
# en la lista en el indice 1, y se suman sus elementos para posteriormente
# dividirlos entre el numero de elementos, calculando el promedio de reseñas
# para ese articulo y colocando en otra lista, lo mismo se hace para calcular
# el promedio de devoluciones de cada articulo
    for rd in concentrado_reseñas_devoluciones_a[indx_año]:
        prom_score = 0
        prom_devol = 0

        for j in rd[1]:
            prom_score+=j
        for k in rd[2]:
            prom_devol+=k

        prom_score/=len(rd[1])
        prom_devol/=len(rd[2])
        ls.append([rd[0],prom_score,prom_devol])
    promedio_reseñas_devoluciones_a.append(ls)
# tenemos una nueva lista que contiene los promedios de reseñas y devoluciones de
# cada articulo, categorizando estas metricas por el año en que se hizo la
# valoración y la devoluvion

# Ordenamiento de los productos con mayores reseñas promedio
# iteramos por los elementos de la lista años ["2020","2019","2002"]
for año in años:
# la siguiente linea se usa para obtener los indices de los años, para poder
# acceder a su respectiva información en las listas que ya estan divididas
# por años, mas imformación de esta estructura consultar B[5]
#------------------------------------------------------------------------------- index
    if len(años)==0:
```

```
        indx_año = -1
    else:
        for x in range(len(años)):
            if años[x]==año:
                indx_año = x
                break
            else:
                indx_año = -1
#------------------------------------------------------------------------------ index

# utilizamos el bloque que nos permite ordenas los elementos de una lista
# utilizando el ordenamiento burbuja  para mas información checar B[4]
# el valro a sirve para decir en que posición tomar el valor par el
# ordenamiento
#------------------------------------------------------------------------------ordenamiento
    a = 1
    condicion=True
    while condicion:
        for i in range(len(promedio_reseñas_devoluciones_a[indx_año])-1):
            if promedio_reseñas_devoluciones_a[indx_año][i][a]<\
            promedio_reseñas_devoluciones_a[indx_año][i+1][a]:
                promedio_reseñas_devoluciones_a[indx_año][i],\
                promedio_reseñas_devoluciones_a[indx_año][i+1] =\
                promedio_reseñas_devoluciones_a[indx_año][i+1],\
                promedio_reseñas_devoluciones_a[indx_año][i]
        cont=0
        for j in range(len(promedio_reseñas_devoluciones_a[indx_año])-1):
            if promedio_reseñas_devoluciones_a[indx_año][j][a]>=\
            promedio_reseñas_devoluciones_a[indx_año][j+1][a]:
                cont+=1
        if cont>=len(promedio_reseñas_devoluciones_a[indx_año])-1:
            condicion=False
#------------------------------------------------------------------------------ordenamiento

# ahora que ya obtuvimos le promedio de las reseñas y las devoluciones,
# y la lista esta ordenada, podremos eliminar aquellos productos que no
# se compraron ni fueron evaluados, sabremos cuales son ya que su promedio
# de reseñas sera igual a cero, obtenemos una lista que encuentre en que
# indices se encuentran dichos productos para posteriormente eliminarlos


# la siguiente lista muestra los INDICES que eliminar
indices_to_remove = []

for año in años:
    ls=[]
# lo mismo, para poder iterar por la lista necesitamos acceder los datos que estan
# organizados por años, por ende utilizamos las siguientes lineas para conocer la
# posición del elemento año un la lista años, y saber donde se encuentra su
# información asociada
# para mas información consultar B[5]
#------------------------------------------------------------------------------ index
    if len(años)==0:
```

```
        indx_año = -1
    else:
        for x in range(len(años)):
            if años[x]==año:
                indx_año = x
                break
            else:
                indx_año = -1
#------------------------------------------------------------------------------ index
# se guarda el indice, que es la posición del elemento cuyo reseña promedio es 0
    for i in range(len(promedio_reseñas_devoluciones_a[indx_año])):
        if promedio_reseñas_devoluciones_a[indx_año][i][1]==0:
            ls.append(i)
    indices_to_remove.append(ls)
# ahora contamos con una lista con los indices de los productos que debemos eliminar

# iteramos por la lista que contiene los años
for año in años:
# para mas información del bloque siguiente consultar el B[5]
# la finalida es poder iterar con años y encontar posiciones, es mas intuitivo
# de esta manera
#------------------------------------------------------------------------------ index
    if len(años)==0:
        indx_año = -1
    else:
        for x in range(len(años)):
            if años[x]==año:
                indx_año = x
                break
            else:
                indx_año = -1
#------------------------------------------------------------------------------ index
# utilizamos la palabra reservada "del" que nos permite eliminar los indices
# de una lista de nuestro interes
# utilizamos la siguiente forma para colcoar los indices :
# - - - indices_to_remove[indx_año][0] nos pemite conocer desde que indice debemos
#comenzar a eliminar ":" que se podria interpretar como hasta y
# - - - indices_to_remove[indx_año][-1]+1] que se podria entender como
# hasta el ultimo elemento mas 1, ya que las rebanadas no cuentan el ultimo
    del promedio_reseñas_devoluciones_a[indx_año]\
    [indices_to_remove[indx_año][0]:indices_to_remove[indx_año][-1]+1]


# Obteniendo listas de 10 productos con mejores reseñas y 20 con las peores
# En las siguientes lineas vamos a sacar unicamente un numero contado de
# productos, con las mayores reseñas y con las peores
promedio_n_max_score = []
promedio_n_min_score = []
# n es un valor que al modificarle le decimos al programa cuantos elementos
# queremos extraer para guardarlos en las variables anteriores
n = 10

# iteramos por el conjunto de los años
```

```
for año in años:
    ls_max = []
    ls_min = []
# utilizamos la siguiente información para encontrar los indices asociados al
# elemento año de la lista año, necesaria para acceder a la información de ese
# año en la lista de registro
# para mas información consultar B[5]
#------------------------------------------------------------------------------ index
    if len(años)==0:
        indx_año = -1
    else:
        for x in range(len(años)):
            if años[x]==año:
                indx_año = x
                break
            else:
                indx_año = -1
#------------------------------------------------------------------------------ index

# aqui hay una confición que quiere decir que en el caso de que la lista no cuente
# con los suficientes n-elementos solicitados, que tenga menos elementos que los n
# solicitados, entonces que use la longitud de la lista completa
    if len(promedio_reseñas_devoluciones_a[indx_año])>=n:
        for i in range(n):
            ls_max.append(promedio_reseñas_devoluciones_a[indx_año][i])
            ls_min.append(promedio_reseñas_devoluciones_a[indx_año][-(i+1)])
    else:
        for j in range(len(promedio_reseñas_devoluciones_a[indx_año])):
            ls_max.append(promedio_reseñas_devoluciones_a[indx_año][j])
            ls_min.append(promedio_reseñas_devoluciones_a[indx_año][j])
# se guardan en las variables adecuadas
    promedio_n_max_score.append(ls_max)
    promedio_n_min_score.append(ls_min)

# ahora ya contamos con dos listas quec contienen los 20 productos con
# o peores reseñas
```

Se mostrara la salida del enunciado para los 10 productos con mejores reseñas de las comprars hechas en el año 2020

```
     id  reseñ  devol                                          nombres
0   1.0    5.0    0.0  Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Q...
1   6.0    5.0    0.0  Procesador Intel Core i9-9900K, S-1151, 3.60GH...
2   7.0    5.0    0.0  Procesador Intel Core i7-9700K, S-1151, 3.60GH...
3   8.0    5.0    0.0  Procesador Intel Core i5-9600K, S-1151, 3.70GH...
4  11.0    5.0    0.0  Tarjeta de Video ASUS AMD Radeon RX 570, 4GB 2...
5  21.0    5.0    0.0  Tarjeta de Video MSI AMD Mech Radeon RX 5500 X...
6  22.0    5.0    0.0  Tarjeta de Video MSI NVIDIA GeForce GTX 1050 T...
7  25.0    5.0    0.0  Tarjeta de Video Sapphire AMD Pulse Radeon RX ...
8  28.0    5.0    0.0  Tarjeta de Video Zotac NVIDIA GeForce GTX 1660...
9  40.0    5.0    0.0  Tarjeta Madre Gigabyte XL-ATX TRX40 Designare,...
```

Se mostrara la salida del enunciado para los 10 productos con menores reseñas en las comrpas hechas en el año 2020

```
     id    reseñ    devol                                              nombres
0  45.0  1.00000  1.00000  Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151...
1  17.0  1.00000  1.00000  Tarjeta de Video Gigabyte AMD Radeon R7 370 OC...
2  31.0  1.83333  0.50000  Tarjeta Madre AORUS micro ATX B450 AORUS M (re...
3  46.0  2.00000  1.00000  Tarjeta Madre Gigabyte micro ATX GA-H110M-DS2,...
4  89.0  3.00000  0.00000  Cougar Audífonos Gamer Phontum Essential, Alám...
5  94.0  4.00000  0.00000  HyperX Audífonos Gamer Cloud Flight para PC/PS...
6  13.0  4.00000  0.00000  Tarjeta de Video Asus NVIDIA GeForce GTX 1050 ...
7  10.0  4.00000  0.00000  MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PC...
8  29.0  4.14286  0.07143  Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GA...
9   2.0  4.33333  0.00000  Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 3...
```

Enunciado 3

```python
# Se creara una lista que permitira organizar las ventas en años > meses
# siedo esta la jerarquia, para poder acceder a la informaión
# deberemos de pasarle como argumento un año y un mes en forma de indices
ventas_registro_am = []
# tendra la sigiente forma
# ventas_registro_am = [año1,año2,...,añon]
#            [[01,02,...,12],[01,02,..,12],[01,02,...,03]]
#      [[[id1,venta,01],[id2,venta,01],...],[[id1,venta,02],[...]],[[...]]]

# se utiliza un ciclo for para iterar por los elementos de la lista
# años. y tambien uno para iterar por los elementos de la lista
# meses.

# la variable lm : contendra una lista vacia la cual sera unicamente
# utilizada en este bloque, ya que sera guardada en una listqa de
# de mayor jerarquia, "lm" va a guardar las listas que cada una
# representa un año diferente

# lavariable l : tiene un objetivo similar a "lm", con la diferencia
# de que esta tendra en su interior, las lista de cada compra hecha,
# en ese mes y ese año, y se guardara en la lista lm, donde ahim cada
# elemento represetantara un mes diferente.
for año in años:
    lm=[]
    for mes in meses:
        l=[]
        for s in lifestore_sales:
# para que el elemento pueda guardarse en la lista l, debe de cumplir
# las condiciones de que el año de la venta y el mes deben de coincidir
# asi como que solo se contaran ventas que no se hayan reembolsado
            if s[3][6:10]==año and  s[3][3:5]==mes and s[4]==0:
                l.append([s[1],año,mes,s[4]])
        lm.append(l)
    ventas_registro_am.append(lm)

# En las siguientes lineas nos tomamtes a la tarea de tomar la lista
# anterioriormente creada, para adicionar al registro de las ventas, el
#precio del articulos
```

```
# se itera sobre el conjunto de los años, y de los meses para poder
#acceder a ventas_registro_am
for año in años:
    for mes in meses:

# las lineas siguiente se utilizan para localizar el indice que
# año como elemento ocupa en la lista años, esto con la intención
# de iterar sobre ventas_registro_am
# para mas información consultar el B[5]
#-------------------------------------------------------------------------index
        if len(años)==0:
            indx_año = -1
        else:
            for x in range(len(años)):
                if años[x]==año:
                    indx_año = x
                    break
                else:
                    indx_año = -1
#-------------------------------------------------------------------------index
# aqui estamos utilizando el bloque es explicado en B[5]
# para poder saber el indice que ocupa la variable mes en la lista meses
        if len(meses)==0:
            indx_mes = -1
        else:
            for x in range(len(meses)):
                if meses[x]==mes:
                    indx_mes = x
                    break
                else:
                    indx_mes = -1
#-------------------------------------------------------------------------index
# Se itera sobre el registro de las ventas ordenados por año
# y mes, para poder tomar de referencia el id, y saber que
# precio adicionar.
        for v in ventas_registro_am[indx_año][indx_mes]:
            for p in lifestore_products:
                if p[0]==v[0]:
                    v.append(p[2])




# Las siguientes lineas tienen como objetivo el calcular las ventas
# y los ingresos promedio mensuales, clasificados en cada año
# y guardarlos en una nueva lista
#
# ventas_ingresos_promedio_mesuales_a = [año1,año2,...,año3]
#         año1 = [[año1,ip,vp],[año1,ip,vp],...,[]]

ventas_ingresos_promedio_mensuales_a = []

# se itera sobre el conjunto de años
```

```
for año in años:
    ip=0
    vp=0
    for mes in meses:
# lineas de codigo para consultar el indice de que ocupa un elemento
# en una colección, mas información consultar B[5]
#-------------------------------------------------------------------------------index
        if len(años)==0:
            indx_año = -1
        else:
            for x in range(len(años)):
                if años[x]==año:
                    indx_año = x
                    break
                else:
                    indx_año = -1
#-------------------------------------------------------------------------------index
# lineas de codigo para consultar el indice de que ocupa un elemento
# en una colección, mas información consultar B[5]
        if len(meses)==0:
            indx_mes = -1
        else:
            for x in range(len(meses)):
                if meses[x]==mes:
                    indx_mes = x
                    break
                else:
                    indx_mes = -1
#-------------------------------------------------------------------------------index
# el ciclo for itera sobre los elementos de la lista
# ventas_registro_am, y mediante una variable que guarda un valor
# despues le suma otro, llamada ip llevara la suma de los ingresos
# asociado a la compra de cada articulo para cada mes
# este valor se le divide entre 12, ya que asi sabremos que las
# ganancias de un año entre 12 seran las ganancias promedio por
# por mes lo mismo hacemos con las ventas, que en este caso se
# obtienen utilizando la función "len()" para saber los elementos
# (ventas) en cada colección
        for v in ventas_registro_am[indx_año][indx_mes]:
            ip+=v[4]
        vp+=len(ventas_registro_am[indx_año][indx_mes])
    ip/=12
    vp/=12
    ventas_ingresos_promedio_mensuales_a.append([año,ip,vp])


