EMTECH

Emerging Technologies Institute

PROYECTO Nº 1 VENTAS DE LIFESTORE

MORENO HERNANDEZ MARIA FERNANDA

12-02-2022

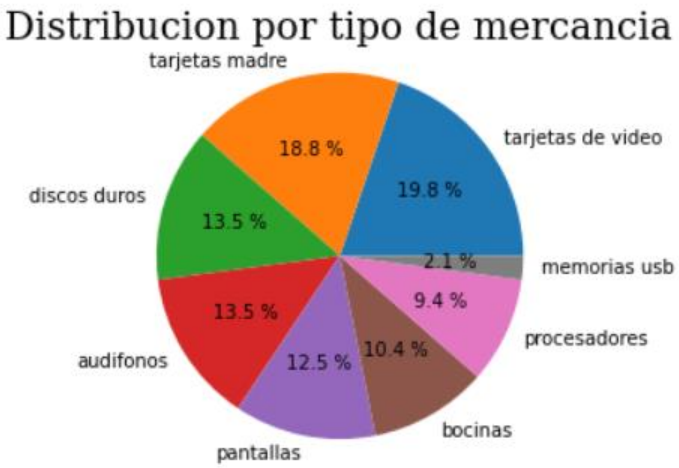
EMTECH

CONTENIDO

1. INTRODUCCION

Este proyecto tiene como objetivo el análisis y la clasificación de datos referente a la base de datos (ventas, búsquedas, stock y reseñas) que obtuvo una empresa respecto a sus productos a lo largo de un tiempo. Para ello se uso la paquetería de PANDAS para manipular la información, así mismo se usará DataFrame para la estructura de tablas.

La siguiente tabla se obtuvo de los datos sobre las ventas que obtuvo la empresa LifeStore.



2. PRODUCTOS MAS VENDIDOS Y PRODUCTOS MAS REZAGADOS

Como se usara la paquetería de pandas vamos a importarla como pd, es decir : import pandas as pd

Tenemos la lista de la empresa LifeStore donde:

- lifestore_searches = [id_search, id product] …. Búsquedas con el id de búsqueda y id de producto
- lifestore_sales = [id_sale, id_product, score (from 1 to 5), date, refund (1 for true or 0 to fal se)] …. Es el id de la venta del producto, la calificación o reseña de 1 a 5 que obtuvo ese producto, así como la fecha y si tuvo devolución será 1 y 0 si no se devolvía.
- lifestore_products = [id_product, name, price, category, stock] … son los productos que tienen una descripción del id del producto, nombre, precio, categoría y su stock

Crearemos DataFrame a partir de las listas que tenemos:

```
Productos=pd.DataFrame(lifestore_products)
Ventas=pd.DataFrame(lifestore_sales)
Busquedas=pd.DataFrame(lifestore_searches)
```

Nombraremos las columnas usando la función:

```
Productos.columns=["id_product", "name", "price", "category", "stock"]

Ventas.columns=["id_sale", "id_product", "score (from 1 to 5)", "date", "refund (1 for true or 0 to fals
e)"]
Busquedas.columns=["id_search", "id_product"]
```

Usamos la función pandas head() la cual se utiliza para obtener las filas superiores de DataFrame. Cuando se usa el número negativo, devuelve todas excepto las últimas filas :

```
Productos.head()
Ventas.head()
Busquedas.head()
```

PRODUCTOS MAS VENDIDOS

La función. iloc[ ] se utiliza para acceder a todas las filas y columnas como una matriz booleana

```
Productos.iloc[:,1]
```

Aquí se cuenta con devolución:

```
Cantidad_de_ventas=Ventas.id_product.value_counts(dropna=False)
```

Solo contamos aquellos que no tuvieran devolución, value.counts es una función se utiliza para obtener una serie que contiene recuentos de valores únicos:

```
Cantidad_de_ventas=Ventas[Ventas["refund (1 for true or 0 to false)"]==0]["id_product"].value_co
unts()
Cantidad_de_ventas=pd.DataFrame(Cantidad_de_ventas)
```

