Reporte Proyecto 1

Análisis de ventas para la empresa Life Store

Zapata López, Juan Manuel 9/6/20 Beca Santander Tech

Índice

Introducción

LifeStore es una tienda virtual que maneja una amplia gama de artículos, recientemente, la Gerencia de ventas, se percató que la empresa tiene una importante acumulación de inventario. Asimismo, se ha identificado una reducción en las búsquedas de un grupo importante de productos, lo que ha redundado en una disminución sustancial de sus ventas del último trimestre.

El presente trabajo busca crear una estrategia para evitar la acumulación de inventario y si ya está ocurriendo, proponer estrategias para vender los productos.

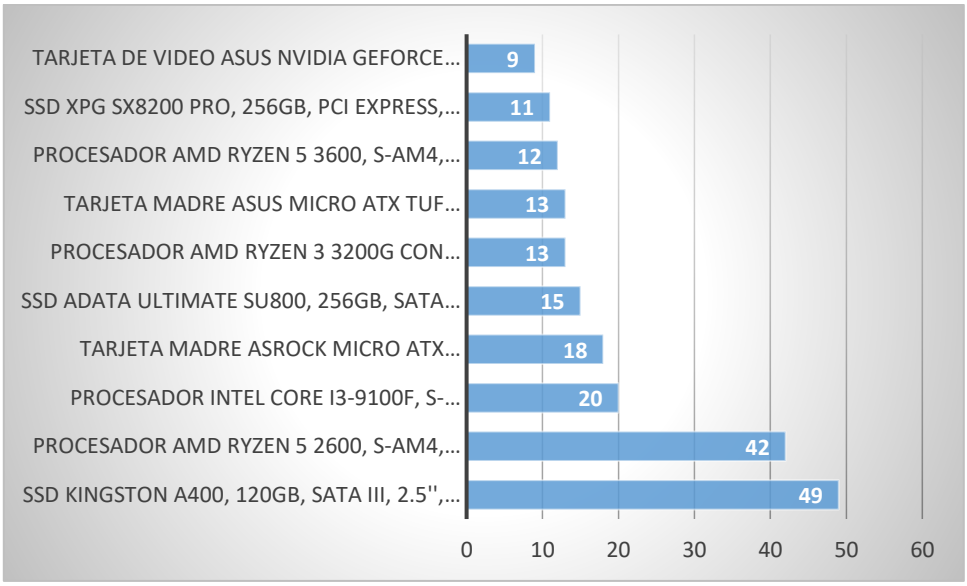
Esto se logró mediante el análisis de los datos proporcionados por la compañía, que se analizaron en Python y cuyos resultados se copiaron a Excel para su graficación.

El objetivo final de este análisis es aumentar las ventas de los próximos meses.

Solución al problema

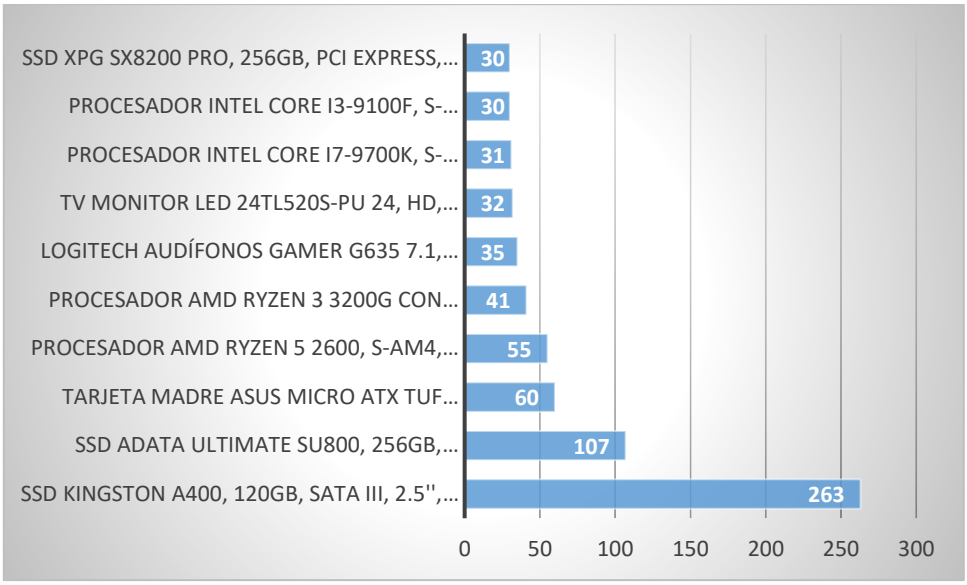
A continuación se presentan las gráficas de las tablas generadas en Python (las gráficas se hicieron en Excel). Para las tablas con más de 10 datos, sólo se presentaran los primeros 10 registros, si se desea consultar las tablas completas se puede consultar el archivo de Excel anexo o correr el programa de Python.