# las siguientes lineas de codigo tienen la finalidad de obtener
# un concentrado de las ventas e ingresos totales realizados
# cada mes, en cada año
#
# ventas_ingresos_am =    [año1  ,año2 ,..., añon]
#         [ [ [01],[02],...,[12] ],[ [01],[02],...,[12] ],[[01],[02],...,[12] ] ]
#             [01]=[id, ingresos_totales, ventas_totales]
```

```
ventas_ingresos_am = []

# se itera sobre el conjunto de años y sobre el conjunto de meses
# para poder acceder a al concentrado de ingresos y ventas
# ordenados por año y mes : ventas_registro_am
# asi obtener las ventas totales por mes y no promedio
for año in años:
    lm=[]
    for mes in meses:
# se utilizan las lineas siguientes para poder encontrar el indice
# que ocupa el elemento año en la lista años
# para mas información consultar B[5]
#-------------------------------------------------------------------------index
        if len(años)==0:
            indx_año = -1
        else:
            for x in range(len(años)):
                if años[x]==año:
                    indx_año = x
                    break
                else:
                    indx_año = -1
#-------------------------------------------------------------------------index
# se utilizan las lineas siguientes para poder encontrar los indices
# necesarios para acceder a la información, para mas información
# consultar B[5]
        if len(meses)==0:
            indx_mes = -1
        else:
            for x in range(len(meses)):
                if meses[x]==mes:
                    indx_mes = x
                    break
                else:
                    indx_mes = -1
#-------------------------------------------------------------------------index
# it es un contador, para llevar el sumario de los ingresos
        it=0
# se itera sobre las ventas del registros
        for i in ventas_registro_am[indx_año][indx_mes]:
            it+=i[4]
# una vez se han sumado todas las ganancias del mes
# se guardan en una lista lm, la cual contendra estas
# metricas , para despues de juntarlas colocar la nueva lista
# en ventas_ingreso, cada elementos correspondera a un año
        vt=len(ventas_registro_am[indx_año][indx_mes])
        lm.append([año,mes,it,vt])
    ventas_ingresos_am.append(lm)


# se crea un nuevo consentrado el cual clasificara unicamente
# las ventas por año en que se hicieron
total_ingresos_ventas_a = []
```

```
# total_ingresos_ventas_a = [año1 , año2 , ... , añon]
# [[año1,total_ganancias,total_ventas],[año2,total_ganancias,total_ventas],[...]]

# se itera con los elementos de la lista de años y de meses
for año in años:
    pi_año=0
    pv_año=0
    for mes in meses:
# las siguientes lineas son para poder encontrar el indice que
# ocupa el elemento año en la lista años
# para mas información consultar B[5]
#-------------------------------------------------------------------------------index
        if len(años)==0:
            indx_año = -1
        else:
            for x in range(len(años)):
                if años[x]==año:
                    indx_año = x
                    break
                else:
                    indx_año = -1
#-------------------------------------------------------------------------------index
# las siguientes lineas son para poder encontrar el indice que
# ocupa el elemento mes en la lista meses
# para mas información consultar B[5]
        if len(meses)==0:
            indx_mes = -1
        else:
            for x in range(len(meses)):
                if meses[x]==mes:
                    indx_mes = x
                    break
                else:
                    indx_mes = -1
#-------------------------------------------------------------------------------index
# se itera sobre la lista que lleva el registro de ventas dividida o
# clasificada en años y meses
        for v in ventas_registro_am[indx_año][indx_mes]:
# pi_año solo se utiliza para sumar y guardar la cantidad de las
# ganancias y pv_año suma y mantiene las ventas
            pi_año+=v[4]
        pv_año+=len(ventas_registro_am[indx_año][indx_mes])
# lo guardan en la lista total_ingresos_ventas_a
    total_ingresos_ventas_a.append([año,pi_año,pv_año])
# ahora contamos con una lista que nos permite conocer cuantos
# ingresos se obtuvieron en un año en especifico asi como
# sus ventas totales


# Creamos una nueva lista, ya que a nuestro criterio sera
# necesaria para analisis posterior de la información
# la cual nos permite registrar las ventas obtenidas por
# la tienda pero divididas en años>categorias>meses
```

```
ventas_registro_acm = []

# se itera sobre los elemenos de la lista años, categorias y
#meses
for año in años:
    lc=[]
    for categoria in categorias:
        lm=[]
        for mes in meses:
# en las siguientes lineas obtendremos los indices asociadas
# a los elementos dentro de sus respectivas listas
# para poder crear dichas claficaciones
#-------------------------------------------------------------------------index
            if len(años)==0:
                indx_año = -1
            else:
                for x in range(len(años)):
                    if años[x]==año:
                        indx_año = x
                        break
                    else:
                        indx_año = -1
#-------------------------------------------------------------------------index
            if len(categorias)==0:
                indx_cat = -1
            else:
                for x in range(len(categorias)):
                    if categorias[x]==categoria:
                        indx_cat = x
                        break
                    else:
                        indx_mes = -1
#-------------------------------------------------------------------------index
            if len(meses)==0:
                indx_mes = -1
            else:
                for x in range(len(meses)):
                    if meses[x]==mes:
                        indx_mes = x
                        break
                    else:
                        indx_mes = -1
# para mas información consultar B[5]
#-----------------------------------------------------------index
# ls es una lista vacia al inicio pero que contendra listas
# donde cada lista tiene la información de cada venta ya
# filtrada
            l=[]
# se itera sobre los elementos de la lista de las ventas
            for s in lifestore_sales:
# se itera sobre los elementos de la lista de los productos
                for p in lifestore_products:
# si se cunmple la condicion que de ambos elementos coinciden en id
```

```
# y el la lista de ventas es del año actual de la iteración for
# sobre años, ademas de que el id esa sociado a la categoria presente
# del ciclo for sobre categorias, lo mismo pasa con los meses y que
# las ventan no tengan rembolsos.
# si se cumplen las condiciones el elemento forma parte de la
# clasificación
                    if s[1]==p[0] and s[3][6:10]==año and \
                    p[3]==categoria and s[3][3:5]==mes and s[4]==0:
                        l.append([s[1],año,categoria,mes,s[4],p[2]])
# lm comienza como una lista vacia pero guardara las listas que contienen
# las ventas de ese mes
            lm.append(l)
# lc comienza como una lista vacia pero guardara las listas que contienen
# las ventas del mes actual del ciclo for sobre meses, del año actual del
# ciclo for sobre años
        lc.append(lm)
# la lista que contiene las ventas, del mes y la categoria correctas del
# ciclo for presente
    ventas_registro_acm.append(lc)


# a continuacion crearmos una lista llamada ventas_ingresos_acm
# la cual tendra los ingresos totales y las ventas totales de
# cada mes para cada año

ventas_ingresos_acm = []
# se itera sobre los elementos de años, categorias y meses
for año in años:
    lc=[]
    for categoria in categorias:
        lm=[]
        for mes in meses:
# necesario para poder acceder a la lista ventas_registro_acm
# para mas información consultar B[5]
#-------------------------------------------------------------------------index
            if len(años)==0:
                indx_año = -1
            else:
                for x in range(len(años)):
                    if años[x]==año:
                        indx_año = x
                        break
                    else:
                        indx_año = -1
#-------------------------------------------------------------------------index
            if len(categorias)==0:
                indx_cat = -1
            else:
                for x in range(len(categorias)):
                    if categorias[x]==categoria:
                        indx_cat = x
                        break
                    else:
```

```
                        indx_mes = -1
#-------------------------------------------------------------------------index
            if len(meses)==0:
                indx_mes = -1
            else:
                for x in range(len(meses)):
                    if meses[x]==mes:
                        indx_mes = x
                        break
                    else:
                        indx_mes = -1
#-------------------------------------------------------------------------index
# it es un contador que los permitira sumar las ingresos de
# las compras hechas ese año, de esa categoria de producto
# en un mes especifico
            it=0
            for v in ventas_registro_acm[indx_año][indx_cat][indx_mes]:
# se suman las ganancas en it que es : ingresos totales
                it+=v[5]
# lm comienza como una lista vacia pero se usa para guar la
# informacion de las ganancias, donde cada elemento corresponde
# a un mes diferente
            lm.append([año,categoria,mes,it])
# lc es una lista que permite guardar la lista lm, en lc donde
# cada elemento representa a cada cateria
        lc.append(lm)
# finalamente lc se guarda en ventas_ingresos_acm donde
# cada elemento representa a la información de cada
#año
    ventas_ingresos_acm.append(lc)
# ahora tenemos una lista que contiene las ventas totales de cada mes
# para cada categoria de cada año, con el fin de contar con estos
# datos para el analisis posterior
```

Se mostrara la salida que solicita el promedio de ingresos y ventas mensuales

```
      ingreso promedio mensuales  ventas promedio mensuales
2020                61471.416667                  22.750000
2019                    0.000000                   0.000000
2002                   21.583333                   0.083333
```

Se mostrara la salida que solicita los ingrsos totales por años

```
      ingreso total anual  venta total anual
2020               737657                273
2019                    0                  0
2002                  259                  1
```

Se mostrara la salida que muestra los ingresos y ventas por meses, se mostrara solo del año 2020

```
     año mes  total ingresos  total ventas
0   2020  01          117738            52
```

1	2020	02	107270	40
2	2020	03	162931	49
3	2020	04	191066	74
4	2020	05	91677	33
5	2020	06	36949	11
6	2020	07	26949	11
7	2020	08	3077	3
8	2020	09	0	0
9	2020	10	0	0
10	2020	11	0	0
11	2020	12	0	0

la función de ser una interfaz grafica mas amigable e intuitiva para poder navegar por toda la información que se recolecto y organizo en el analisis.

Interfaz de usuario

```
# USUARIO : Emtech2021
# CONTRASEÑA : BecasEmtech2021

# Estas variables fueron definidas para darle una interfaz mas
# amigable, para poder nevegar por la información
Inicio = """
Interfaz : teclear el numero de la entrada deseada
1: Iniciar sesión  2:Crear usuario
-----------------------------------------------------
#                               #                 #
#1) Ingresar Usuario/Contraseña  #2) Crear Usuario #
#                               #                 #
-----------------------------------------------------
"""
Crear1 = """
-----------------------------------------------------
# 1) Ingresar Nombre completo:                  [*]
-----------------------------------------------------
# 2) Ocupación dentro de la institución:        [ ]
-----------------------------------------------------
# 3) IDE de empleado:                           [ ]
-----------------------------------------------------
# 4) Correo electronico:                        [ ]
-----------------------------------------------------
# 5) Ingresar contraseñar de 6-15 caracteres    [ ]
-----------------------------------------------------
# 6) Repetir la contraseña de 6 a 15 caracteres [ ]
-----------------------------------------------------
"""
Crear2 = """
-----------------------------------------------------
# 1) Ingresar Nombre completo:                  [ ]
-----------------------------------------------------
# 2) Ocupación dentro de la institución:        [*]
-----------------------------------------------------
```

```
# 3) IDE de empleado:                          [ ]
-------------------------------------------------
# 4) Correo electronico:                       [ ]
-------------------------------------------------
# 5) Ingresar contraseñar de 6-15 caracteres   [ ]
-------------------------------------------------
# 6) Repetir la contraseña de 6 a 15 caracteres [ ]
-------------------------------------------------
"""
Crear3 = """
-------------------------------------------------
# 1) Ingresar Nombre completo:                 [ ]
-------------------------------------------------
# 2) Ocupación dentro de la institución:       [ ]
-------------------------------------------------
# 3) IDE de empleado:                          [*]
-------------------------------------------------
# 4) Correo electronico:                       [ ]
-------------------------------------------------
# 5) Ingresar contraseñar de 6-15 caracteres   [ ]
-------------------------------------------------
# 6) Repetir la contraseña de 6 a 15 caracteres [ ]
-------------------------------------------------
"""
Crear4 = """
-------------------------------------------------
# 1) Ingresar Nombre completo:                 [ ]
-------------------------------------------------
# 2) Ocupación dentro de la institución:       [ ]
-------------------------------------------------
# 3) IDE de empleado:                          [ ]
-------------------------------------------------
# 4) Correo electronico:                       [*]
-------------------------------------------------
# 5) Ingresar contraseñar de 6-15 caracteres   [ ]
-------------------------------------------------
# 6) Repetir la contraseña de 6 a 15 caracteres [ ]
-------------------------------------------------
"""
Crear5 = """
-------------------------------------------------
# 1) Ingresar Nombre completo:                 [ ]
-------------------------------------------------
# 2) Ocupación dentro de la institución:       [ ]
-------------------------------------------------
# 3) IDE de empleado:                          [ ]
-------------------------------------------------
# 4) Correo electronico:                       [ ]
-------------------------------------------------
# 5) Ingresar contraseñar de 6-15 caracteres   [*]
-------------------------------------------------
# 6) Repetir la contraseña de 6 a 15 caracteres [ ]
-------------------------------------------------
```

```
"""
Crear6 = """
-----------------------------------------------------
# 1) Ingresar Nombre completo:                  [ ]
-----------------------------------------------------
# 2) Ocupación dentro de la institución:        [ ]
-----------------------------------------------------
# 3) IDE de empleado:                           [ ]
-----------------------------------------------------
# 4) Correo electronico:                        [ ]
-----------------------------------------------------
# 5) Ingresar contraseñar de 6-15 caracteres    [ ]
-----------------------------------------------------
# 6) Repetir la contraseña de 6 a 15 caracteres [*]
-----------------------------------------------------
"""
Ingresar1 = """
Si se registró, su IDE es el usuario para ingresar
-----------------------------------------------------
# - Ingresar Usuario-Administrador:             [*]
-----------------------------------------------------
# - Ingrese Contraseña:                         [ ]
-----------------------------------------------------
"""
Ingresar2 = """
Si se registró, su IDE es el usuario para ingresar
-----------------------------------------------------
# - Ingresar Usuario-Administrador:             [ ]
-----------------------------------------------------
# - Ingrese Contraseña:                         [*]
-----------------------------------------------------
"""

Lobby = """

Interfaz : teclear el numero de la entrada deseada
para consultar cada apartado (1,2,3,4,5)
-----------------------------------------------------
#                                                   |
# 1)Productos mas vendidos y rezagados              |
#                                                   |
----------------------------------------------------|
#                                                   |
# 2)Productos por reseñas en el servicio            |
#                                                   |
----------------------------------------------------|
#                                                   |
# 3)Total de ingresos y ventas promedio             |
#                                                   |
----------------------------------------------------|
#                                                   |
# 4)Reporte Completo                                |
#                                                   |
```