Extraemos el index para hacer una columna, el idex proporciona un objeto de rango desde la fila superior a la fila inferior de nuestro DataFrame. Y usaremos .reset_index que es un método para restablecer el índice y establece una lista de enteros que van desde 0

```
Cantidad_de_ventas=Cantidad_de_ventas.reset_index()
```

Como nos pide el problema saber cuales son los 5 productos más vendidos entonces usaremos head es decir:

```
Top_5_mas_ventas=Cantidad_de_ventas.head(n=5)
```

Y así nos arrojara la tabla de las 5 mayores ventas que se obtuvieron en la empresa LifeStore

```
Top 5 de productos con mayores ventas
   id_product  ... N ventas
0          54  ...       49
1           3  ...       42
2           5  ...       20
3          42  ...       18
4          57  ...       15
```

Teniendo como resultado que:

1. El producto 54 es una SSD Kingston A400 con un total de 49ventas
2. El producto 3 es un Procesador AMD Ryzen con un total de 42 venas
3. El producto 5 Procesador Intel CORE i3 con un total de 20 venta
4. El producto 42 Tarjeta madre ASRock con un total de 18 ventas
5. El producto 20 SSD Adata con un total de 15 ventas

PRODUCTOS CON MAYORES BUSQUEDAS

Aquí se cuenta con devolución:

```
Cantidad_de_Busquedas=Busquedas.id_product.value_counts(dropna=False)

Cantidad_de_Busquedas=pd.DataFrame(Cantidad_de_Busquedas)
```

Extraemos el index para hacer una columna y se reinicia:

```
Cantidad_de_Busquedas=Cantidad_de_Busquedas.reset_index()
```

Usaremos merge este Se puede usar para combinar Dataframes dependiendo del nombre de columna o índice, pero el nombre de columna pasado o el nivel de índice deben estar presentes en ambos DataFrames.

```
Cantidad_de_Busquedas=Cantidad_de_Busquedas.merge(Productos[["id_product","name"]],on='id_product')
```

Ahora bien, el problema nos indica obtener los 10 productos con mayores búsquedas nuevamente usaremos head ahora con n=10

```
Top_n_mayores_busquedas=Cantidad_de_Busquedas.head(n=10)
```

Y así nos arrojara la tabla de los 10 productos con mayor número de búsquedas que se obtuvieron en la empresa LifeStore

```
Top 10 productos con mayor número de búsquedas

   id_product  N busquedas                                              name
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
```

PRODUCTOS CON MENORES VENTAS POR CATEGORIA

Como nos pide el problema saber cuáles son los 5 productos menos vendidos entonces usaremos tail y este se utiliza para devolver las filas inferiores.

```
Top_5_menos_ventas=Cantidad_de_ventas.tail(n=5)
```

Y así nos arrojara la tabla de la empresa LifeStore:

```
Top 5 productos con menores ventas con la categoría

discos duros
    id_product  N ventas  Venta_final  Stock_final
15          51         3         7197           -3
10          48         9        23031           41
8           47        11        13299           -3
4           57        15        13335            0
0           54        49        12691          251
```

Un dato curioso es que en dos productos no se contaba con stock y fueron vendidos, por ello es que sale negativo su stock, es decir que allí hubo un problema con el sistema de la empresa.