Productos con más ventas



De todos los datos, podemos notar que el producto SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm es el que tiene más unidades vendidas. De la gráfica podemos concluir que los discos duros, tarjetas madres y procesadores son los productos que generan más ventas. Se recomienda seguir comprando estos productos para su posterior venta.

Productos con más búsquedas

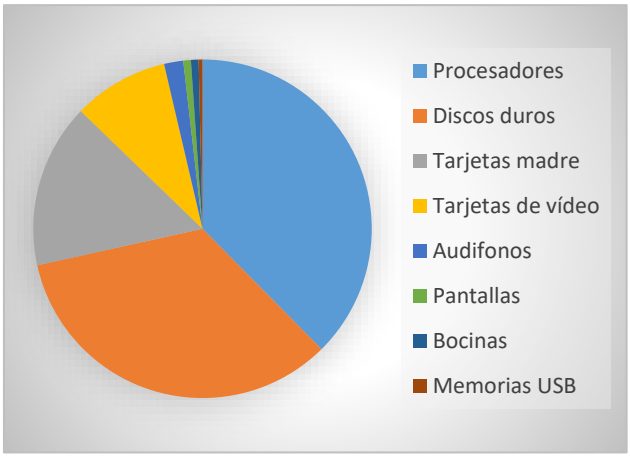


De esta gráfica podemos concluir que los usuarios que visitan el sitio web de la empresa, están mayormente interesados en adquirir SSDs, tarjetas madres y procesadores. Se recomienda continuar con la venta de estos artículos y si es posible, aumentar el nivel de inventario para ellos.

Productos con menos ventas por categoría

El resumen de la información generada es el siguiente:

Categoría	total de unidades vendidas
Procesadores	103
Discos duros	93
Tarjetas madre	43
Tarjetas de vídeo	25
Audifonos	5
Pantallas	2
Bocinas	2
Memorias USB	1



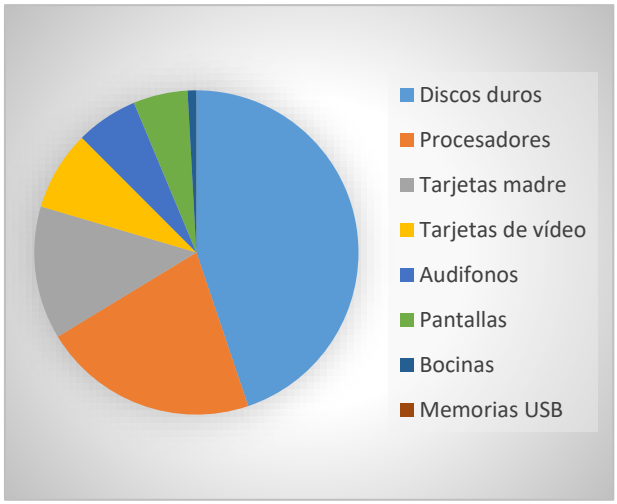
Podemos notar que más de ¾ partes del total de ventas están siendo generadas sólo por las categorías de procesadores, discos duros y tarjetas madres. Se recomienda mantener la estrategia y si es posible aumentar el inventario para los productos de estas categorías.