```
-------------------------------------------------|
#                                                |
# 5)Salir                                        |
#                                                |
-------------------------------------------------
"""
Confirmacion = """
-------------------------------------------------
#                #                               #
#1) Confirmar    #  2) Salir y volver a intentar  #
#                #                               #
-------------------------------------------------

"""
# La lista vacia llamda Info : es para almacenar los datos
# de la persona que se ha dado de alta al sistema
# la integridad de la información es la prioridad
# por lo tanto debemos de registrar quien tiene acceso

# Usuarios : es una lista que contienen los usuarios
# predefinidos para logearse con el programa

# Contraseñas : es una lista que contiene la
# contraseña predefinida para logearse

# Condicion_loby : es una variable que solo puede
# tomar 2 valores [0 ó 1], 0 para que empecemos
# desde la pagina del login, 1 para saltarnos el login
# y mantener la sesión sin cerrar

Info = []
Usuarios = ["Emtech2021"]
Contraseñas = ["BecasEmtech2021"]
condicion_lobby = 0

# todo el codigo de la intefaz se desarrolla en un ciclo
# while, para asgurarnos siempre mantenernos en la interfaz

while True:
# aqui se manda a consola la escena del login, para poder
# inicar sesión o crear otro usuario
    print(Inicio)

# la condicion_lobby se aplica, si es igual a 0 ejecutara
# el codigo, si condicion_lobby == 1 se saltara varias
# escenas pera que no te solicite el usuario y
# contraseñas de nuevo
    if condicion_lobby==0:
# pide al usuario que ingrese una entrada [1 o 2] para
# guardarlo en la variable menu_p1
        menu_p1 = int(input("1-2"))

# Se evaluar menu_p1 si es igual a 1 pasamos al apartado
```

```
# de logearse con el usuario predeterminado, en el caso
# de no cumplirse la condición pasaremos al apartado
# para registrar un nuevo usuario
    if menu_p1 == 1:

# de nuevo se evalua la condicion_lobby de ser igual a 0
# imprimira en pantalla la escena de apoyo para inciar
# sesion y pedira al usuario que introduzca una entrada
# para guardarle en usuario, que sera el usuario que
# intenta iniciar sesion
        if condicion_lobby==0:
            print(Ingresar1)
            usuario = input("Introducir usuario: ")

# Se evaluar una condición donde se busca si el elemento
# usuario existe en la lista Usuarios, en el caso de haber
# coindicencia se imprime la pantalla que te ayuda y
# solicita la contraseña

# en el caso de no cumplirse la condición del if, se
# imprimira en pantalla un mensaje que el usuario es
# incorrecto y se termina la iteración
        if usuario in Usuarios:
            if condicion_lobby==0:
                print(Ingresar2)
# pide al usuario que escriba la contraseña y la guarda en
# una variable
                contraseña = input("Introducir contraseña: ")

# Si la contraseña introducida existe en la lista que contiene
# a las contraseñas validas entonces permite continuar
# de lo contrario imprime en pantalla un aviso informando
# que las contraseñas no coinciden

# imprimire la variable lobby que contiene un apoyo grafico que
# ayuda a la navegación
# pide al usuario una entrada entre [1-5]
            if contraseña in Contraseñas:
                print(Lobby)
                consulta = int(input("Ingrese 1-5"))
                print(" ")
# si la entrada es igual a 1 se ejecuta el bloque de codigo
                if consulta==1:
#-------------------------------------------------------------------#
#-------------------------------------------------------------------#
# hay un detalle con la codificón y es que si no se tiene cuidado
# podemos tener lineas muy largas de codigo, en este contexto
# s permite ahorrarnos 2 espacios de caracteres
                    s = "|"
# separador visual, se usa la muitiplicación de caracter y numero
# entero, y se presemta un anuncio de lo que vendra
                    print(15*" "+22*"> "+22*"< ")
                    print("                    A continuación se \
```

```
presentan los 15 productos con mayores ventas registradas")
                    print(" ")
# permite colocar barras horizontales, dichos calculo ya se hicieron
# de cuantas barras son lo ideal
                    print(6*"-"+s+8*"-"+s+14*"-"+s+17*"-"+s+70*"-")
# se crearon estas variables para la parte de la cabecera ya es
# espaciados y en formato
                    ordn = "|Orden|"
                    ide = 2*" "+"Ide"+3*" "+"|"
                    nv = 4*" "+"ventas"+4*" "+"|"
                    cat = 4*" "+"Categoria"+4*" "+"|"
                    no = 6*" "+"Nombre"
                    print(ordn+ide+nv+cat+no)
                    print(6*"-"+s+8*"-"+s+14*"-"+s+17*"-"+s+70*"-")
# contador se usa para saber cuantas entradas se han mostrado
                    cont=1
# se itera sonbre la colección "conteio_mejores_ventas"
# tambien se itera sobre el conjunto de productos para usar
# sus caracteristicas, so ambos id concuerdan entonces
# se imprime la información relevate a ese articulo
                    for v in conteo_mejores_ventas:
                        for p in lifestore_products:
                            if p[0]==v[0]:
                                c = str(cont)
                                sp1 = str(v[0])
                                sp2 = str(v[1])
                                sp3 = p[3]
                                sp4 = p[1]

# dentro del codigo se implementaron algo que denominé
# "espacios dinamicos" y es hacer que dos valores de diferentes
# longitudes no arruien el formato espacio ya que es adaptativa
# formula : string + (cantidad maxima - longitud(str))*" "
# mantendra todo perfectamente en formato
                                c = 1*" "+"#"+c+(2-len(c))*" "+2*" "
                                sp1 = 3*" "+sp1+(2-len(sp1))*" "+3*" "
                                sp2 = 5*" "+sp2+(3-len(sp2))*" "+6*" "

                                sp3 = sp3+(17-len(sp3))*" "

# se imprime el producto uno por uno, hasta abarcar la lista de conteo
                                print(c+s+sp1+s+sp2+s+sp3+s+sp4)
                                print(" ")
                                cont+=1
# visualmente otro separador
# se imprime lo que se va a mostrar
                    print(15*" "+22*"> "+22*"< ")
                    print("                     A continucación se\
 presentan los 20 productos con mayores busquedas registradas")
                    print(" ")
# emcabezado de la tabla respectivamente espaciado
                    ordn = "|Orden|"
                    ide = "  Ide   |"
```

```
                    nb = "No. busquedas |"
                    cat = 4*" "+"Categoria"+4*" "+"|"
                    no = "      Nombre"
# se concatena todo junto, al igual que los delimatores
                    print(ordn+ide+nb+cat+no)
                    print(6*"-"+s+8*"-"+s+14*"-"+s+17*"-"+s+70*"-")
                    cont=1
# itera sobre el conjunto que tiene las mejores busquedas y
# de los de los productos.

# Si las id coinciden entonces se procede a mostrar la
# información del producto, apoyandose la lista de
# productos

# c,sp1,sp2,sp3,sp4 permite tratar los datos a mostrar
# y guardarlo como cadenas para poder contatenarlos y
# hacer uso de los espacios dinamicos
                    for b in conteo_mejores_busquedas:
                        for p in lifestore_products:
                            if p[0]==b[0]:
                                c = str(cont)
                                sp1 = str(b[0])
                                sp2 = str(b[1])
                                sp3 = p[3]
                                sp4 = p[1]

                                c = 1*" "+"#"+c+(2-len(c))*" "+2*" "
                                sp1 = 3*" "+sp1+(2-len(sp1))*" "+3*" "
                                sp2 = 5*" "+sp2+(3-len(sp2))*" "+6*" "
                                sp3 = sp3+(17-len(sp3))*" "
# se concatena toda la salida y la muestra por pantalla
                                print(c+s+sp1+s+sp2+s+sp3+s+sp4)
                                print(" ")
                                cont+=1

# se itera sobre los años y las categorias para poder acceder
# a la lista llamada conteo_ventas_ac
                    for año in años:
                        for categoria in categorias:
                            print(" ")
                            print(60*"__")
                            print("                              Listas de \
productos por categorias con menores ventas")
                            print("Se presenta una lista los productos m\
enos vendidos del año: "+año+" en la categoria de: "+categoria)
                            print(60*"__")
# permite colocar un emcabezado de forma bonita y ordenada
# y se imprime la salida ya ordenada
                            print(7*"-"+s+5*"-"+s+11*"-"+s+5*"-"+s+19*"-"+s+70*"-")
                            ordn = 1*" "+"Orden"+1*" "+"|"
                            ide = 1*" "+"Ide"+1*" "+"|"
                            nv = 3*" "+"ventas"+2*" "+"|"
                            añ = 1*" "+"año"+1*" "+"|"
```

```
                            ca = 5*" "+"Categoria"+5*" "+"|"
                            no = 6*" "+"Nombre"
                            print(ordn+ide+nv+añ+ca+no)
                            print(7*"-"+s+5*"-"+s+11*"-"+s+5*"-"+s+19*"-"+s+70*"-")

# las siguientes lineas tienen la finalidad de encontrar en que indice se
# encuentra un elemento en una lista, con la finalidad de iterar en los
# datos asociados a esa lista. Para mas información consultar B[5]
#-----------------------------------------------------------------index
                            if len(años)==0:
                                indx_año=-1
                            else:
                                for x in range(len(años)):
                                    if años[x]==año:
                                        indx_año = x
                                        break
                                    else:
                                        indx_año=-1

                            if len(categorias)==0:
                                indx_cat=-1
                            else:
                                for x in range(len(categorias)):
                                    if categorias[x]==categoria:
                                        indx_cat = x
                                        break
                                    else:
                                        indx_cat =-1
#-----------------------------------------------------------------index
# "cont" es una variable que nos ayuda a numerar en que orden van saliendo
# las salidas
#se itera por los datos de la lista que contienen el numero de ventas
# cuyo orden en el que esta dispueso los datos es: [id,año,categoria,venta]
#
# las variables, c,sp1,sp2,sp3,sp4 se usan para guardar en forma de strings
# los datos que se mostraran en consola con la finalidad de no hacer muy
# largas las lineas de codigo
                            cont=1
                            for v in conteo_ventas_ac[indx_año][indx_cat][:5]:
                                for p in lifestore_products:
                                    if v[0]==p[0]:
                                        c = str(cont)
                                        sp1 = str(v[0]) #ide
                                        sp2 = v[1] #año
                                        sp3 = v[2] #categoria
                                        sp4 = str(v[3]) #ventas
                                        sp5 = p[1] #nombre

                                        c = 2*" "+"#"+c+" "+2*" "
                                        sp1 = 2*" "+sp1+(2-len(sp1))*" "+1*" "
                                        sp3 = 1*" "+sp3+(17-len(sp3))*" "+1*" "
                                        sp4 = 4*" "+sp4+(2-len(sp4))*" "+5*" "
                                        sp2 = sp2+1*" "
```

```
                                    print(c+s+sp1+s+sp4+s+sp2+s+sp3+s+sp5)
                                    print(" ")
                                    cont+=1

# pasamos al segundo enuncia que nos pide mostrar los productos con menos ventas
# y menos busquedas por categoria
# iteramos por los elementos de la lista categoria
# y se crean las variables que contendran el emcabezado de la tabla, para
# despues contantenarse y mostrarse en consola de una menera estilizada
                    for categoria in categorias:
                        print(" ")
                        print(60*"__")
                        print("                              Listas de \
productos por categorias con menores busquedas")
                        print("Se presenta una lista de los productos\
con menos busquedas en la categoria de: "+categoria)
                        print(60*"__")
                        ordn = 1*" "+"Orden"+1*" "+"|"
                        ide = 1*" "+"Id."+1*" "+"|"
                        nb = 1*" "+"Busqueda"+1*" "+"|"
                        ca = 5*" "+"Categoria"+5*" "+"|"
                        no = 6*" "+"Nombre"
                        print(ordn+ide+nb+ca+no)
                        print(7*"-"+s+5*"-"+s+10*"-"+s+19*"-"+s+70*"-")
# bloque para consultar el indice de un elemento en una lista
# para mas información consultar el apartado B[5]
#-----------------------------------------------------------index
                        if len(categorias)==0:
                            indx_c = -1
                        else:
                            for x in range(len(categorias)):
                                if categorias[x]==categoria:
                                    indx_c = x
                                    break
                            else:
                                indx_c = -1
#-----------------------------------------------------------index
# "cont" es una variable que nos ayuda a numerar en que orden van saliendo
# las salidas
#se itera por los datos de la lista que contienen el numero de busquedas
# cuyo orden en el que esta dispueso los datos es: [id,categoria,n_busqueda]
#
# las variables, c,sp1,sp2,sp3,sp4 se usan para guardar en forma de strings
# los datos que se mostraran en consola con la finalidad de no hacer muy
# largas las lineas de codigo
                        cont=1
                        for b in conteo_busquedas_c[indx_c][:20]:      #[id, cat, nb]
                            for p in lifestore_products:
                                if b[0]==p[0]:

                                    c = str(cont) # orden
                                    sp1 = str(b[0]) # id
                                    sp2 = b[1] # categoria
```

```
                                    sp3 = str(b[2]) #numero busquedas
                                    sp4 = p[1] #nombre

                                    c = 2*" "+"#"+c+(2-len(c))*" "+2*" "
                                    sp1 = 2*" "+sp1+(2-len(sp1))*" "+1*" "
                                    sp2 = 1*" "+sp2+(17-len(sp2))*" "+1*" "
                                    sp3 = 4*" "+sp3+(3-len(sp3))*" "+3*" "

                                    print(c+s+sp1+s+sp3+s+sp2+s+sp4)
                                    print(" ")
                                    cont+=1
# aqui se muestra por primera vez  la variable "condicion_lobby"
# el codigo le pregunta al usuario si quiere volver al lobby o
# cerrar sesión, y guarda un estado logico para esta condición
# en el caso de introducir un 0 estaremos diciendo al programa que
# nos solicite las contraseñas de nuevo
                    print(" ")
                    condicion_lobby=int(input("Para voler al lobby \
teclea '1', para cerrar sesión teclear '0' "))

