



**EMTECH**  
Emerging Technologies Institute

## **Curso:**

Fundamentos de Programación con Python

## **Proyecto 1**

**domingo, 6 de septiembre de 2020**

**Docente/  
Tutor:** Javier Ramírez

**Estudiante:** Ramírez Barrios, Jorge

## Índice

Introducción .....	1
Definición del código .....	2
Solución al problema .....	10
Conclusión .....	11

## Introducción

Durante los últimos años se ha observado con mayor detenimiento los datos que nos ofrecen los eventos, estos eventos son de todo tipo desde una erupción volcánica, juegos de azar, juegos deportivos entre otros; esto se ha permitido que con el tiempo comencemos como humanidad a analizar estos datos de forma que podamos llegar a predecir determinadas circunstancias o poder cambiar por completo la conclusión o incluso el inicio del evento en sí mismo. Con este principio es que de pasamos de usarlo inicialmente en catástrofes las cuales eran primordial predecir a hoy por día aplicarlo a cualquier evento cotidiano, aunque su auge no es tan grande es algo que se aplica en todos lados sin que nos percatemos de ello, a esto se le llama ciencia de datos y es lo que utilizaremos junto con la ayuda de la programación para solventar un problema en una tienda digital.

Para este proyecto analizaremos las ventas de la tienda LifeStore la cual es una tienda virtual que maneja una gran grama de artículos orientados a hardware, la gerencia detecto que tiene una gran cantidad de acumulamiento de inventario y a su vez que los productos más buscados dejaron de ser tan solicitados lo que causo una baja en las ventas del último trimestre. Estos datos por si mismo no nos sirven de mucho debido a que no sabemos las circunstancias solo sabemos los efectos, para la elaboración del proyecto se nos proporcionaron las listas de ventas, listas de productos y una lista de búsquedas de los productos. Estas tres listas son con las que trabajaremos para poder darle una solución objetiva a LifeStore con la que puedan revertir o eliminar sus problemas.

Como punto para esto han solicitado tres consignas que deberemos resolver:

- Productos mas vendidos y productos rezagados: Analizar las categorías con menores ventas y categorías con menores búsquedas.
- Productos con reseña en el servicio: Analizar las categorías con mayores ventas y categorías con mayores búsquedas.
- Sugerir una estrategia: Sugerir productos a retirar del mercado, así como sugerir de cómo reducir la acumulación de inventario considerando los datos de ingresos y venta.

## Definición del código

El proyecto está dividido en dos archivos .py, el primero es el código principal y el segundo contiene las listas proporcionadas por la gerencia. Decidí dividir en dos el proyecto para poder trabajar de una forma que no me estorbaran las declaraciones de variables de las listas las cuales son largas. ( <https://github.com/diangs/Proyecto-01> )

master	Proyecto-01 / Proyecto Python /	Go to file	Add file
diangs Add files via upload 1a26871 now History			
..			
__pycache__	Add files via upload	now	
PROYECTO-01-RAMIREZ-BARRIOS-JORGE.py	Add files via upload	now	
listas.py	Add files via upload	now	

Para el proyecto utilice dos librerías, getpass la cual se utilice para darle un toque de seguridad al Login en el momento de ingresar la contraseña y os que permite interactuar con el sistema, este solo fue utilizado para limpiar la pantalla del prompt. La tercera importación es del archivo que contiene las listas.

404 líneas (383 sloc) 20.3 KB	Raw	Blame	Copy	Edit	Delete
<pre>1 #Importación de librerías y cadenas predeterminadas de datos de ventas 2 import getpass 3 from os import system 4 import listas as l 5</pre>					

A su vez definí tres funciones la primera ordena los arreglos de mayor a menor utilizando como pivote el elemento 0, la segunda crea un formato de impresión para los datos y la tercer que al igual que la primera ordena los arreglos solo que lo hace de menor a mayor.