Por el contrario, las categorías menos vendidas son audífonos, pantallas, bocinas y memoria usb. Para ellos, se recomienda iniciar una actividad promocional consistente en rebajas graduales de forma mensual hasta agotar las unidades y no volver a adquirir inventarios para ellas.

Mes	Descuento
1	10%
2	20%
3	30%
4	40%

Productos con menos búsquedas

Categoría	Total de búsquedas
Discos duros	463
Procesadores	222
Tarjetas madre	137
Tarjetas de vídeo	82
Audifonos	64
Pantallas	56
Bocinas	9
Memorias USB	0

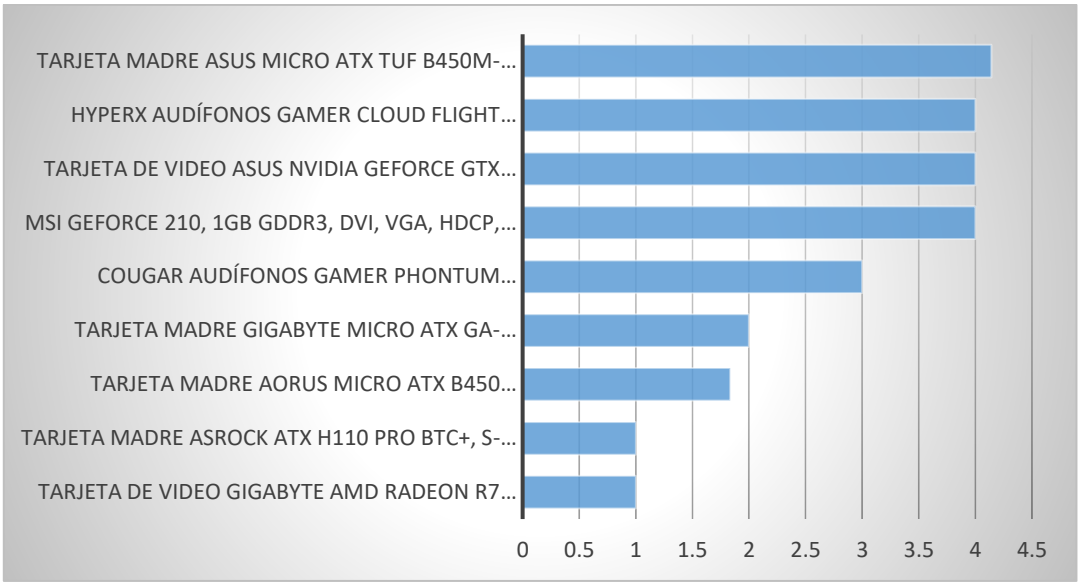


Se observa la misma tendencia que la del análisis previo, por lo que se refuerza la misma conclusión. Adicionalmente, se nota que los audífonos y pantallas si generan búsquedas, pero dado su bajo nivel de ventas, se deben promocionar.

Reseñas de productos

Sobre los productos con mejores reseñas, se observa que en general tienen buenos scores. Se recomienda aumentar el inventario de ellos.

Sobre los productos con peores reseñas, veamos el gráfico:



Se recomienda que para los productos cuyo promedio de score sean menor a tres, se deje de comercializar ese producto en específico, siguiendo la misma estrategia de descuentos por mes mencionada arriba, con la diferencia de que en su lugar, se compren productos de la misma categoría pero de las marcas mejor puntuadas según los usuarios.