#-------------------------------------------------------------------------------------------------------
#-------------------------------------------------------------------------------------------------------
# La opción dos del menu, la cual nos permite visualizar lo
# requerido en el apartado 2 del proyecto
# si la entrada consulta es igual a 2 se ejecuta el codigo siguiente
#
# En esta primera parte preparmos las variables que nos pemtiran
# construir el encabezado de una menra estilizada, y visulmente
# mas atractiva
# las variables ordn,ide,rp,ca,no permiten no hacer tan ancho
# el codigo y que este pueda entrar en la hoja del reporte
                elif consulta==2:
                    s = "|"
                    print(60*"# ")
                    print(" ")
                    print("Se mostrara una lista de los 20 artic\
ulos con mejores reseñas considerando devoluciones")
                    print(" ")
                    print(60*"# ")
                    for año in años:
                        print(" ")
                        print(60*"__")
                        print("                        Listas de\
 10 productos con mejores reseñas considerando devoluciones")
                        print("              Se presenta una li\
sta de los 10 productos con mayores reseñas en los años: "+año)
                        print(60*"__")
                        ordn = " Orden |"
                        ide = " Ide |"
                        rp = " reseña promedio  |"
                        dp = "Devolucion promedio|"
                        ca = "    Categoria    |"
                        no = "      Nombre"
```

```
                        print(ordn+ide+rp+dp+ca+no)
                        print(7*"-"+s+5*"-"+s+18*"-"+s+19*"-"+s+17*"-"+s+40*"-")
# se hace uso uso del bloque que permite encontrar el indice de un
# elemento en una lista, para mas informacion consultar B[5]
#-------------------------------------------------------------index
                        if len(años)==0:
                            indx_año=-1
                        else:
                            for x in range(len(años)):
                                if años[x]==año:
                                    indx_año = x
                                    break
                                else:
                                    indx_año=-1
#-------------------------------------------------------------index
# "cont" es una variable que nos ayuda a numerar en que orden van saliendo
# las salidas
# se itera por los datos de la lista que contienen las mejores reseñas
# cuyo orden en el que esta dispueso los datos es: [id,reseña promedio,devoluciones p]
#
# las variables, c,sp1,sp2,sp3,sp4,sp5 se usan para guardar en forma de strings
# los datos que se mostraran en consola con la finalidad de no hacer muy
# largas las lineas de codigo
                        cont=1
                        for r in promedio_n_max_score[indx_año]:
                            for p in lifestore_products:
                                if r[0]==p[0]:
                                    c = str(cont)
                                    sp1 = str(r[0])
                                    sp2 = str(r[1])
                                    sp3 = str(r[2])
                                    sp4 = p[1]
                                    sp5 = p[3]

                                    c = 2*" "+"#"+c+(2-len(c))*" "+2*" "
                                    sp1 = 2*" "+sp1+(2-len(sp1))*" "+1*" "
                                    sp2 = sp2+(18-len(sp2))*" "
                                    sp3 = sp3+(18-len(sp3))*" "+1*" "
                                    sp5 = sp5+(17-len(sp5))*" "

                                    print(c+s+sp1+s+sp2+s+sp3+s+sp5+s+sp4)
                                    cont+=1
                                    print(" ")

# pasamos a mostrar los 20 elementos con peores reseñas
# en esta parte colcoamos las variables y mostramos
# una interfaz estilizada para el encabezado
# las variables, ord, ide, rp, dp,ca,no, se usan para
# guardar los encabezados, para despues contatenarlos e imprimirlos
                    print(" ")
                    print(60*"# ")
                    print(" ")
                    print("Se mostrara una lista de los 10 artic\
```

```
ulos con las peores reseñas considerando devoluciones")
                    print(" ")
                    print(60*"# ")
                    for año in años:
                        print(" ")
                        print(60*"__")
                        print("                      Listas de 10\
 productos con las peores reseñas considerando devoluciones")
                        print("            Se presenta una list\
a de los 10 productos con las peores reseñas en los años: "+año)
                        print(60*"__")
                        ordn = " Orden |"
                        ide = " Ide |"
                        rp = " reseña promedio  |"
                        dp = "Devolucion promedio|"
                        ca = "    Categoria    |"
                        no = "      Nombre"
                        print(ordn+ide+rp+dp+ca+no)
                        print(7*"-"+s+5*"-"+s+18*"-"+s+19*"-"+s+17*"-"+s+40*"-")
# hacemos uso del bloque que nos permmite ocnsultar el indice de un elemento
# en una lista, para mas información conultar B[5]
#-----------------------------------------------------------------index
                        if len(años)==0:
                            indx_año=-1
                        else:
                            for x in range(len(años)):
                                if años[x]==año:
                                    indx_año = x
                                    break
                                else:
                                    indx_año=-1
#-----------------------------------------------------------------index
# "cont" es una variable que nos ayuda a numerar en que orden van saliendo
# las salidas
# se itera por los datos de la lista que contienen los productos con las preores
#reseñas
# cuyo orden en el que esta dispueso los datos es: [id,reseña_promed,devoluciones_p]
#
# las variables, c,sp1,sp2,sp3,sp4 se usan para guardar en forma de strings
# los datos que se mostraran en consola con la finalidad de no hacer muy
# largas las lineas de codigo
                        cont=1
                        for r in promedio_n_min_score[indx_año]:
                            for p in lifestore_products:
                                if r[0]==p[0]:

                                    c = str(cont)
                                    sp1 = str(r[0])
                                    sp2 = str(r[1])
                                    sp3 = str(r[2])
                                    sp4 = p[1]
                                    sp5 = p[3]
```

```
                                    c = 2*" "+"#"+c+(2-len(c))*" "+2*" "
                                    sp1 = 2*" "+sp1+(2-len(sp1))*" "+1*" "
                                    sp2 = sp2+(18-len(sp2))*" "
                                    sp3 = sp3+(18-len(sp3))*" "+1*" "
                                    sp5 = sp5+(17-len(sp5))*" "

                                    print(c+s+sp1+s+sp2+s+sp3+s+sp5+s+sp4)
                                    cont+=1
                                    print(" ")
# aqui se termina lo solicitado en el apartado 2, y solicitamos al
# usuario introduzca 1 para continuar en la sesion y 0 para salir
# esto se guarda en la variable condicion_lobby
                    print(" ")
                    condicion_lobby=int(input("Para voler al lobby \
teclea '1', para cerrar sesión teclear '0' "))
#-------------------------------------------------------------------------------------------------------------
#-------------------------------------------------------------------------------------------------------------

# acabamos de empezar las lineas de nos ayudara a presentar la información
# solicitada en el apartado 3
# si la variable consulta es igual a 3 se ejecutara el codigo siguiente.
#
# primero se crearan las variables y ss imprimiran los delimitadores
# necesarios para presentar el encabezado de la tabla de una forma
# mas estilizada.
#
# se iterara por los elementos de la lista años, para poder acceder
# a los datos de la lista que contiene los ingresos y las ventas
#
# las variables ordn, it,vt,añ, nos permiten guardar los encabezados
# para despues mostrarlos por consola
                elif consulta==3:
                    s = "|"
                    print(60*"# ")
                    print(" ")
                    print("Se mostrara una lista del total de ingre\
sos y ventas promedio mensuales por año, asi como ingresos totales \
por año")
                    print(" ")
                    print(60*"# ")

                    for año in años:
                        print(" ")
                        print(45*"__")
                        print("                  Lista de los i\
ngresos y ventas promedio por mes")
                        print("   Se presenta una lista de los \
ingresos y ventas promedio del año: "+año+" por mes: ")
                        print(45*"__")
                        ordn = " Orden |"
                        it = " Ingreso promedio |"
                        vt = "  Ventas promedio  |"
                        añ = " año"
```

```
                        print(ordn+it+vt+añ)
                        print(7*"-"+s+18*"-"+s+19*"-"+s+8*"-")
# las siguientes lineas nos permiten buscar en que indice pertene
# un esta un elementoe una lista, para mas info consultar B[5]
#-------------------------------------------------------------index
                        if len(años)==0:
                            indx_año=-1
                        else:
                            for x in range(len(años)):
                                if años[x]==año:
                                    indx_año = x
                                    break
                                else:
                                    indx_año=-1
#-------------------------------------------------------------index
# se itera por los datos de la lista que contienen los ingresos promedio
# mensuales cuyo orden en el que esta dispueso los datos es:
# [año, promedio mensual, ventas mensuales]
#
# las variables, sp1,sp2,sp3, se usan para guardar en forma de strings
# los datos que se mostraran en consola con la finalidad de no hacer muy
# largas las lineas de codigo y al final se imprimen por pantalla
                        v = ventas_ingresos_promedio_mensuales_a[indx_año][:]
                        sp1 = v[0]
                        sp2 = str(v[1])
                        sp3 = str(v[2])

                        c = 3*" "+"#1"+2*" "
                        sp2 = sp2+(18-len(sp2))*" "
                        sp3 = sp3+(19-len(sp3))*" "

                        print(c+s+sp2+s+sp3+s+sp1)

# ahora nos pide los ingresos totales por cada año
# estas primeras lineas nos permite mostrar de una forma
# estilizada el encabezado de la tabla, con sus espacios
# y delimitadores
#
# se itera por todos los elementos de la lista años
#
# las variables orn,ir,vt,añ, las usamos para guardar
# los encabezados y despues imprimirlos en pantalla
                    print(" ")
                    print(60*"# ")
                    print(" ")
                    print("Se mostrara una lista de los ingresos\
 totales por año")
                    print(" ")
                    print(60*"# ")
                    for año in años:
                        print(" ")
                        print(45*"__")
                        print("                  Lista de los in\
```

```
gresos totales por año")
                        print("  Se presenta una lista de los in\
gresos y ventas totales del año: "+año)
                        print(45*"__")
                        ordn = " Orden |"
                        it = "   Ingreso Total    |"
                        vt = "    Ventas Totales  |"
                        añ = " año"
                        print(ordn,it,vt,añ)
                        print(7*"-"+s+21*"-"+s+21*"-"+s+8*"-")
# bloque de codigo que nos premite buscar en que indice un elmento
# esta en una lista, para mas información consultar B[5]
#-----------------------------------------------------------------index
                        if len(años)==0:
                            indx_año=-1
                        else:
                            for x in range(len(años)):
                                if años[x]==año:
                                    indx_año = x
                                    break
                                else:
                                    indx_año=-1
#-----------------------------------------------------------------index
# se itera por los datos de la lista que contienen el total de ingresos
# de las ventas por año cuyo orden esta dispuesto de la siguiente manera
# : [año,total de ingresos, total de ventas]
#
# las variables sp1,sp2,sp3  se usan para guardar en forma de strings
# los datos que se mostraran en consola con la finalidad de no hacer muy
# largas las lineas de codigo
                        v = total_ingresos_ventas_a[indx_año][:]

                        sp1 = v[0]
                        sp2 = str(v[1])
                        sp3 = str(v[2])

                        c = 3*" "+"#1"+2*" "
                        sp2 = 8*" "+sp2+(6-len(sp2))*" "+7*" "
                        sp3 = 9*" "+sp3+(3-len(sp3))*" "+9*" "

                        print(c+s+sp2+s+sp3+s+sp1)
                        print(" ")

# ahora se mostraran el total de ingresos y ventas ordenados por años
# y meses
# Estas primeras lineas se usan para colcoar el encabezado de la table
# de una menra estilizada, espaciada y ordenada
#
# Se itera sobre los elementos de la lista años y meses
#
# las variables añ,ms,im,vm, se usan para guarlas los encabezados
# en forma de strings para despues contatenarlos y mostrarlos
                    print(60*"# ")
```

```
                    print(" ")
                    print("Se mostrara una lista del total de ingr\
esos y ventas de cada mes para por cada año")
                    print(" ")
                    print(60*"# ")
                    for año in años:
                        print(20*"__")
                        print("ventas de cada mes del año: "+año)
                        print(20*"__")
                        print(8*"-"+s+6*"-"+s+10*"-"+s+6*"-")
                        añ = 3*" "+"año"+2*" "+"|"
                        ms = 1*" "+"mes"+2*" "+"|"
                        im = 1*" "+"ingresos"+1*" "+"|"
                        vm = 1*" "+"ventas"+1*" "+"|"
                        print(añ+ms+im+vm)
                        print(8*"-"+s+6*"-"+s+10*"-"+s+6*"-")
                        for mes in meses:
# El bloque siguiente es par poder conocer el indice de un elemento en
# una lista. Para mas informacion consultar B[5]
#-------------------------------------------------------------------index
                            if len(años)==0:
                                indx_año=-1
                            else:
                                for x in range(len(años)):
                                    if años[x]==año:
                                        indx_año = x
                                        break
                                    else:
                                        indx_año=-1
#-------------------------------------------------------------------index
                            if len(meses)==0:
                                indx_mes=-1
                            else:
                                for x in range(len(meses)):
                                    if meses[x]==mes:
                                        indx_mes = x
                                        break
                                    else:
                                        indx_mes=-1
#-------------------------------------------------------------------index
# se le asigna la variable v una copía de los elemento de la lista
# ventas_ingresos_am cuyos elementos estan dispuestos como :
# [año,mes,ingresos totales, ventas totales]
# se guardan en variable en forma de string para que sea mas sencillo
# mostrarlos por consila, colocando los "espaciados dinamicos"
# que varian conforme la longitud del string
# y la información se orden para mostrarse en pantalla
                            v = ventas_ingresos_am[indx_año][indx_mes]
                            sp_añ = v[0] #año str
                            sp_ms = v[1] # mm str
                            sp_im = str(v[2]) # ingresos str
                            sp_vm = str(v[3]) # ventas str
```

```
                            sp_añ = 2*" "+sp_añ+2*" "
                            sp_ms = 2*" "+sp_ms+2*" "
                            sp_im = 2*" "+sp_im+(8-len(sp_im))*" "
                            sp_vm = sp_vm+(6-len(sp_vm))*" "

                            print(sp_añ+s+sp_ms+s+sp_im+s+sp_vm)
                            print(" ")
# se le solicitara al usuario dar un valor a la variable
# condicion_lobby para salir de la sesion o mantenerse
                    print(" ")
                    condicion_lobby=int(input("Para voler al lobby \
teclea '1', para cerrar sesión teclear '0' "))
#-------------------------------------------------------------------------------------------------------------
#-------------------------------------------------------------------------------------------------------------
                elif consulta==4:
# estamos en el ultimo a partado, toda este bloque de codigo se
# ejecutara su consulta contiene el numero 4
#
# en esta parte el tratarse de un reporte la gran mayoria es
# escrito, cadenas de caracteres o strings multilineas que se
# mostraran en consola, y unas ciclois for para mostrar ciertos
# articulos que pertenecen a un de las 5 categorias encontradas
                    s = "|"
                    print(60*"# ")
                    print("""
Se presentara a conticuación un reporte detallado del analisis de\
 los datos del dataset de lifestore:

Observaciones : En el dataset se encuentran compras hechas en los\
 años 2020, 2019 y 2002, estos ultimos dos años no bridan los
suficientes datos para hacer un analisis por lo cual solo se dara\
n resultados del año 2020.