<pre>9 def orden(best): 10     for k in range(len(best)-1,0,-1): 11         for l in range(k): 12             if best[l]&lt;best[l+1]: 13                 temp = best[l] 14                 best[l] = best[l+1] 15                 best[l+1] = temp 16     return best 17 #Impresion de datos con formato 18 def impresion(best, tipo): 19     if tipo == "ventas": 20         Tabla = """ 21 +-----+-----+-----+ 22   Ventas  ID      Producto 23  -----+-----+-----+ 24 {} 25 +-----+-----+-----+ 26 """ 27         Tabla = (Tabla.format('\n'.join("{}&lt;10&gt; {}&lt;10&gt; {}&lt;120&gt; {}".format(*fila) 28             for fila in best))) 29         print (Tabla)</pre>					
--	--	--	--	--	--

Para el sistema de Login utilice un usuario root con contraseña root predefinidos dentro del mismo archivo, el usuario tiene un total de tres intentos para ingresar el usuario y contraseña correcto de lo contrario el sistema terminara su ejecución. Por cada error se muestra el numero restante de intentos y un mensaje general de error.

```
97 #Declaración de usuario y contraseña
98 user="root"
99 passw="root"
100
101 #Sistema de control de ingreso, se permiten tres intentos para ingresar el usuario y contraseña correctos. Por cada fallo se limpia la pantalla y se despliega un mensaje de er
102 #En caso de que el usuario falle los tres intentos el programa terminara su ejecución y en caso contrario puede ingresar al sistema principal.
103
104 for attempts in range(3, 0, -1):
105     print("**** Sistema de login *** \n")
106     try_user = input("Ingrese el Usuario: ")
107     try_pass = getpass.getpass(prompt="Contraseña: ")
108     system("cls")
109     if try_user == user and try_pass == passw:
110         break
111     elif attempts==1:
112         print("Limite de intentos alcanzados")
113         exit(0)
114     else:
115         print("Contraseña o usuario incorrecto")
116         print("Intentos restantes: "+ str(attempts-1))
117
```

Si el usuario ingresa al sistema se desplegará un menú con las opciones, hay siete opciones disponibles y adicionalmente una que sirve como método para la terminación del programa. Si la opción está dentro del rango de las disponibles el programa procederá con la solicitud creando y vaciando tres variables de uso general, de lo contrario solicitará una opción válida.

```
117
118 #Sistema principal, el sistema principal muestra las opciones disponibles para el usuario y espera que seleccione una, al hacerlo pasa por validaciones.
119
120 opcion = 0
121 while opcion != "10":
122     system("cls")
123     print("**** Bienvenido al Sistema de Control ***\n\n")
124     print("[1] Productos más vendidos \n")
125     print("[2] Productos más buscados \n")
126     print("[3] Productos menos vendidos por categoría \n")
127     print("[4] Productos menos buscados \n")
128     print("[5] Mejores reseñas \n")
129     print("[6] Peores reseñas \n")
130     print("[7] Ingresos y ventas \n")
131     print("[10] Salir \n")
132     opcion = input("Ingrese la opción: ")
133     system("cls")
134     """
135     This is the LifeStore-SalesList data:
136     lifestore-searches = [id_search, id product]
137     lifestore-sales = [id_sale, id_product, score (from 1 to 5), date, refund (1 for true or 0 to false)]
138     lifestore-products = [id_product, name, price, category, stock]
139     """
140     #Se comprueba que el valor seleccionado este dentro del rango de opciones disponibles y que sea valido
141     try:
142         if int(opcion) in range(1,8):
143             supp_list = []
144             best = []
145             categories = []

```

La primera opción disponible lista los 100 productos con mayores ventas, esto lo hace mediante el llenado de una lista con el id de producto de todas las ventas excluyendo aquellas que tuvieron devoluciones y posterior a eso contando las ventas por medio del id guardado comparándolo con los id de la lista de productos. Las ventas, id y nombre del producto se guardan en otra lista que se ordena con la función orden, antes de imprimirla se confirma que la lista solo sea de los 100 productos más vendidos después es impresa por el método impresión (la comprobación de largo de lista de productos se hace con la finalidad de predecir actualizaciones en las listas al igual que en otras opciones dentro del sistema).

```
146
147     #Opcion 1: Listado de los 100 productos más vendidos
148     if int(opcion) == 1:
149         supp_list = []
150         best = []
151         #Lleno una lista con los productos que han sido vendidos excluyendo las devoluciones
152         for sales in l.lifystore_sales:
153             if sales[4] != 1:
154                 supp_list.append(sales[1])
155         #Lleno una lista con la cantidad de veces que ha sido vendido un producto junto a su id y nombre
156         for product in l.lifystore_products:
157             best.append([supp_list.count(product[0]), product[0], product[1]])
158         #Llamo al metodo orden creado previamente para acomodar el arreglo de mayor a menor con referencia a las ventas
159         best = orden(best)
160         #Me aseguro que las ventas a mostrar sean las mejores 100
161         print("Lista de los 100 productos mas vendidos \n\n")
162         if(len(best) > 100):
163             best = best[:100]
164         #Llamo al metodo impresion creado previamente para concanear los datos y posteriormente imprimirlos
165         impresion(best,"ventas")
166         input("Continuar")
167     ...
```