Ventas

Para los análisis de ventas, se decidió calcular las ventas unidad (como el total de unidades vendidas) y las ventas valor (que toma en cuenta el precio al que se vendió el producto): Los resultados ordenados por cada tipo de venta son los siguientes:

Ventas unidad de mayor a menor

MES	VENTAS UNIDAD	VENTAS VALOR
202004	75	193295
202001	53	120237
202003	51	164729
202002	41	110139
202005	35	96135
202006	11	36949
202007	11	26949
202008	3	3077
200205	1	259
201911	1	4209
202009	1	4199

Ventas valor de mayor a menor

MES	VENTAS UNIDAD	VENTAS VALOR
202004	75	193295
202003	51	164729
202001	53	120237
202002	41	110139
202005	35	96135
202006	11	36949
202007	11	26949
201911	1	4209
202009	1	4199
202008	3	3077
200205	1	259

De ambos resultados, se concluye que en los primeros 4 meses de este año se han tenido los mejores resultados.

Ventas anuales

Finalmente, las ventas por año han sido las siguientes:

AÑO	VENTAS UNIDAD	VENTAS VALOR
2020	281	755709
2019	1	4209
2002	1	259

Con lo que se concluye que seguramente hubo un error de captura para el año 2002 y que para el 2019 la tienda estaba en periodo de pruebas o algo similar y que durante 2020 se aperturó con éxito.

Conclusiones

- Al parecer la tienda LifeStore inició su incursión en el mercado desde el primer minuto del 2020.

- Se observa que los primeros meses de su operación fueron los mejores en ventas.

- Para incrementar sus ventas, se deben incrementar el inventario de las categorías Procesadores, discos duros y tarjetas madres; preferentemente con artículos que se sabe, han tenido las mejores reseñas.

- Para reducir los costos y maximizar el margen de ganancias, se debe eliminar gradualmente la adquisición de productos de las categorías Memorias USB, pantallas, audífonos y bocinas.

- Para lograr sacar del inventario los productos anteriores, se debe iniciar una campaña de descuentos con aumento gradual al mes.

Anexo. Código.