PRODUCTOS CON MENORES BUSQUEDAS

En este caso, haremos lo mismo que en el punto de 10 productos con mayores números de búsquedas, pero en esta ocasión utilizaremos tail.

```
Top_n_menores_busquedas=Cantidad_de_Busquedas.tail(n=10)
```

```
Top 10 productos de menores búsquedas

    id_product  N busquedas                                              name
46          76            2  Acteck Bocina con Subwoofer AXF-290, Bluetooth...
47           9            1  Procesador Intel Core i3-8100, S-1151, 3.60GHz...
48          93            1  Ginga Audífonos con Micrófono GI18ADJ01BT-RO, ...
49          35            1  Tarjeta Madre Gigabyte micro ATX Z390 M GAMING...
50          80            1  Ghia Bocina Portátil BX800, Bluetooth, Inalámb...
51          70            1  Samsung Smart TV LED 43, Full HD, Widescreen, ...
52          10            1  MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PC...
53          45            1  Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151...
54          27            1  Tarjeta de Video VisionTek AMD Radeon HD5450, ...
55          59            1            SSD Samsung 860 EVO, 1TB, SATA III, M.2
```

3. PRODUCTOS POR RESEÑA EN EL SERVICIO.

Los productos que tuvieron reseña: Resena=Ventas

La función unique identifica los valores únicos de una serie de Pandas:

```
Prod_resenas=Resena.id_product.unique()
Prod_resenas
```

Hacemos una nueva lista para guardar la información:

```
Resena_lista=[]
```

Se genera un ciclo de for para iterar en cada producto de igual forma:

```
for Prod_resena in Prod_resenas:
```

Vamos a filtrar por reseña y el id del producto, luego imprimiremos los datos por reseña

```
 Filtro_resena=Resena["id_product"]==Prod_resena

 Datos_por_resena_producto=Resena[Filtro_resena]

 print("El producto id ",Prod_resena)
```

Imprimimos pero con la puntuación que sea de 1 a 5 con value_counts() que es una forma rápida de ver cuántos valores diferentes hay en una columna de una tabla y calcular cuántos valores duplicados hay en esa columna para cada valor diferente. Y dropna = False significa que está reservado, dropna es true por defecto, es decir, no está incluido por defecto:

```
print(Datos_por_resena_producto["score (from 1 to 5)"].value_counts(dropna=False))

Resena_lista.append(Datos_por_resena_producto["score (from 1 to 5)"].value_counts(dropna=Fal
se))
```

Hacemos una lista nueva de calificación promedio:

```
Cal_promedio=[]
```

Usaremos un ciclo for y len que es para la longitud de la lista (para que vaya iterando conforme deseamos). El producto reseña es una lista que igual se ira iterando, vamos a transformarlo en un DataFrame, y vamos a reiniciar el index nuevamente:

```
for i in range(len(Resena_lista)):
print("El id del producto es",Prod_resenas[i])
Resena_producto=pd.DataFrame(Resena_lista[i]).reset_index()
```

Ahora queremos que nos arroje en columnas los nombres de cal y cantidad :

```
Resena_producto.columns=["Cal","Cantidad"]
producto=Resena_producto["Cal"]*Resena_producto["Cantidad"]
```

Sacamos la Media ponderada:

```
sum_producto=producto.sum()
sum_cal=Resena_producto["Cantidad"].sum()
```

Round se usa para redondear con los puntos decimales y le ponemos 2 para que aparezcan solo 2 decimales, y haremos la operación de la suma del producto entre la suma de las calificaciones:

```
Cal_promedio.append(round(sum_producto/sum_cal,2))
```

Nuevamente hacemos un DataFrame de la calificación promedio y agregamos una columna del id del producto de la reseña, le pondremos el nombre a la primera columna que será promedio de calificaciones y luego su id pero esta se tendrá que renombrar:

```
Cal_promedio=pd.DataFrame(Cal_promedio)
Cal_promedio["id_product"]=Prod_resenas
Cal_promedio.columns=["Promedio de calificacion","id_product"]
Cal_promedio=Cal_promedio[["id_product","Promedio de calificacion"]]
```