La segunda opción disponible lista los 100 productos con mayores búsquedas, esto lo hace llenando una lista con todos los id de productos de las búsquedas hechas y posterior a esto contando las búsquedas por medio del id guardado comparándolo con los id de la lista productos. Las búsquedas, id y nombre del producto se guardan en otra lista que se ordena con la función orden, antes de imprimirla se confirma que la lista solo sea de los 100 productos más buscados después es impresa por el método impresión.

```
168     #Opcion 2: Listado de los 100 productos con mayores busquedas
169     if int(opcion) == 2:
170         supp_list = []
171         best = []
172         #Lleno una lista con los productos que han sido buscados
173         for serch in l.lifystore_searches:
174             supp_list.append(serch[1])
175         #Llenado de una lista con la cantidad de veces que ha sido buscado un producto junto a su id y nombre
176         for product in l.lifystore_products:
177             best.append([supp_list.count(product[0]), product[0], product[1]])
178         #Llamo al metodo orden creado previamente para acomodar el arreglo de mayor a menor con referencia a las busquedas
179         best = orden(best)
180         #Me aseguro que las ventas a mostrar sean las mejores 100
181         print("Lista de los 100 productos mas vendidos \n\n")
182         if(len(best) > 100):
183             best = best[:100]
184         #Llamo al metodo impresion creado previamente para concanear los datos y posteriormente imprimirlos
185         impresion(best,"busquedas")
186         input("Continuar")
```

La tercera opción disponible lista los 50 productos menos vendidos por categoría, primero recorre la lista de productos buscando todas las categorías presentes en los productos posterior a esto lo imprime al usuario para la selección de la categoría que desee analizar. Con la selección de la categoría del usuario llena una lista de todas los id de productos de las ventas que pertenezcan a la categoría seleccionada (siempre que no tengan devolución) recorriendo la lista de ventas y posterior a esto cuenta los id comparándolo con los productos de la lista de productos considerando que estén dentro de la categoría. Las ventas, id y nombre de producto se guardan en otra lista que se ordena con el método ordeni, antes de imprimirla se confirma que sean los 50 productos con menores ventas.

```

188     #Opcion 3: Listar los 50 productos con menores ventas por categoría
189     if int(opcion) == 3:
190         categories = []
191         for i in l.lifystore_products:
192             if i[3] not in categories:
193                 categories.append(i[3])
194         #Se lista como opciones las categorías encontradas y se espera una seleccion
195         try:
196             while int(opcion) != len(categories)+1:
197                 print("Seleccione la categoría a revisar: \n")
198                 for categorie in categories:
199                     print(f"[*+str(categories.index(categorie)+1)+*] "+categorie)
200                 print(f"[*+str(len(categories))+*] Regresar")
201                 opcion = input("\nSeleccione una opción: ")
202                 system("cls")
203                 supp_list = []
204                 best = []
205                 #Se comprueba que la seleccion este dentro del rango
206                 if int(opcion) in range(1,len(categories)+1):
207                     categorie = categories[int(opcion)-1]
208                     #Ingreso en un arreglo todas las ventas excluyendo las devoluciones
209                     for sales in l.lifystore_sales:
210                         if (sales[4]) != 1:
211                             supp_list.append(sales[1])
212                     #Compruebo los productos y el que este dentro de la categoría seleccionada se comparara con las ventas del arreglo supp_list
213                     #Al comprobarse se ingresa en un arreglo
214                     for product in l.lifystore_products:
215                         if product[3] == categorie:
216                             best.append([supp_list.count(product[0]), product[0], product[1]])
217                     #Llamo al metodo orden creado previamente para acomodar el arreglo de menor a mayor con referencia a las ventas de esta categoría
218                     best = ordeni(best)
219                     #Me aseguro que las ventas de esta categoría a mostrar sean las peores 50
220                     if len(best) > 50:
221                         best = best[:50]
222                     #Llamo al metodo impresion creado previamente para concanear los datos y posteriormente imprimirlos
223                     impresion(best,"ventas")
224                     input("Continuar...")
225                     system("cls")
226         except:
227             print("Ingrese una opción válida.")
228