El código se puede descargar desde https://github.com/manuelzapata04/ProyectoEmtech01/

```
# -*- coding: utf-8 -*-
"""
Created on Sat Sep  5 11:16:00 2020

@author: zaju9001
"""
import operator
from tabulate import tabulate
from lifestore_file import lifestore_products
from lifestore_file import lifestore_sales
from lifestore_file import lifestore_searches



# users = [id_user, password, type]
users_relation = [["Manu", "2174", "admin"], ["Mauricio", "holamundo", "normal"] , ["Guacho", "soyunperrito", "normal"]  ]
print("Bienvenido a la plataforma LifeStore Analytics\nSi desea ver los reportes ingrese un usuario normal, si desea modificar los usuarios ingrese como administrador")


#Acceso
while True:
    user = input("Ingrese su usuario: ")
    found = False
    for user_list in users_relation:
        if user_list[0] == user:
            found = True
```

```
            password = user_list[1]
            user_type = user_list[2]
            break
    if(found):
        passw = input("Usuario encontrado, por favor ingrese su contraseña: ")
        if passw == password:
            print("\nAcceso correcto")
            break
        else:
            print("Contraseña incorrecta, por favor vuelva a ingresar")
    else:
        print("Usuario no encontrado, por favor ingrese un usuario válido o comuniquese con el
administrador")

print("¡Bienvenido ",user,"!\n")

if user_type == "normal":
    #Calcula las ventas por producto, sin tomar en cuenta los productos que fueron regresados

    producto_ventas = []
    for producto in lifestore_products:
        contador = 0
        for ventas in lifestore_sales:
            if producto[0]==ventas[1] and ventas[4] == 0:
                contador += 1
        formato_ideal = producto + [contador]
        producto_ventas.append(formato_ideal)

    #ordena los productos por ventas, de mayor a menor
```

```
    producto_ventas_orden  = sorted(producto_ventas, key=operator.itemgetter(5), reverse = True)


    #imprime los 50 productos con más ventas, toma en cuenta sólo los que tiene ventas mayor a
cero

print("""\n**********************************************************************
******************************
    50 productos con más ventas

**********************************************************************************
**********************
        """)
    contador = 0
    lista_actual = []
    for producto in producto_ventas_orden:
        if producto[5] > 0 and contador < 50:
            lista_actual.append([producto[0], producto[1], producto[5]])
            contador += 1
        else:
            print("\nNo se pueden mostrar 50 productos, sólo existen ", contador, " productos con
ventas, del resto no se ha vendido nada.\n")
            break
    print(tabulate(lista_actual, headers = ["ID_PRODUCTO", "PRODUCTO", "VENTAS"]))


    #Calcula el número de búsquedas por producto


    producto_busquedas = []
    for producto in lifestore_products:
        contador = 0
        for busquedas in lifestore_searches:
            if producto[0] == busquedas[1]:
```

```
            contador += 1
        formato_ideal = producto + [contador]
        producto_busquedas.append(formato_ideal)

    #ordena los productos por búsquedas, de mayor a menor
    producto_busquedas_orden  = sorted(producto_busquedas, key=operator.itemgetter(5), reverse = True)

    #imprime los 100 productos con más búsquedas, toma en cuenta sólo productos que han sido buscados al menos una vez

print("""\n**********************************************************************************************************
    100 productos con más busquedas

**********************************************************************************************************
        """)

    contador = 0
    lista_actual = []
    for producto in producto_busquedas_orden:
        if producto[5] > 0 and contador < 100:
            lista_actual.append([producto[0], producto[1], producto[5]])
            contador += 1
        else:
            print("\nNo se pueden mostrar 100 productos, sólo existen ", contador, " productos que han sido buscados\n")
            break
    print(tabulate(lista_actual, headers = ["ID_PRODUCTO", "PRODUCTO", "N BUSQUEDAS"]))
```

```
    ##genera una lista con las categorías disponibles
    categorias = []
    for producto in lifestore_products:
        categoria_actual = producto[3]
        if categoria_actual not in categorias:
            categorias.append(categoria_actual)

    #genera el informe con los 50 productos con menos ventas por categoría

print("""\n**********************************************************************
******************************
    50 productos con menos ventas por categoría
    Observación: Si la categoría tiene menos de 50 productos, sólo se mostraran los disponibles

**************************************************************************
**********************
        """)