1) El mes con mayor ventas en 2020 es : Marzo
con una ganancia de $191'066 pesos  y un total de 74 ventas en ese mes

2) Analizando los datos nos encontramos que podemos segmentar los\
datos en 5 tipos diferentes de productos:

    - 1) Productos no conocidos y no atrayentes
    - 2) Productos poco conocidos y poco atrayentes
    - 3) Productos conocidos y poco atrayentes
    - 4) Productos conocidos y atrayentes
    - 5) Productos muy conocidos y atrayentes

A continuacion se presentan los productos que caen en estas categ\
orias y como mejorar las ventas y ganancias promedio:

# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #\
 # # # # # # # # # # # # # #
-------------------------------------------------------------------\
-----------------------------
|      #1:      |Si hay exceso inventario dejar de introducir nue\
```

```
vo hasta agotar el existente|
|   Productos   |hay 2 formas de tratar estos productos:         \
                            |
| no conocidos  | 1) Sacar el producto del catalogo y el inventar\
io sacarlo con alguna estra-|
| no atrayentes |    tegia de ventas ejemplo : rebajas           \
                            |
|               | 2) Iniciar campaña de publicidad y de ventas de\
 alto impacto               |
|---------------|-------------------------------------------------\
----------------------------|
|Caracteristicas| Productos que no dejan ganancia ya sea por el p\
oco interes de la los com-  |
|               | pradores o su desconocimiento de nuestro catalo\
go de productos.           |
 ---------------------------------------------------------------\
----------------------------
| No | id. |    categoria     | nombre                         \
                            |
 ---------------------------------------------------------------\
----------------------------""")
# cont nos ayuda a llevar una enumeracion de las salidas
# una vez encontrados los elementos que deben de mostrarse para
# este apartado, iteramos sobre ellos y mostramos solo aquellos
# que coincidan con el id solicitado.
#
# se colocaran en string para que sea facil poder mostrarlos en
# pantalla, hay que recordar que lifestore_products tiene sus elementos
# [id, nombre, ,precio, categoria]
                    cont=1
                    for p in lifestore_products:
                        for i in [9,10,13,14,15,16,17,19,20,23,24,\
                                  26,27,30,32,34,35,36,37,38,39,41,\
                                  43,45,46,31,53,55,56,58,59,60,61,\
                                  62,63,64,65,68,69,70,71,72,73,75,\
                                  76,77,78,79,80,81,82,83,86,87,88,\
                                  90,91,92,93,95,96]:
                            if p[0]==i:
                                c = str(cont)
                                sp1 = str(p[0])
                                sp2 = p[3]
                                sp3 = p[1]

                                c="|"+1*" "+"#"+c+(2-len(c))*" "
                                sp1 = 1*" "+sp1+(2-len(sp1))*" "+2*" "
                                sp2 = 1*" "+sp2+(17-len(sp2))*" "

                                print(c+s+sp1+s+sp2+s+sp3)
                                print(" ")
                                cont+=1
# se muestran los grupos en pantalla
                    print("""
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #\
```

```
 # # # # # # # # # # # # # #
-------------------------------------------------------------------\
---------------------------
|      #2       |Si hay exceso inventario dejar de introducir nue\
vo hasta agotar el existente|
|   Productos   |                                                  \
                            |
| poco conocidos|1) Iniciar campaña de publicidad y de ventas de \
alcance e impacto alto      |
|poco atrayentes|                                                  \
                            |
|---------------|-------------------------------------------------\
---------------------------|
|Caracteristicas| Productos que cerecen de interes por parte de l\
os compradores, no conocen  |
|               | de su existencia y no se sienten animados a adq\
uirirlos                    |
 -------------------------------------------------------------------\
---------------------------
| No | id. |    categoria     | nombre                           \
                            |
 -------------------------------------------------------------------\
----------------------------"""
# cont nos ayuda a llevar una enumeracion de las salidas
# una vez encontrados los elementos que deben de mostrarse para
# este apartado, iteramos sobre ellos y mostramos solo aquellos
# que coincidan con el id solicitado.
#
# se colocaran en string para que sea facil poder mostrarlos en
# pantalla, hay que recordar que lifestore_products tiene sus elementos
# [id, nombre, ,precio, categoria]
                    cont=1
                    for p in lifestore_products:
                        for i in [1,6,8,11,12,18,21,22,25,28,33,40\
                                  ,47,48,49,50,51,52,74,84,89,94]:
                            if p[0]==i:
                                c = str(cont)
                                sp1 = str(p[0])
                                sp2 = p[3]
                                sp3 = p[1]

                                c="|"+1*" "+"#"+c+(2-len(c))*" "
                                sp1 = 1*" "+sp1+(2-len(sp1))*" "+2*" "
                                sp2 = 1*" "+sp2+(17-len(sp2))*" "

                                print(c+s+sp1+s+sp2+s+sp3)
                                print(" ")
                                cont+=1
# se muestra el tercer grupo en pantalla
                    print("""
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #\
 # # # # # # # # # # # # # #
|---------------|-------------------------------------------------\
```

```
----------------------------|
|      #3       |Si hay exceso inventario, reducir el existente h\
asta tener adecuado         |
|   Productos   |                                                \
                            |
|   conocidos   |1) Iniciar campaña de publicidad de alcance mode\
rado y campaña de ventas    |
|poco atrayentes|    de alcance e impacto alto                    \
                            |
|---------------|-------------------------------------------------\
----------------------------|
|Caracteristicas| Productos que gozan de cierta notoriedad pero q\
ue no inspira lo suficiente |
|               | a los compradores para adquirilos               \
                            |
 ---------------------------------------------------------------\
----------------------------
| No | id. |    categoria     | nombre                          \
                            |
 ---------------------------------------------------------------\
----------------------------""")
# cont nos ayuda a llevar una enumeracion de las salidas
# una vez encontrados los elementos que deben de mostrarse para
# este apartado, iteramos sobre ellos y mostramos solo aquellos
# que coincidan con el id solicitado.
#
# se colocaran en string para que sea facil poder mostrarlos en
# pantalla, hay que recordar que lifestore_products tiene sus elementos
# [id, nombre, ,precio, categoria]
                    cont=1
                    for p in lifestore_products:
                        for i in [2,4,5,7,29,42,66,67,85]:
                            if p[0]==i:
                                c = str(cont)
                                sp1 = str(p[0])
                                sp2 = p[3]
                                sp3 = p[1]

                                c="|"+1*" "+"#"+c+(2-len(c))*" "
                                sp1 = 1*" "+sp1+(2-len(sp1))*" "+2*" "
                                sp2 = 1*" "+sp2+(17-len(sp2))*" "

                                print(c+s+sp1+s+sp2+s+sp3)
                                print(" ")
                                cont+=1
# se muestra el cuarto grupo y descripcion en pantalla
                    print("""
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #\
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
 ---------------------------------------------------------------\
----------------------------
|      #4       |Mantener el inventario completo en epocas de may\
ores ventas                 |
```

```
|   Productos   |                                                \
                          |
|   conocidos   | 1) Campañas de publicidad y campaña de ventas d\
e alcance moderado        |
|  atrayentes   |                                                \
                          |
|---------------|------------------------------------------------\
----------------------------
|Caracteristicas| Productos que tienen notoriedad dentro del cata\
logo de la empresa, generan |
|               | atención que se traduce en un ingreso de capita\
l                         |
 ---------------------------------------------------------------\
----------------------------|
| No | id. |    categoria     | nombre                         \
                          |
 ---------------------------------------------------------------\
----------------------------""")
# cont nos ayuda a llevar una enumeracion de las salidas
# una vez encontrados los elementos que deben de mostrarse para
# este apartado, iteramos sobre ellos y mostramos solo aquellos
# que coincidan con el id solicitado.
#
# se colocaran en string para que sea facil poder mostrarlos en
# pantalla, hay que recordar que lifestore_products tiene sus elementos
# [id, nombre, ,precio, categoria]
                    cont=1
                    for p in lifestore_products:
                        for i in [3,57]:
                            if p[0]==i:
                                c = str(cont)
                                sp1 = str(p[0])
                                sp2 = p[3]
                                sp3 = p[1]

                                c="|"+1*" "+"#"+c+(2-len(c))*" "
                                sp1 = 1*" "+sp1+(2-len(sp1))*" "+2*" "
                                sp2 = 1*" "+sp2+(17-len(sp2))*" "


                                print(c+s+sp1+s+sp2+s+sp3)
                                print(" ")
                                cont+=1
# se muestra el quinto grupo en pantalla
                    print("""
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #\
 # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
 ---------------------------------------------------------------\
----------------------------
|      #5       |Mantener el inventario completo en epocas de may\
ores ventas               |
|   Productos   |                                                \
                          |
```

```
| muy conocidos | 1) Campañas de ventas de alcance moderado      \
                            |
|  atrayentes   |                                                \
                            |
|-------------------------------------------------------------\
---------------------------
|Caracteristicas| Productos que tienen una alta pertenencia en el\
 mercado al ser conocidos   |
|               | y generar atención que se traduce en un ingreso\
del capital                 |
 -------------------------------------------------------------\
---------------------------|
| No | id. |    categoria     | nombre                         \
                            |
 -------------------------------------------------------------\
---------------------------""")
# cont nos ayuda a llevar una enumeracion de las salidas
# una vez encontrados los elementos que deben de mostrarse para
# este apartado, iteramos sobre ellos y mostramos solo aquellos
# que coincidan con el id solicitado.
#
# se colocaran en string para que sea facil poder mostrarlos en
# pantalla, hay que recordar que lifestore_products tiene sus elementos
# [id, nombre, ,precio, categoria]
                    cont = 1
                    for p in lifestore_products:
                        for i in [54]:
                            if p[0]==i:
                                c = str(cont)
                                sp1 = str(p[0])
                                sp2 = p[3]
                                sp3 = p[1]

                                c="|"+1*" "+"#"+c+(2-len(c))*" "
                                sp1 = 1*" "+sp1+(2-len(sp1))*" "+2*" "
                                sp2 = 1*" "+sp2+(17-len(sp2))*" "

                                print(c+s+sp1+s+sp2+s+sp3)
                                print(" ")
                                cont+=1
# se imprimimen las conclusiones para el analisis, y soluciones
# para mejorar el rendimiento de la tienda, asi como la menera
# propuesta de tratar el inventario
                    print("La rotacion de inventario debe hacerse\
 considerando los grupos anteriores asi como el mes con mayores ventas")
                    print("Se recomienda tener una rotación de in\
ventario de entre 3-4 vueltas por año, tratando de tener la mayor cantidad")
                    print("de este mismo para los primeros  4 mes\
es siendo abril el de mayor demanda de productos")

                    print(" ")
# pregunta al usuario si desea salir de la sesión o solo al lobby
                    condicion_lobby=int(input("Para voler al lobby \
```

```
teclea '1', para cerrar sesión teclear '0' "))

                elif consulta==5:
# si en la parte del lobby el usuario ingreso el numero 5
# quiere decir salir de la sesion por lo tanto la condicion
# lobby se pasa a 0 y se termina la iteracion
                    condicion_lobby=0
                    continue
                else:
# aviso de que el valor introducido se sale del rango o no es una
# entrada valida
                    print("Entrada no valida, debe de ser un nume\
ro entero de entre 1-4")
                    continue
            else:
# se ejecuta la advertencia cuando la contrseña escrita por el ususario
# no existe en la variable contraseñas y se entiende que no existe
                print("Contraseña Incorrecta")

        else:
# se ejecuta la advertencia cuando el usuario escrito por el ususario
# no existe en la variable Usuarios y se entiende que no existe
            print("Usuario Incorrecto")

    elif menu_p1==2:
# si en la pantalla principal se ingreso el numero 2, el programa ejecutara
# las lineas de codigo para dar de alta un usuario y una contraseña que
# quedaran guardas en las listas Usuarios y Contraseñas para poder ingresar
# con estas claves, pero al crear el usuario pedira otros datos extra
# para saber quieres han creado nuevos usuarios
        print(Crear1)
        nomc = input("Nombre completo: ")
        print(Crear2)
        carg = input("Cargo: ")
        print(Crear3)
        IDE = input("IDE: ")
        print(Crear4)
        e_mail = input("e.mail: ")
        print(Crear5)
        contr = input("Contraseña: ")
        if len(contr)>=6 and len(contr)<=15:
            print(Crear6)
            contr2 = input("Contraseña: ")
        else:
# se ejecuta esta linea si la contraseña carece de la longitud minima
# de caracteres
            print("Contraseña no cuenta con la longitud requerida\
, intentelo de nuevo.")
            continue

        if contr==contr2:
# la siguiente parte del codigo evalua si ambas contraseñas coinciden
# ya que al crear un usuario te pide escribir la contraseña 2 veces
```