Uniremos la base de datos de productos con la base que acabamos de hacer, usando la función merge la primera tiene llave de Id producto. Ahora usamos drop para eliminar las columnas que se nos duplicaron al unir las dos bases y por consiguiente vamos a ordenarlo dependiendo de como deseemos vayan nombradas las columnas:

```
Cal_promedio=Cal_promedio.merge(Productos, left_on='id_product', right_on='id_product')
Cal_promedio=Cal_promedio.drop(columns=["stock","price"]).sort_values('Promedio de calificacio
n',ascending=False)
Cal_promedio=Cal_promedio[["id_product","name","category","Promedio de calificacion"]]
```

PRODUCTOS CON MEJOR RESEÑA

Como nos pide el problema saber cuáles son los 5 productos con mejor reseña entonces usaremos head es decir:

```
Cal_promedio.head(n=5)
```

	id_product	name	category	Promedio de calificacion
0	1	Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Q...	procesadores	5.0
14	21	Tarjeta de Video MSI AMD Mech Radeon RX 5500 X...	tarjetas de video	5.0
39	85	Logitech Audífonos Gamer G635 7.1, Alámbrico, ...	audifonos	5.0
38	84	Logitech Audífonos Gamer G332, Alámbrico, 2 Me...	audifonos	5.0
36	67	TV Monitor LED 24TL520S-PU 24, HD, Widescreen,...	pantallas	5.0

PRODUCTOS CON PEOR RESEÑA

Haremos lo mismo que el problema anterior pero en este caso usaremos tail es decir:

```
Cal_promedio.tail(n=5)
```

	id_product	name	category	Promedio de calificacion
40	89	Cougar Audífonos Gamer Phontum Essential, Alám...	audifonos	3.00
25	46	Tarjeta Madre Gigabyte micro ATX GA-H110M-DS2,...	tarjetas madre	2.00
19	31	Tarjeta Madre AORUS micro ATX B450 AORUS M (re...	tarjetas madre	1.83
12	17	Tarjeta de Video Gigabyte AMD Radeon R7 370 OC...	tarjetas de video	1.00
24	45	Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151...	tarjetas madre	1.00

4. TOTAL DE INGRESOS Y VENTAS PROMEDIO MENSUALES, TOTAL ANUAL Y MESES CON MÁS VENTAS AL AÑO.

TOTAL, DE INGRESOS Y VENTAS MENSUALES

Para las ventas por mes primero se actualiza como un dato tipo fecha utilizando to datetime este es una Serie de objetos de fecha y hora:

```
Ventas.date=pd.to_datetime(Ventas.date,infer_datetime_format=True)
```

Sacamos las ventas por mes, de igual forma se puede por día pero solo se nos solicita por mes por eso va con # y hacemos nuestro DataFrame de ambas.

```
Ventas["Mes"]=pd.DatetimeIndex(Ventas.date).month
#Ventas["Dia"]=pd.DatetimeIndex(Ventas.date).day
```

```
Decimos que:
i=1
```

```
suma=0
```

Hacemos una Lista que guardara las ventas de cada mes:

```
Lista=[]
```

```
for i in range(12):
```

```
 i=1+i
```

Decimos que Filtro_mes sera aquel que cambien segun el mes:

```
Filtro_mes=pd.DatetimeIndex(Ventas.date).month==i
```

Aplicaremos el filtro con el for en Ventas_mes:

```
Ventas_mes=Ventas[pd.DatetimeIndex(Ventas.date).month==i]
```

Contaremos las ventas según el producto:

```
Cantidad_de_ventas_mensuales=Ventas_mes[Ventas_mes["refund (1 for true or 0 to false)"]==0]
["id_product"].value_counts(dropna=False)
```

```
Cantidad_de_ventas_mensuales=pd.DataFrame(Cantidad_de_ventas_mensuales)
```

Extraemos el index para hacer una columna:

```
Cantidad_de_ventas_mensuales=Cantidad_de_ventas_mensuales.reset_index()
```

```
Cantidad_de_ventas_mensuales.columns=["id_product","N ventas"]
```