    producto_ventas_orden_inverso = sorted(producto_ventas, key=operator.itemgetter(5))

    for categoria in categorias:
        contador = 0
        lista_actual = []

        for ventas in producto_ventas_orden_inverso:
            if categoria == ventas[3] and contador < 50:
                lista_actual.append([ventas[0], ventas[1], ventas[5]])
                contador +=1
        print("\nlos " ,contador," productos con menos ventas para la categoría ", categoria, " son:")
        print(tabulate(lista_actual, headers=['id', 'product', 'ventas']))
```

```
    #genera el informe con los 100 productos con menores búsquedas por categoría

print("""\n**********************************************************************
******************************

    100 productos con menos búsquedas por categoría

    Observación: Si la categoría tiene menos de 100 productos, sólo se mostraran los disponibles

**********************************************************************************
**********************

        """)

    producto_busquedas_orden_inverso = sorted(producto_busquedas,
key=operator.itemgetter(5))


    for categoria in categorias:

        contador = 0

        lista_actual = []


        for busquedas in producto_busquedas_orden_inverso:

            if categoria == busquedas[3] and contador < 50:

                lista_actual.append([busquedas[0], busquedas[1], busquedas[5]])

                contador +=1

        print("\nlos " ,contador," productos con menos busquedas para la categoría ", categoria, "
son:")

        print(tabulate(lista_actual, headers=['id', 'product', 'busquedas']))



    ####Cálculo de reseñas por producto

    #Calcula el promedio del score por producto


    producto_score = []
```

```
    for producto in lifestore_products:
        contador = 0
        score_total = 0
        for score in lifestore_sales:
            if producto[0]==score[1]:
                contador += 1
                score_total += score[2]
        formato_ideal = producto + [contador] + [score_total]
        producto_score.append(formato_ideal)

    producto_score_final = []
    for score in producto_score:
        if score[5] > 0:
            producto_score_final.append([score[0],score[1] ,score[6]/score[5]])



    #ordena los productos por score de mayor a menor
    producto_score_orden  = sorted(producto_score_final, key=operator.itemgetter(2), reverse =
True)



    #ordena los productos por score de menor a mayor
    producto_score_orden2  = sorted(producto_score_final, key=operator.itemgetter(2))

    #imprime los 20 productos con mejor score

print("""\n**********************************************************************
*******
    20 productos con las mejores reseñas
    Observación, si hay menos de 20 productos con reseñas, sólo se mostrarán los disponibles
```

```
**********************************************************************************
    """)

contador = 0
lista_actual = []
for score in producto_score_orden:
    if contador < 20:
        lista_actual.append(score)
        contador += 1
    else:
        break


print("\nlos " ,contador," productos con mejores reseñas son:")
print(tabulate(lista_actual, headers = ["ID_PRODUCTO", "PRODUCTO", "SCORE_PROMEDIO"]))


#imprime los 20 productos con peor score

print("""\n**********************************************************************************
*******
    20 productos con las peores reseñas
    Observación, si hay menos de 20 productos con reseñas, sólo se mostrarán los disponibles
    **********************************************************************************
    """)

contador = 0
lista_actual = []
for score in producto_score_orden2:
    if contador < 20:
```

```
        lista_actual.append(score)
        contador += 1
    else:
        break


print("\nlos " ,contador," productos con peores reseñas son:")
print(tabulate(lista_actual, headers = ["ID_PRODUCTO", "PRODUCTO", "SCORE_PROMEDIO"]))


### añade variables de mes y año a lifestore_sales, también añade el precio del producto
for ventas in lifestore_sales:
    mes =ventas[3][-4:] + ventas[3][3:5]
    anio = ventas[3][-4:]
    ventas.append(mes)
    ventas.append(anio)
    for producto in lifestore_products:
        if ventas[1] == producto[0]:
            ventas.append(producto[2])

#lifestore-sales = [id_sale, id_product, score, date, refund, mes_año, anio, precio]

##genera una lista con las distintos meses_año
keys = []
for venta in lifestore_sales:
    key_actual = venta[5]
    if key_actual not in keys:
        keys.append(key_actual)
```

```
    #trae las ventas valor y unidad por mes
    resumen_ventas_mes = []
    for key in keys:
        ventas_unidad = 0
        ventas_valor = 0
        for ventas in lifestore_sales:
            if key == ventas[5]:
                ventas_unidad += 1