```
# si son correctas y coinciden entonces se despliega un menu
# para confirmar si la información es correcta
            print("Favor de comprobar que su información es correcta : ")
            print("Nombre: "+nombc)
            print("Cargo: "+carg)
            print("IDE: "+IDE)
            print("e-mail:"+e_mail)
            print("contraseña"+len(contr)*"*")
            print(Confirmacion)
# pide al usuario que introduzca 1 si es correcta 2 si no es correcta
            confirmacion = int(input("1-2"))
# al dar uno se entiende que son correctas y se guardan par
# poder ingresar con el nuevo usario y contraseña
            if confirmacion == 1:
                Info.append([entrada2,entrada3,entrada5])
                Usuarios.append(entrada4)
                Contraseñas.append(entrada7)
            else:
                continue

        else:
# si ambas contraseñas no coinciden se muestra el mensaje y se repite el bucle
            print("Contraseñas no coinciden intentelo de nuevo")
```

El programa cuenta con un menú principal (figura 1) que nos permitirá decidir si iniciar sesión con un usuario ya existente o crear uno nuevo, el cual es un aparatado aparte que nos permitira accesar registrandonos (figura 8). La segunda pantalla tiene un login para el usuario (figura 2) que nos guiara para introducir las claves de acceso, el programa esta preparado para mandar mensajes de error si alguna de las claves es incorrecta, un lobby o menú de acción (figura 3) que nos permite acceder a la información puntual de nuestro interés, entre las opciones para acceder y consultar la información que tenemos.

1. Que nos permite conocer los 15 productos más vendidos así como los 20 productos con mayorea búsquedas, además de los 15 y 20 productos con menores ventas y búsquedas por categoría (figura 4).

2. Nos permite visualizar los productos 10 productos con mayores reseñas así como los 10 peores considerando devoluciones (figura 5).

3. Nos permite tener una lista de los ingresos y ventas promedio mensuales (figura 6), así como los ingresos y ventas totales por año y las ganancias totales por año.

4. La cual nos permite tener acceso a un resumen de toda la información recopilada, y estrategias para tratar el exceso de inventario (figura 7).

5. Nos permite salir de la sesión y nos imprime el menu principal.

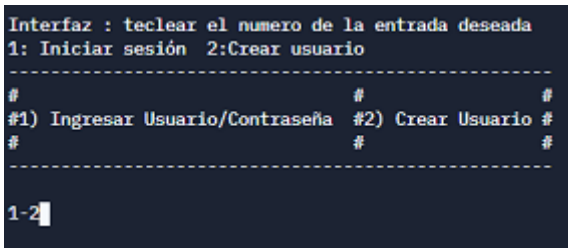


Figura 1: *menu principal de la interfaz grafica*

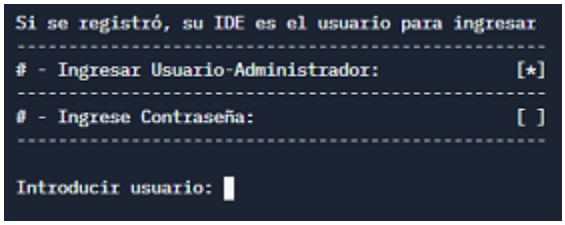


Figura 2: *Login de usuario con contraseña*

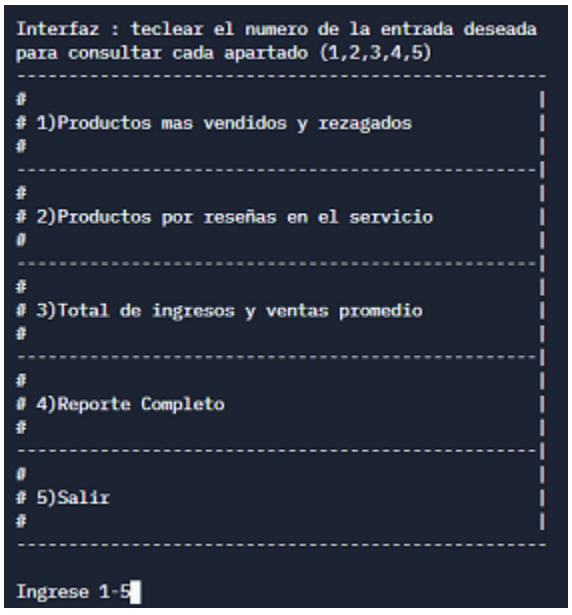
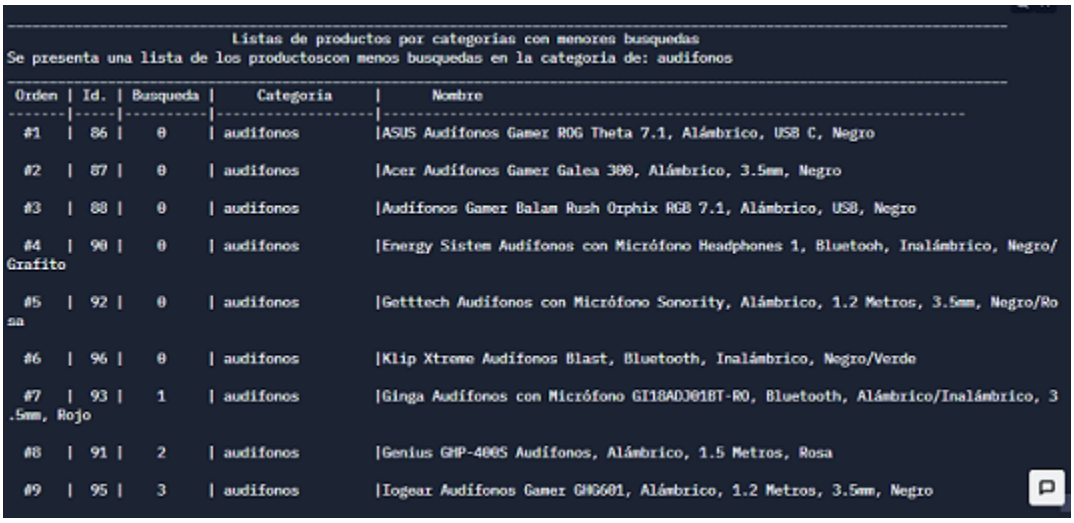
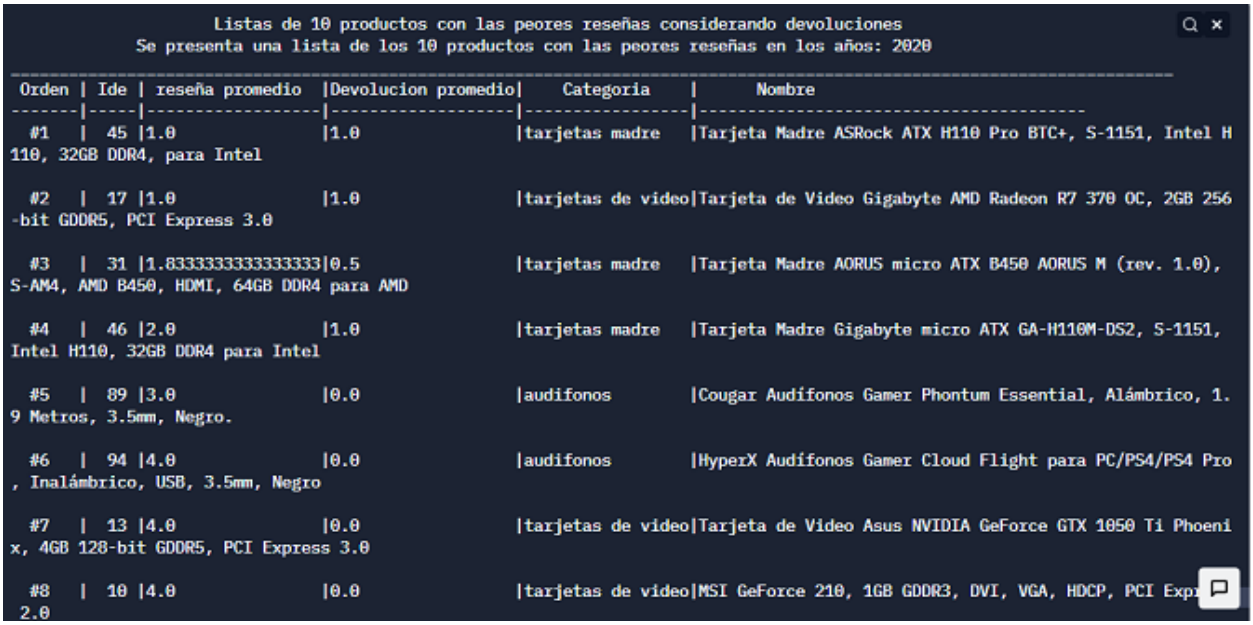


Figura 3: *Menu de acción para consultar apartados*



```
                                Listas de productos por categorias con menores busquedas
Se presenta una lista de los productoscon menos busquedas en la categoria de: audifonos

 Orden | Id. | Busqueda |     Categoria     |      Nombre
-------|-----|----------|-------------------|-------------------------------------------------------------------------
  #1   |  86 |    0     | audifonos         |ASUS Audífonos Gamer ROG Theta 7.1, Alámbrico, USB C, Negro

  #2   |  87 |    0     | audifonos         |Acer Audífonos Gamer Galea 300, Alámbrico, 3.5mm, Negro

  #3   |  88 |    0     | audifonos         |Audífonos Gamer Balam Rush Orphix RGB 7.1, Alámbrico, USB, Negro

  #4   |  90 |    0     | audifonos         |Energy Sistem Audífonos con Micrófono Headphones 1, Bluetooh, Inalámbrico, Negro/
Grafito

  #5   |  92 |    0     | audifonos         |Getttech Audífonos con Micrófono Sonority, Alámbrico, 1.2 Metros, 3.5mm, Negro/Ro
sa

  #6   |  96 |    0     | audifonos         |Klip Xtreme Audífonos Blast, Bluetooth, Inalámbrico, Negro/Verde

  #7   |  93 |    1     | audifonos         |Ginga Audífonos con Micrófono GI18ADJ01BT-RO, Bluetooth, Alámbrico/Inalámbrico, 3
.5mm, Rojo

  #8   |  91 |    2     | audifonos         |Genius GHP-400S Audífonos, Alámbrico, 1.5 Metros, Rosa

  #9   |  95 |    3     | audifonos         |Iogear Audífonos Gamer GHG601, Alámbrico, 1.2 Metros, 3.5mm, Negro
```

Figura 4: *opcion numero uno dentro del menu de apartados*



```
                Listas de 10 productos con las peores reseñas considerando devoluciones
            Se presenta una lista de los 10 productos con las peores reseñas en los años: 2020

 Orden | Ide | reseña promedio  |Devolucion promedio|    Categoria    |      Nombre
-------|-----|------------------|-------------------|-----------------|-------------------------------------
  #1   |  45 |1.0               |1.0                |tarjetas madre   |Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151, Intel H
110, 32GB DDR4, para Intel

  #2   |  17 |1.0               |1.0                |tarjetas de video|Tarjeta de Video Gigabyte AMD Radeon R7 370 OC, 2GB 256
-bit GDDR5, PCI Express 3.0

  #3   |  31 |1.8333333333333333|0.5                |tarjetas madre   |Tarjeta Madre AORUS micro ATX B450 AORUS M (rev. 1.0),
S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD

  #4   |  46 |2.0               |1.0                |tarjetas madre   |Tarjeta Madre Gigabyte micro ATX GA-H110M-DS2, S-1151,
Intel H110, 32GB DDR4 para Intel

  #5   |  89 |3.0               |0.0                |audifonos        |Cougar Audífonos Gamer Phontum Essential, Alámbrico, 1.
9 Metros, 3.5mm, Negro.

  #6   |  94 |4.0               |0.0                |audifonos        |HyperX Audífonos Gamer Cloud Flight para PC/PS4/PS4 Pro
, Inalámbrico, USB, 3.5mm, Negro

  #7   |  13 |4.0               |0.0                |tarjetas de video|Tarjeta de Video Asus NVIDIA GeForce GTX 1050 Ti Phoeni
x, 4GB 128-bit GDDR5, PCI Express 3.0