```

La cuarta opción disponible lista los 100 productos con menores búsquedas por categoría, en esencia hace lo mismo que la tercera opción solo que en lugar de llenar una lista con los id de productos de la lista ventas lo hace con los id de productos de la lista búsquedas. También cambia la comprobación de impresión.

```

229 #Opcion 4: Listar los 100 productos con menores ventas por categoria
230 if int(opcion) == 4:
231     categories = []
232     for i in l.lifstore_products:
233         if i[3] not in categories:
234             categories.append(i[3])
235     #Se lista como opciones las categorias encontradas y se espera una seleccion
236     try:
237         while int(opcion) != len(categories)+1:
238             print("Seleccione la categoria a revisar: \n")
239             for categorie in categories:
240                 print("[ "+str(categories.index(categorie)+1)+"] "+categorie)
241             print("[ "+str(len(categories))+"] Regresar")
242             opcion = input("\nSeleccione una opción: ")
243             system("cls")
244             supp_list = []
245             best = []
246             #Se comprueba que la seleccion este dentro del rango
247             if int(opcion) in range(1,len(categories)+1):
248                 categorie = categories[int(opcion)-1]
249                 #Ingreso en un arreglo todas las busquedas realizadas
250                 for search in l.lifstore_searches:
251                     supp_list.append(search[1])
252                 #Compruebo los productos y los que pertenezcan a la categoria seleccionada seran contados dentro del arreglo supp_list
253                 #Al comprobarse se ingresa a un arreglo
254                 for product in l.lifstore_products:
255                     if product[3] == categorie:
256                         best.append([supp_list.count(product[0]), product[0], product[1]])
257                 #Llamo al metodo orden creado previamente para acomodar el arreglo de menor a mayor con referencia a las busquedas de esta categoria
258                 best = ordeni(best)
259                 #Me aseguro que las busquedas de esta categoria a mostrar sean las peores 100
260                 if len(best) > 100:
261                     best = best[:100]
262                 #Llamo al metodo impresion creado previamente para concanear los datos y posteriormente imprimirlos
263                 impresion(best,"busquedas")
264                 input("Continuar...")
265                 system("cls")
266     except:
267         print("Ingrese una opción valida.")
268

```

La quinta opción disponible lista los 20 productos con mejores reseñas, para esto se recorre toda la lista de productos creando dos variables una contiene el numero de reseñas y la segunda el valor almacenado de esas reseñas, por cada producto se recorre la lista de ventas con la finalidad de comprobar si tiene ventas y por ende reseñas; si tiene ventas se suma el valor de su reseña y aumenta el contador en uno. Si el producto tiene

```

269 #Opcion 5: Listar los 20 productos con mejores reseñas
270 if int(opcion) == 5:
271     supp_list = []
272     best = []
273     #Recorro todo el arreglo de productos
274     for product in l.lifstore_products:
275         count = 0;
276         val = 0;
277         #Recorro todo el arreglo de ventas
278         for sales in l.lifstore_sales:
279             #Verifico si el producto tiene alguna venta de tenerla sumo a val e incremento count
280             if product[0] == sales[1]:
281                 val += sales[2]
282                 count += 1
283         #Agrego a un arreglo todos los productos que tengan por lo menos una reseña junto a su id y nombre
284         if count != 0:
285             best.append([round(val/count,2), product[0], product[1]])
286         #Llamo al metodo orden creado previamente para acomodar el arreglo de mayor a menor con referencia a las reseñas generalbes del producto
287         best = orden(best)
288         #Me aseguro que las reseñas a mostrar sean las mejores 20
289         if len(best) > 20:
290             best = best[:20]
291         #Llamo al metodo impresion creado previamente para concanear los datos y posteriormente imprimirlos
292         impresion(best, "reseña")
293         input("Continuar...")

```



reseñas se suman a una lista con el promedio de reseñas, id de producto y nombre, se ordenan de menor a mayor con el método orden y se imprimen las mejores 20.