                ventas_valor += ventas[7]
        resumen_ventas_mes.append([key, ventas_unidad, ventas_valor])



print("""\n**********************************************************************
*******
    A continuación se presenta la información de ventas resumida por mes:
    VENTAS UNIDAD = número de productos vendidos en el mes
    VENTAS VALOR = total de ingresos por todos los productos vendidos
    **********************************************************************
        """)

    resumen_ventas_mes = sorted(resumen_ventas_mes, key=operator.itemgetter(0))
    print(tabulate(resumen_ventas_mes, headers = ["MES", "VENTAS UNIDAD", "VENTAS VALOR"]))



    #calcula el promedio de ventas para los meses disponibles
    num_meses = 0
    ventas_valor_mes = 0
```

```
    ventas_unidad_mes = 0
    for ventas in resumen_ventas_mes:
        num_meses += 1
        ventas_unidad_mes += ventas[1]
        ventas_valor_mes += ventas[2]

    prom_ventas_unidad = ventas_unidad_mes / num_meses
    prom_ventas_valor = ventas_valor_mes / num_meses

    print("tomando en cuenta todos los meses de ventas sin impotar el año:\nel promedio de ventas unidad es: ",
          round(prom_ventas_unidad, 2), "\nel promedio de ventas valor es ", round(prom_ventas_valor, 2))

    print("\nSi ordenas la tabla de ventas mensuales por ventas unidad, podemos notar que Abril del 2020 es el mes con más ventas unidad, seguido de Enero y Marzo del mismo año: ")
    resumen_ventas_mes = sorted(resumen_ventas_mes, key=operator.itemgetter(1), reverse = True)
    print(tabulate(resumen_ventas_mes, headers = ["MES", "VENTAS UNIDAD", "VENTAS VALOR"]))

    print("\nSi ordenas la tabla de ventas mensuales por ventas valor, podemos notar que Abril del 2020 es el mes con más ventas unidad, seguido de Marzo y Enero del mismo año: ")
    resumen_ventas_mes = sorted(resumen_ventas_mes, key=operator.itemgetter(2), reverse = True)
    print(tabulate(resumen_ventas_mes, headers = ["MES", "VENTAS UNIDAD", "VENTAS VALOR"]))
```

```
print("""\n**********************************************************************
*******
   A continuación se presenta la información de ventas resumida por año:
   VENTAS UNIDAD = número de productos vendidos en el mes
   VENTAS VALOR = total de ingresos por todos los productos vendidos
   **********************************************************************
       """)

   ##genera una lista con las distintos años
   anios = []
   for venta in lifestore_sales:
      key_actual = venta[6]
      if key_actual not in anios:
         anios.append(key_actual)



   resumen_ventas_anio = []
   for key in anios:
      ventas_unidad = 0
      ventas_valor = 0
      for ventas in lifestore_sales:
         if key == ventas[6]:
            ventas_unidad += 1
            ventas_valor += ventas[7]
      resumen_ventas_anio.append([key, ventas_unidad, ventas_valor])

   print(tabulate(resumen_ventas_anio, headers = ["AÑO", "VENTAS UNIDAD", "VENTAS VALOR"]))


   print("Todos los reportes han sido generados, hasta la próxima !!!!!!")
```

```
else:
  opcion = input("Eliga la opción deseada\n1 Ingresar usuario nuevo \n2 Eliminar usuario\n")
  if opcion == '1':

    usuarios_existentes = []
    for user in users_relation:
      usuarios_existentes.append(user[0])

    condicion = True

    while condicion:
      new_user = input("Ingrese nombre de usuario: ")
      if new_user in usuarios_existentes:
        print("usuario en uso, por favor ingresa otro\n")
      else:
        condicion = False

    condicion = True
    while condicion:
      new_pass = input("Ingrese contraseña de al menos 4 dígitos: ")
      if len(new_pass) < 4:
        print("Contraseña muy corta, vuelve a intentar\n")
      else:
        condicion = False

    condicion = True
    while condicion:
      new_type = input("Ingrese tipo de usuario [admin / normal]: ")
```

```
        if new_type not in ["admin", "normal"]:
            print("Tipo no válido, vuelve a intentar\n")
        else:
            condicion = False

    users_relation.append([new_user, new_pass, new_pass])
    print("Usuario añadido con éxito\n Hasta la próxima!!!!")

elif opcion == '2':
    usuarios_existentes = []
    for user in users_relation:
        usuarios_existentes.append(user[0])

    condicion = True

    while condicion:
        old_user = input("Ingrese nombre de usuario a eliminar: ")
        if old_user not in usuarios_existentes:
            print("usuario no registrado, por favor ingresa otro\n")
        else:
            condicion = False

    contador = 0
    for user in users_relation:
        if user[0] != old_user:
            contador += 1
        else:
            break
    usuarios_existentes.pop(contador)
```

```
    print("Usuario eliminado con éxito, hasta la próxima")

else:
    print("Opción no válida, hasta la próxima !!!!!")
```