  #8   |  10 |4.0               |0.0                |tarjetas de video|MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PCI Exp
 2.0
```

Figura 5: *opcion numero dos dentro del menu de apartados*

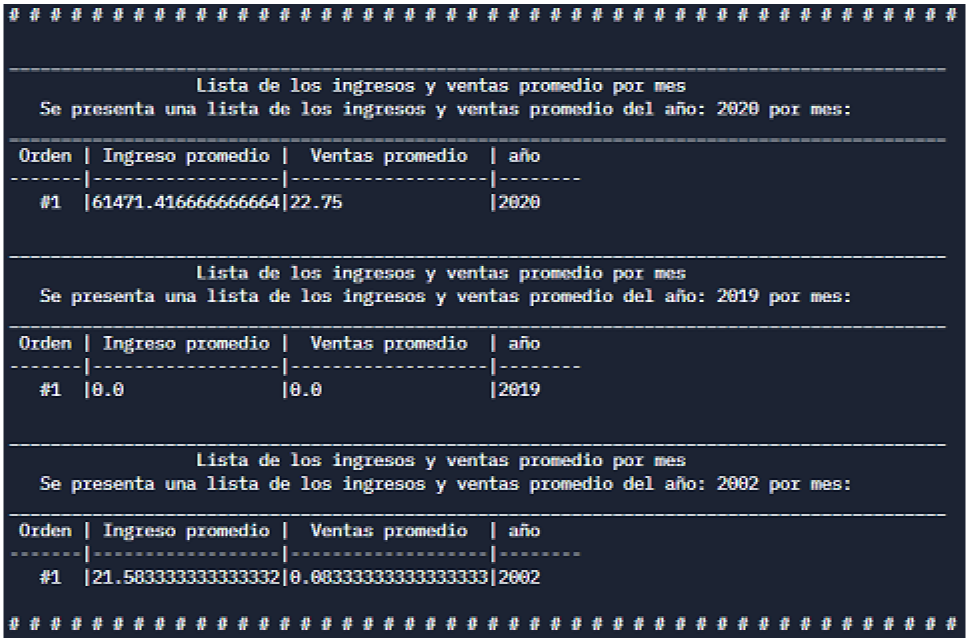


Figura 6: *opcion numero tres dentro del menu de apartados*

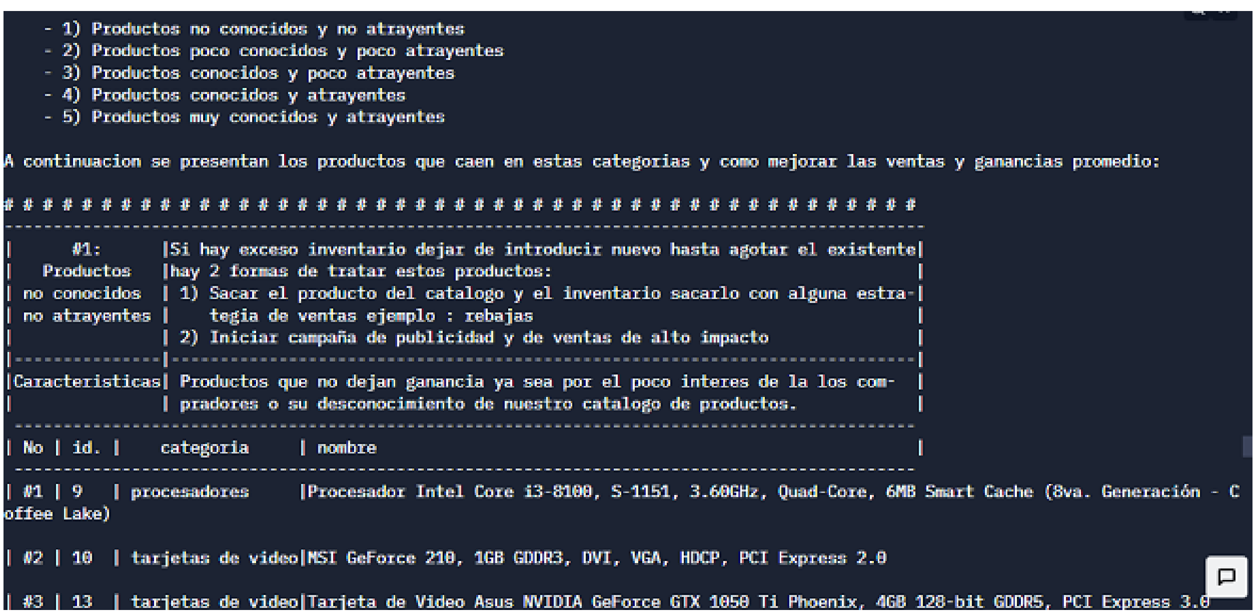


Figura 7: *opcion numero cuatrp dentro del menu de apartados*

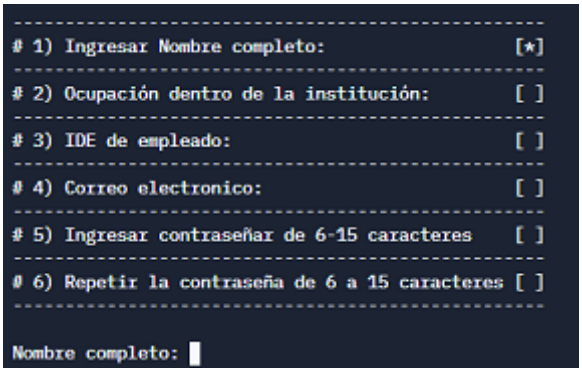


Figura 8: *Menu principal para crear usuario nuevo*

Solución al problema.

Planteamiento

Considerando el tener los datos necesarios para el análisis, ordenados y organizados podremos proponer estrategias que puedan satisfacer lo solicitado en el problema, con el objetivo de mejorar las ganancias y minimizar los gastos de la empresa Lifestore al analizar el comportamiento de sus ventas, así como su rotación de inventario.

Es un buen punto de partida una vez teniendo las listas, seleccionar aquellas que aporten cierta relevancia para saber si un artículo es bien recibido por los compradores, si goza de buena reputación o si es conocido dentro del catálogo de productos que tiene la empresa.

Teniendo estas listas que en nuestro caso de estudio fueron las siguientes:

1. ventas totales por artículos dividida en años y categorías de productos
2. búsquedas totales por articulo dividida en años y categorías de productos
3. reseñas promedio de cada articulo

podemos considera que estas tres características repercutirán directamente en que tan bien aceptado sea un producto, en este caso se enuncia que las reseñas están directamente relacionadas a la calidad del producto, si un producto tiene bajas valoraciones se espera que puedan darse un reembolso por eso este último criterio no forma parte de este paso del análisis.

Lo siguiente será utilizar un agrupamiento k-medias, el cual es un tipo de análisis exploratorio que nos brinda la oportunidad de agrupar elementos dependiendo en las características comunes que estos tengan, dicho análisis forma parte de un grupo de técnicas que deben de hacerse en todo conjunto de datos para identificar grupos y encontrar explicaciones para ciertos comportamientos.

En nuestro caso nuestros nuestras observaciones serán los productos de la tienda Lifestore, y las características serán:

- El número de ventas totales en el año
- Las búsquedas totales
- Las valoraciones promedio en el año

La explicación de porque se utilizaron estas tres características es porque el número de ventas totales representa que tan confianza inspira el comprador. El número de búsquedas totales repercute en que tan conocido o con que tanta fama cuenta el producto dentro de la empresa, y que tan conocido es dentro del catálogo. Las reseñas promedio o valoraciones promedio representan que tan satisfechos han quedado los clientes y

que tan probable es que ellos hablen de nuestros productos, aumento el alcance y permitiéndole al producto contar con un renombre dentro de la compañía.

Agrupamiento k-medias

metodo del codo

Teniendo en claro nuestro plan de acción, procederemos a obtener el número de agrupamientos óptimos, usando el método del codo, como se muestre en la (figura 9) podemos observar que cada categoría tiene un número determinado de agrupamientos, escogeremos los que a nuestro criterio sean los óptimos y graficaremos las entradas teniendo en cuenta el agrupamiento de cada uno.

Es importante aclarar que la categoría de *memorias USB* no formó parte del análisis de agrupamientos puesto que no cuenta con las muestras suficientes para agruparlo, por lo cual utilizando una inspección por sus características importantes para determinaren que grupo pertenece.

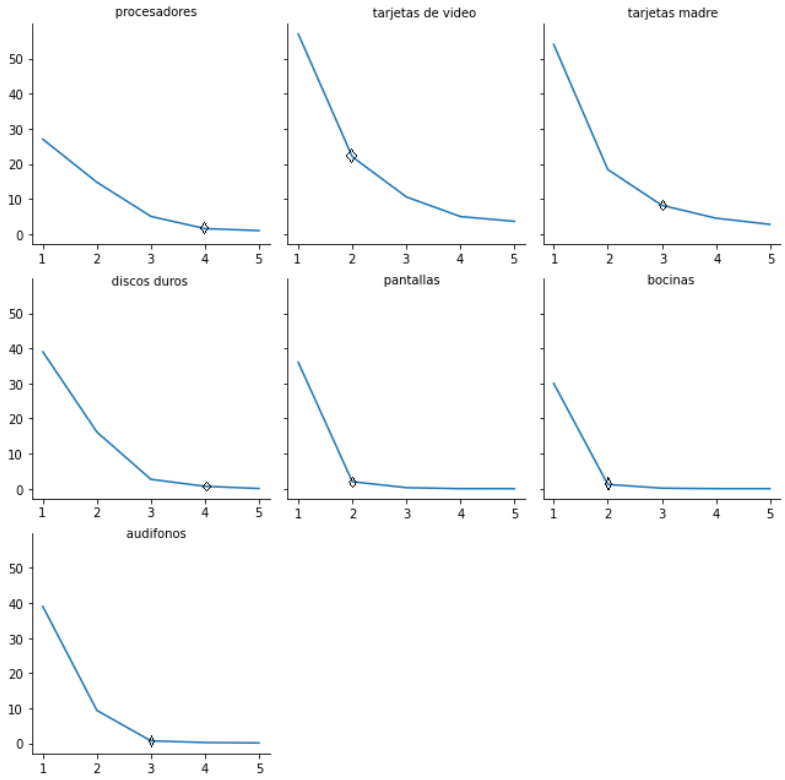


Figura 9: *Metodo del codo para determinar numero optimo de agrupamientos*

Representación grafica de los conjuntos

Utilizando un diagrama de dispersión podemos observar por la diferencia de colores y la aglomeración en ciertos puntos, podemos comenzar a analizar y dar una explicación a este resultado. Como se pude ver en la (figura 10) podemos apreciar por la diferencia de colores que cada categoría tienen su agrupamientos, pero podemos encontrar ciertas similitudes entre los agrupamientos de diferentes categorías, y de acuerdo a que el eje de las **abscisas** es el número de ventas y el eje de las **ordenadas** es el número de búsquedas, podemos nombrar estos agrupamientos, teniendo en consideración que uno de los criterios para el agrupamiento fueron las reseñas promedio de cada producto, donde cada punto en la (figura 10) es un producto.

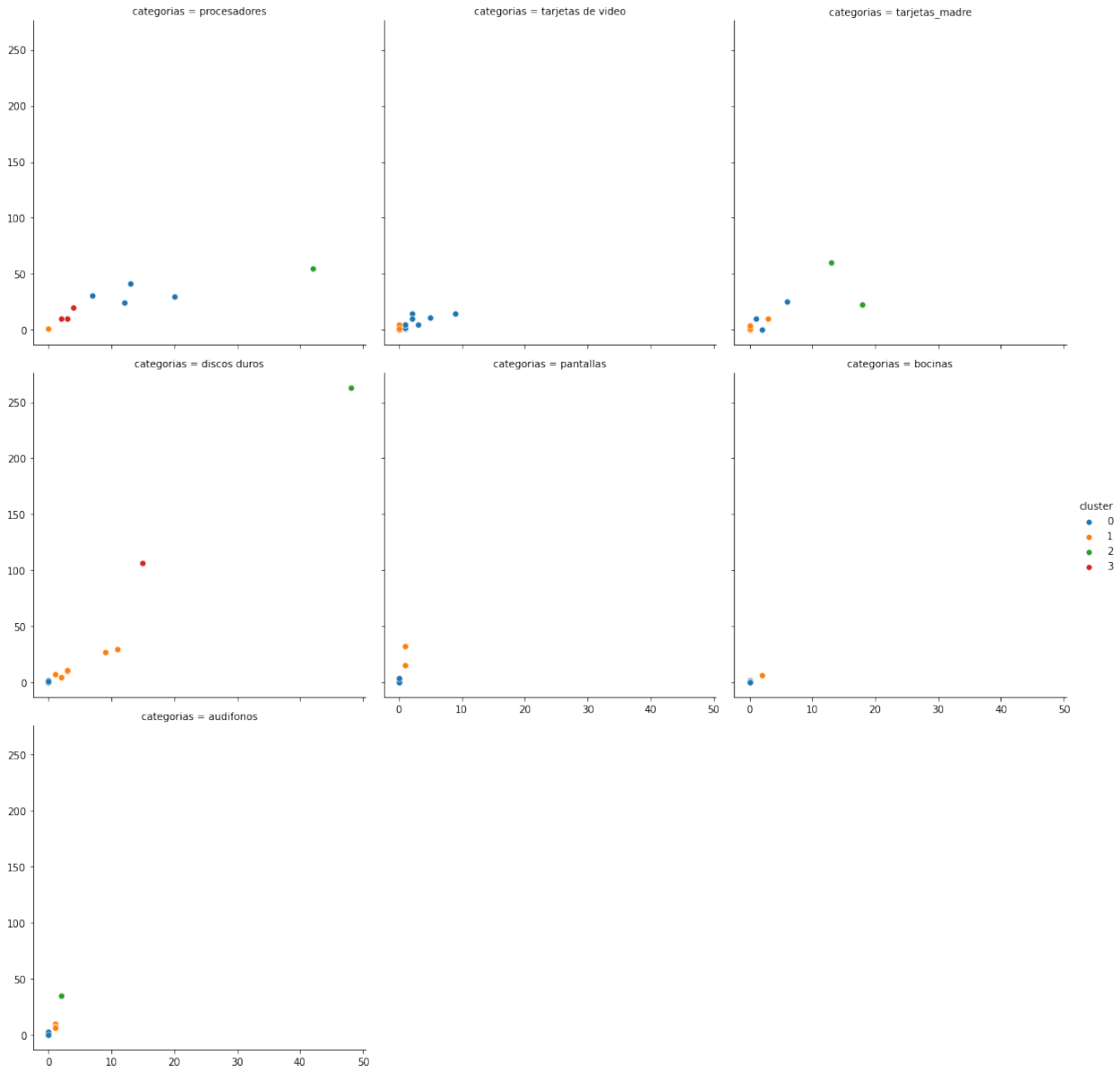


Figura 10: *agrupamiento k-medias para cada categoria de productos en existencia*

Agrupamientos

estrategias

Después de analizar la gráfica de los agrupamientos, podemos nombrar los grupos de la siguiente manera, para posteriormente proponer estrategias de cómo tratarlos. Los grupos son los siguientes:

1. **Productos no conocidos y no atrayentes**: Son aquellos productos que no dejan ganancia ya sea por el poco interés de los compradores o su desconocimiento del artículo en nuestro catálogo son productos que se caracterizan por tener malas reseñas o no tener ninguna puesto que no se hay registros de sus compras en el año 2020. La manera de tratar estos artículos es de dos formas.

- Quitándolos completamente del catálogo de la empresa, y para recuperar una parte de la inversión inicial para adquirir los productos que están almacenados es alguna estrategia de ventas como lo son las rebajas.
- Si se está seguro de la pertenencia de dichos artículos en el catálogo se sugiere no adquirir más productos de los existentes, y que se inicien campañas de publicidad y ventas de alcance e impacto alto. Esto podría presentar una gran inversión que puede no recuperarse.