La sexta opción disponible lista los 20 productos con peores reseñas, en esta opción no cambia nada comparada con la quinta, la única diferencia es en el ordenamiento. Se llama a la función ordeni en lugar de oren.

```
295         #Opcion 6: Listar los 20 productos con peores reseñas
296         if int(opcion) == 6:
297             supp_list = []
298             best = []
299             #Recorro todo el arreglo de productos
300             for product in l.lifestore_products:
301                 count = 0;
302                 val = 0;
303                 #Recorro todo el arreglo de ventas
304                 for sales in l.lifestore_sales:
305                     #Verifico si el producto tiene alguna venta de tenerla sumo a val e incremento count
306                     if product[0] == sales[1]:
307                         val += sales[2]
308                         count += 1
309                 #Agrego a un arreglo todos los productos que tengan por lo menos una reseña junto a su id y nombre
310                 if count != 0:
311                     best.append([round(val/count,2), product[0], product[1]])
312             #llamo al metodo orden creado previamente para acomodar el arreglo de menor a mayor con referencia a las reseñas generalbes del producto
313             best = ordeni(best)
314             #Me aseguro que las reseñas a mostrar sean las peores 20
315             if len(best) > 20:
316                 best = best[:20]
317             #llamo al metodo impresion creado previamente para concanear los datos y posteriormente imprimirlos
318             impresion(best, "reseña")
319             input("Continuar...")
320     ~~~
```

La séptima opción disponible lista los ingresos y ventas, esta opción tiene un submenú que espera la selección de una de sus opciones. Posterior a la selección comprueba que este dentro del rango de las disponibles o que sea regresar.

```
321         #Opcion 7: Mostrar el total de ingresos y ventas promedio mensuales, total anuales y meses con más ventas del año
322         if int(opcion) == 7:
323             years = []
324             var = []
325             while opcion != "4":
326                 try:
327                     system("cls")
328                     print("[1] Ventas e ingresos promedio mensuales \n")
329                     print("[2] Ventas anuales \n")
330                     print("[3] Meses con mayores ventas \n")
331                     print("[4] Regresar \n")
332                     opcion = input("Ingrese la opción: ")
333                     if int(opcion) in range(1,4):
334
```

Séptima opción submenú uno, recorre la lista de ventas con el fin de obtener todos los meses donde existen ventas. Con estos meses entra en otro ciclo en el que recorre todo el arreglo de meses, donde por cada mes recorre la lista de ventas sumando cada venta en una variable si es esa venta se hizo en ese mes, con el id de producto en venta se busca el precio del producto dentro producto recorriendo la lista de productos. Por cada mes se guarda en una lista el mes, ventas e ingresos que son ordenados con respecto al mes y posteriormente son impresos. Como nota se registraron dos ventas no presentes en el mismo año, pero en diferentes meses por lo que si obtuviéramos el promedio de las

ventas tomando en cuenta esas únicas dos ventas afectarían considerablemente el análisis final por lo que se omitieron para este caso.

```
335         #Mostrar las ventas e ingresos promedio mensuales
336         if int(opcion) == 1:
337             months = []
338             best = []
339             for sales in l.lifefore_sales:
340                 var = sales[3].split("/")
341                 if var[1] not in months:
342                     months.append(var[1])
343             for month in months:
344                 global_sales = 0
345                 income = 0
346                 for sales in l.lifefore_sales:
347                     var = sales[3].split("/")
348                     if var[1] == month:
349                         global_sales += 1
350                         for product in l.lifefore_products:
351                             if sales[1] == product[0]:
352                                 income += product[2]
353             best.append([month, global_sales, income])
354         orden(best)
355         impresion(best, "months")
356         input("Continuar")
```

Séptima opción submenú dos, recorre todas las ventas con el fin de obtener todos los años que presentaron alguna venta. Con estos años entra en otro ciclo en el que recorre todos los años donde existen ventas, donde por cada año recorre la lista de ventas sumando cada venta en una variable si se hizo en ese año. Por cada año se guarda en una lista las ventas totales y el año, las ventas son impresas con la función impresión. Como nota se encontraron tres años que tuvieron ventas sin embargo dos fueron una sola venta, se mostraron estas ventas ya que no afectan el análisis general.

```
358         #Mostrar las ventas anuales
359         if int(opcion) == 2:
360             years = []
361             best = []
362             for sales in l.lifefore_sales:
363                 var = sales[3].split("/")
364                 if var[2] not in years:
365                     years.append(var[2])
366             for year in years:
367                 global_sales = 0
368                 for sales in l.lifefore_sales:
369                     var = sales[3].split("/")
370                     if year == var[2]:
371                         global_sales += 1
372             best.append([global_sales, year, ""])
373         impresion(best, "years")
374         input("Continuar... ")
```