```
           categoria                                              nombre
9         procesadores  Procesador Intel Core i3-8100, S-1151, 3.60GHz...
10  tarjetas de video  MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PC...
13  tarjetas de video  Tarjeta de Video Asus NVIDIA GeForce GTX 1050 ...
14  tarjetas de video  Tarjeta de Video EVGA NVIDIA GeForce GT 710, 2...
15  tarjetas de video  Tarjeta de Video EVGA NVIDIA GeForce GTX 1660 ...
16  tarjetas de video  Tarjeta de Video EVGA NVIDIA GeForce RTX 2060 ...
17  tarjetas de video  Tarjeta de Video Gigabyte AMD Radeon R7 370 OC...
19  tarjetas de video  Tarjeta de Video Gigabyte NVIDIA GeForce GTX 1...
20  tarjetas de video  Tarjeta de Video Gigabyte NVIDIA GeForce RTX 2...
23  tarjetas de video  Tarjeta de Video MSI Radeon X1550, 128MB 64 bi...
24  tarjetas de video  Tarjeta de Video PNY NVIDIA GeForce RTX 2080, ...
26  tarjetas de video  Tarjeta de Video VisionTek AMD Radeon HD 5450,...
27  tarjetas de video  Tarjeta de Video VisionTek AMD Radeon HD5450, ...
30     tarjetas madre  Tarjeta Madre AORUS ATX Z390 ELITE, S-1151, In...
32     tarjetas madre  Tarjeta Madre ASRock Z390 Phantom Gaming 4, S-...
34     tarjetas madre  Tarjeta Madre ASUS ATX ROG STRIX B550-F GAMING...
35     tarjetas madre  Tarjeta Madre Gigabyte micro ATX Z390 M GAMING...
36     tarjetas madre  Tarjeta Madre Gigabyte micro ATX Z490M GAMING ...
37     tarjetas madre  Tarjeta Madre ASRock ATX Z490 STEEL LEGEND, S-...
38     tarjetas madre  Tarjeta Madre Gigabyte Micro ATX H310M DS2 2.0...
39     tarjetas madre  ASUS T. Madre uATX M4A88T-M, S-AM3, DDR3 para ...
41     tarjetas madre  Tarjeta Madre ASUS micro ATX Prime H370M-Plus/...
43     tarjetas madre  Tarjeta Madre ASUS ATX ROG STRIX Z390-E GAMING...
45     tarjetas madre  Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151...
46     tarjetas madre  Tarjeta Madre Gigabyte micro ATX GA-H110M-DS2,...
31     tarjetas madre  Tarjeta Madre AORUS micro ATX B450 AORUS M (re...
53       discos duros  SSD Addlink Technology S70, 512GB, PCI Express...
55       discos duros  SSD para Servidor Supermicro SSD-DM128-SMCMVN1...
56       discos duros  SSD para Servidor Lenovo Thinksystem S4500, 48...
58       discos duros  SSD para Servidor Lenovo Thinksystem S4510, 48...

       categoria                                          nombre
59  discos duros           SSD Samsung 860 EVO, 1TB, SATA III, M.2
```

```
60  memorias usb  Kit Memoria RAM Corsair Dominator Platinum DDR...
61  memorias usb  Kit Memoria RAM Corsair Vengeance LPX DDR4, 24...
62     pantallas  Makena Smart TV LED 32S2 32'', HD, Widescreen,...
63     pantallas  Seiki TV LED SC-39HS950N 38.5, HD, Widescreen,...
64     pantallas  Samsung TV LED LH43QMREBGCXGO 43, 4K Ultra HD,...
65     pantallas  Samsung Smart TV LED UN70RU7100FXZX 70, 4K Ult...
68     pantallas  Makena Smart TV LED 40S2 40'', Full HD, Widesc...
69     pantallas  Hisense Smart TV LED 40H5500F 39.5, Full HD, W...
70     pantallas  Samsung Smart TV LED 43, Full HD, Widescreen, ...
71     pantallas  Samsung Smart TV LED UN32J4290AF 32, HD, Wides...
72     pantallas  Hisense Smart TV LED 50H8F 49.5, 4K Ultra HD, ...
73     pantallas  Samsung Smart TV LED UN55TU7000FXZX 55, 4K Ult...
75       bocinas  Lenovo Barra de Sonido, Alámbrico, 2.5W, USB, ...
76       bocinas  Acteck Bocina con Subwoofer AXF-290, Bluetooth...
77       bocinas  Verbatim Bocina Portátil Mini, Bluetooth, Inal...
78       bocinas  Ghia Bocina Portátil BX300, Bluetooth, Inalámb...
79       bocinas  Naceb Bocina Portátil NA-0301, Bluetooth, Inal...
80       bocinas  Ghia Bocina Portátil BX800, Bluetooth, Inalámb...
81       bocinas  Ghia Bocina Portátil BX900, Bluetooth, Inalámb...
82       bocinas  Ghia Bocina Portátil BX400, Bluetooth, Inalámb...
83       bocinas  Ghia Bocina Portátil BX500, Bluetooth, Inalámb...
86     audifonos  ASUS Audífonos Gamer ROG Theta 7.1, Alámbrico,...
87     audifonos  Acer Audífonos Gamer Galea 300, Alámbrico, 3.5...
88     audifonos  Audífonos Gamer Balam Rush Orphix RGB 7.1, Alá...
90     audifonos  Energy Sistem Audífonos con Micrófono Headphon...
91     audifonos  Genius GHP-400S Audífonos, Alámbrico, 1.5 Metr...
92     audifonos  Getttech Audífonos con Micrófono Sonority, Alá...
93     audifonos  Ginga Audífonos con Micrófono GI18ADJ01BT-RO, ...
95     audifonos  Iogear Audífonos Gamer GHG601, Alámbrico, 1.2 ...
96     audifonos  Klip Xtreme Audífonos Blast, Bluetooth, Inalám...
```

2. **Productos poco conocidos, poco atrayentes** : son aquellos productos que carecen de interés por parte de los compradores, no conocen de su existencia salvo por algunos compradores, la poca fama que tienen y el desconocimiento provoca que no se sienten animados a adquirirlos. La manera de tratar estos artículos es de la siguiente manera.

- No adquirir nuevos productos hasta no agotar los existentes, puesto que tener productos almacenados que no se venden en un tiempo considerable genera pérdidas, una vez implementadas las siguientes estrategias se puede volver a adquirir inventario. Se propone iniciar campañas de publicidad y ventas de alcance e impacto altos.

```
           categoria                                          nombre
1        procesadores  Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Q...
6        procesadores  Procesador Intel Core i9-9900K, S-1151, 3.60GH...
8        procesadores  Procesador Intel Core i5-9600K, S-1151, 3.70GH...
11  tarjetas de video  Tarjeta de Video ASUS AMD Radeon RX 570, 4GB 2...
12  tarjetas de video  Tarjeta de Video ASUS NVIDIA GeForce GTX 1660 ...
18  tarjetas de video  Tarjeta de Video Gigabyte NVIDIA GeForce GT 10...
21  tarjetas de video  Tarjeta de Video MSI AMD Mech Radeon RX 5500 X...
22  tarjetas de video  Tarjeta de Video MSI NVIDIA GeForce GTX 1050 T...
25  tarjetas de video  Tarjeta de Video Sapphire AMD Pulse Radeon RX ...
28  tarjetas de video  Tarjeta de Video Zotac NVIDIA GeForce GTX 1660...
33     tarjetas madre  Tarjeta Madre ASUS ATX PRIME Z390-A, S-1151, I...
```

40	tarjetas madre	Tarjeta Madre Gigabyte XL-ATX TRX40 Designare,...
47	discos duros	SSD XPG SX8200 Pro, 256GB, PCI Express, M.2
48	discos duros	SSD Kingston A2000 NVMe, 1TB, PCI Express 3.0, M2
49	discos duros	Kit SSD Kingston KC600, 1TB, SATA III, 2.5, 7mm
50	discos duros	SSD Crucial MX500, 1TB, SATA III, M.2
51	discos duros	SSD Kingston UV500, 480GB, SATA III, mSATA
52	discos duros	SSD Western Digital WD Blue 3D NAND, 2TB, M.2
74	bocinas	Logitech Bocinas para Computadora con Subwoofe...
84	audifonos	Logitech Audífonos Gamer G332, Alámbrico, 2 Me...
89	audifonos	Cougar Audífonos Gamer Phontum Essential, Alám...
94	audifonos	HyperX Audífonos Gamer Cloud Flight para PC/PS...

3. **Productos conocidos, poco atrayentes** : Son aquellos productos que gozan de cierta notoriedad pero que no inspira lo suficiente a los compradores para adquirirlos, muestran cierto interés de acercamiento pero no el suficiente para que se concrete una compra. La manera de tratar con estos artículos es la siguiente:

- Tener un inventario bajo, con los suficientes artículos para tener una cuarta parte de las compras por año, se propone iniciar campañas de publicidad de alcance e impacto moderadas en cambio tener campañas de ventas de alcance e impacto altos.

	categoria	nombre
2	procesadores	Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 3...
4	procesadores	Procesador AMD Ryzen 3 3200G con Gráficos Rade...
5	procesadores	Procesador Intel Core i3-9100F, S-1151, 3.60GH...
7	procesadores	Procesador Intel Core i7-9700K, S-1151, 3.60GH...
29	tarjetas madre	Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GA...
42	tarjetas madre	Tarjeta Madre ASRock Micro ATX B450M Steel Leg...
66	pantallas	TCL Smart TV LED 55S425 54.6, 4K Ultra HD, Wid...
67	pantallas	TV Monitor LED 24TL520S-PU 24, HD, Widescreen,...
85	audifonos	Logitech Audífonos Gamer G635 7.1, Alámbrico, ...

4. **Productos conocidos, atrayentes** : Productos que tienen notoriedad dentro del catálogo de la empresa, generan atención que se traduce en un ingreso de capital, estos productos son los preferidos por los compradores. La manera de tratar con estos productos es la siguiente:

- mantener el inventario a un nivel alto la primera mitad del año que es donde más ventas se hacen. Se pueden implementar campañas de publicidad y ventas de alcance e impacto moderado para atraer a nuevos compradores y mantener a los antiguos

	categoria	nombre
3	procesadores	Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, S...
57	discos duros	SSD Adata Ultimate SU800, 256GB, SATA III, 2.5...

	categoria	nombre
0	procesadores	Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, S...
1	discos duros	SSD Adata Ultimate SU800, 256GB, SATA III, 2.5...

5. **Productos muy conocidos y atrayentes** : Productos que tienen una alta pertenencia en el mercado al ser conocidos y generar atención que se traduce en un ingreso del capital alto, estos productos son conocidos y los compradores se sienten atraídos a adquirirlos. La manera de tratar estos productos es la siguiente.

- mantener el inventario alto en los meses de mayores compras, estos productos no necesitan tanta inversión en campañas, pero nunca está de más tener campañas de ventas para mantener al producto siendo llamativo para los compradores.

```
       categoria                                          nombre
54  discos duros  SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm
```

Teniendo los productos identificados y con sus respectivas estrategias propuesta, nos es importante determinar que meses son los más demandantes, para así tener una buena estrategia de rotación de inventario.

Como podemos apreciar en la *figura 11* el cual es una mapa de calor, su función es poder mostrar la intensidad de las características entre observaciones, para poder identificar qué mes fue más significativo en cuanto ventas, en el eje horizontal podemos apreciar las características de ingresos promedio y ventas del otro, en el eje vertical podemos ver los meses, la intensidad del color de más apagado a más brillante nos muestra que mes en el que se obtuvieron mayores ventas, el cual fue el mes de marzo.

Es importante recalcar que los análisis aquí proporcionados son del año 2020, ya que es el único año de las muestras de ventas que nos presenta una cantidad significativa de datos que podemos analizar.

Se puede observar que la mayor concentración de ventas se encuentra en los primeros cinco meses.

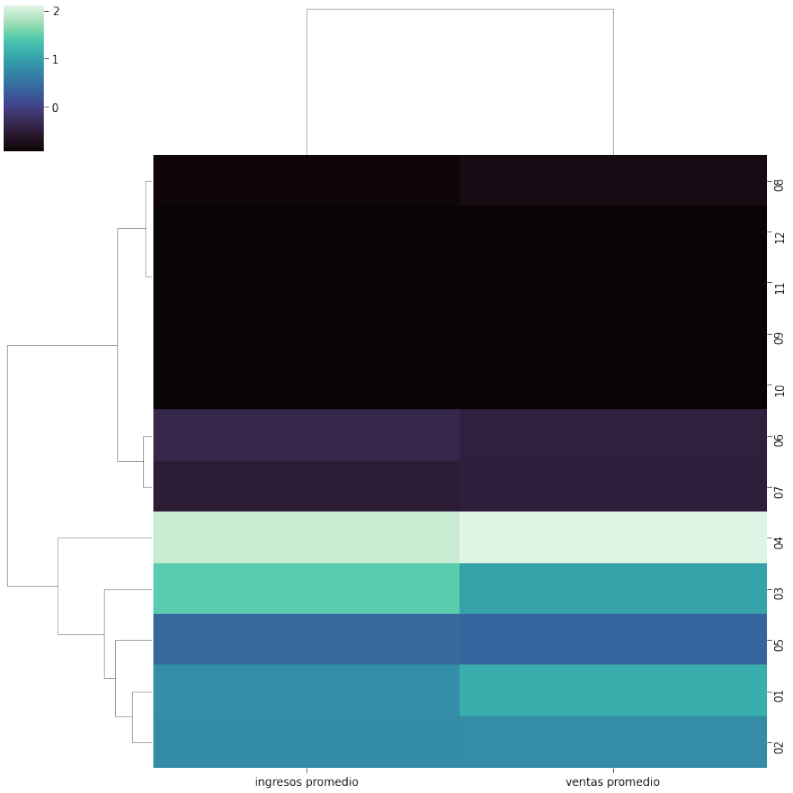


Figura 11: *Mapa de calor para comparar el nivel de ingresos y compras*

Se puede denotar una mayor cantidad de ventas en los primeros cinco meses por lo tanto una buena alternativa para la rotación de inventario considerando los agrupamientos de los productos así como la identificación de los meses, podemos decir que la cantidad de inventario debe de programarse considerando los meses con mayores ventas, y si se debe de adquirir más productos para inventario debemos asegurar de que estemos moviendo los productos almacenados al menos unas 4 veces por año, manteniendo la mayor cantidad en los primeros cinco meses del año.

Conclusión

En conclusión, los datos analizados solo fueron concluyentes para aquellas muestras que tuvieran relevancia con ventas hechas en el año 2020.

Se implementó un código que permite la obtención de los datos de forma eficiente, utilizando las estructuras básicas que el curso cumple, se implementaron bloques de código que suplen la ausencia de métodos especializados no abarcados en el curso, pero esto permitió al código ser más apegado a las características esperadas para la salida. Se implementó una sintaxis graficas para que el usuario se sintiese en control de la información, para poder consultar en cualquier momento y cuantas veces se deban de consultar, dicha implementación permite un flujo de la información de manera orgánica, elegante y simple. Además de contar con seguridad que garantice un control y registro de quien puede consultar la información.

Mediante una serie de análisis exploratorios utilizando el conjunto de datos ya ordenado y correctamente tratado se logró identificar cinco grupos que describen el comportamiento de los productos en relación a que tanta confianza inspiran a los comprados para adquirirlos o que tanto son conocidos dentro de la empresa, de estos cinco grupos se logró aportar estrategias que aplicadas de manera inmediata pueden representar una alza en las ventas, un ahorro de los gastos y en general una mejoría en el modo de operar de la empresa. Se lograron identificar los meses más representativos en cuanto a mayores compras, y en base a esta información y el agrupar los productos podemos conocer cuales beben de tener una mejor rotación de inventario y cuales reducirlo ya que solo generaría perdidas tener un exceso de inventario de un producto que no se está moviendo.

Se implementaron herramientas muy poderosas para el análisis exploratorio como lo serian el agrupamiento k-medias, el grafico valor cuadrático intragrupo para determinar el número correcto de agrupamientos, así como herramientas visuales que faciliten el entendimiento de la información presentada