Séptima opción submenú tres, recorre la lista de ventas con el fin de obtener todos los meses donde existen ventas. Con estos meses entra en otro ciclo en el que recorre todo el arreglo de meses, donde por cada mes recorre la lista de ventas sumando cada venta en una variable si es esa venta se hizo en ese mes. Por cada mes se guarda en una lista el mes, ventas e ingresos que son ordenados con respecto al mes y posteriormente son impresos.

```

376         #Meses con mayores ventas
377         if int(opcion) == 3:
378             months = []
379             best = []
380             for sales in l.lifystore_sales:
381                 var = sales[3].split("/")
382                 if var[1] not in months:
383                     months.append(var[1])
384             for month in months:
385                 global_sales = 0
386                 income = 0
387                 for sales in l.lifystore_sales:
388                     var = sales[3].split("/")
389                     if var[1] == month:
390                         global_sales +=1
391             best.append([global_sales, month, ""])
392         orden(best)
393         impresion(best,"bestmonths")
394         input("Continuar")

```

## Solución al problema

Primer punto:

- Como recomendación general les sugeriría bajar drásticamente los pedidos a proveedores de todos los productos ya que contrasta mucho el inventario existente con las ventas obtenidas.
- Una opción que resolvería rápido sus problemas de inventario es eliminar la categoría pantallas, bocinas y audífonos de los pedidos a proveedores ya que no representan ventas significativas y sin embargo el inventario de estos productos es muy elevado. En mi perspectiva una solución adicional podría ser dar promociones de estos productos para poder sacar rápido los inventarios existentes y aprovechar el espacio para otros nuevos.
- Como mencionaba gerencia las búsquedas de productos están cambiando con respecto a las ventas por lo que un cambio de precios para ser competitivos podría ser bueno. Es necesario observar que cambia los productos buscados, pero no la categoría de estos productos por lo que un cambio de generación podría estarlo causando.

Segundo punto:

- Los productos con peores calificaciones son en su mayoría los productos menos vendidos, mejorar la calidad de productos ofrecidos es una buena opción con la que los clientes puedan quedar satisfechos con su compra.
- Dentro de los productos menos buscados por categoría se encuentran en su mayoría productos de gama baja (procesadores, tarjetas de video, tarjetas madre) por lo que renovar inventario es necesario. Sin embargo, las mayores búsquedas se encuentran dentro de la gama media de hardware por lo que es ahí donde debe centrarse esta renovación.

Tercer punto:

- Los primeros cuatro meses del negocio registraron buenas ventas que mostraban un incremento del primero a cuarto mes mismas que decayeron a la mitad en el quinto mes, estas ventas fueron decreciendo con el tiempo hasta llegar al onceavo mes.
- Las mejores ventas fueron en el cuarto mes y cayeron a la mitad en el quinto drásticamente, este tipo de cambios puede que hayan sido producidos externamente lo que requeriría un análisis externo de la situación ya que no hay mucho que analizar desde este punto. Estas situaciones pueden ser apertura de un nuevo negocio, cambio generacional, nueva gama de productos entre otros.
- Si hubo ajustes en las practicas de estas ventas o un cambio en su organización deberían observar estos mismos ya que podrían estar afectando las ventas.

## Conclusión

La ciencia de datos es una de las principales causas actualmente de cambios de rumbo de muchos negocios ya que nos ayuda a predecir algunos sucesos con los datos presentados sin embargo no es 100% certera en todos los cálculos hechos ya que puede haber variables que desconozcamos y afecten todo lo planteado como en este proyecto.

Para el caso de este proyecto observamos los datos presentados sin embargo solo en mi perspectiva podríamos hacer recomendaciones de inventario ya que es son los principales datos que se nos muestran por su relación con las ventas y los productos existentes. Dar una recomendación de los meses con mayores ventas requerirían una empresa con un mayor grado de maduración y trayectoria o por lo menos datos de alguna otra tenga el mismo giro para determinar los meses con mayores ventas y preparar inventario para estas ocasiones. Por otro lado, podríamos actualmente con las ventas por años determinar el éxito de la empresa y analizar la efectividad de la toma de decisiones de ventas que se hicieron en esos meses. Me parece una buena de gerencia usar el análisis de datos para entender el problema en el que se encuentran sus ventas ya que ayuda mucho a ver la gravedad de la baja de ventas y en que punto estas bajaron: y a su vez ayudar a controlar los inventarios fuera de control.

En general creo que el análisis de datos es la mejor opción que la empresa tomo para analizar su problema sin embargo la visión con la que quieren analizar el problema se queda algo corta ya que el análisis requiere una mayor visión para poder determinar problemas más específicos que creo que serian los que se tendrían que cubrir en lugar de problemas tan generales que puede que no logren ayudar a estabilizar sus ventas nuevamente.