Unimos la base de ventas mensuales con la de productos para asi extraer el costo:

```
Cantidad_de_ventas_mensuales=Cantidad_de_ventas_mensuales.merge(Productos, left_on='id_product', right_on='id_product')
Cantidad_de_ventas_mensuales["Venta_final"]=Cantidad_de_ventas_mensuales.price*Cantidad_de_ventas_mensuales["N ventas"]
print("Mes",i)
print(Cantidad_de_ventas_mensuales["Venta_final"].sum())
Lista.append([Cantidad_de_ventas_mensuales["Venta_final"].sum(),Cantidad_de_ventas_mensuales["N ventas"].sum()])
```

Ahora hacemos una nuevo DataFrame:

```
meses_ventas=pd.DataFrame(Lista)
```

Vamos a ordenar los meses con sus ventas y cantidad de productos:

```
meses_ventas.columns=["Ventas","Cantidad_de_produtos"]
meses_ventas.index=["Enero",
"Febrero",
"Marzo",
"Abril",
"Mayo",
"Junio",
"Julio",
"Agosto",
"Septiembre",
"Octubre",
"Noviembre",
"Diciembre"]
meses_ventas["Venta_promedio_producto"]=round(meses_ventas.Ventas/meses_ventas.Cantidad_de_produtos,2)
```

Imprimiremos la suma de las ventas mensuales:

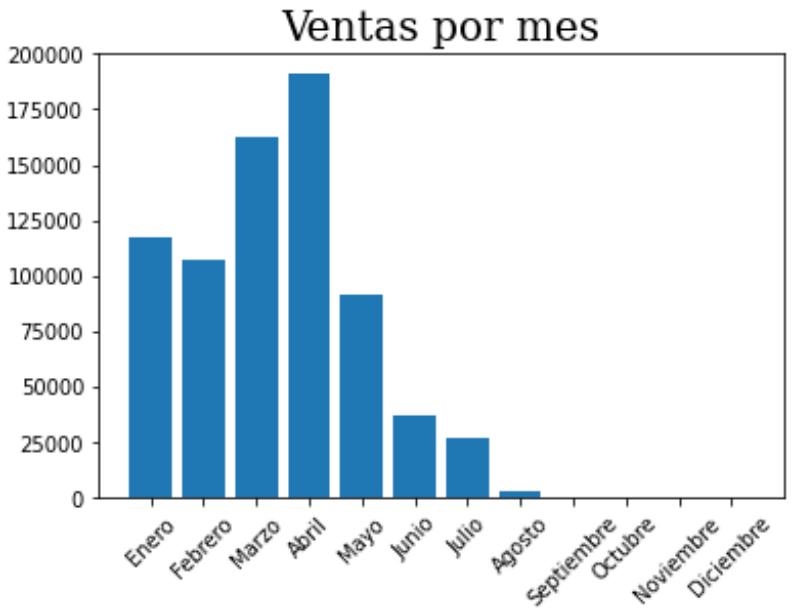
```
print("Ventas mensuales son:\n\n",meses_ventas)
```

```
Ventas mensuales son:

             Ventas  Cantidad_de_produtos  Venta_promedio_producto
Enero        117738                    52                  2264.19
Febrero      107270                    40                  2681.75
Marzo        162931                    49                  3325.12
Abril        191066                    74                  2581.97
Mayo          91936                    34                  2704.00
Junio         36949                    11                  3359.00
Julio         26949                    11                  2449.91
Agosto         3077                     3                  1025.67
Septiembre        0                     0                      NaN
Octubre           0                     0                      NaN
Noviembre         0                     0                      NaN
Diciembre         0                     0                      NaN
```

Haremos una tabla de las ventas por mes:

```
def plot_ventas():
  plt.bar(meses_ventas.index,meses_ventas.Ventas)
  font1 = {'family':'serif','color':'black','size':20}
  plt.title('Ventas por mes',fontdict = font1)
  plt.xticks(rotation=45)
  return(plt.show())
```



TOTAL, ANUAL

Con el código anterior solo imprimiremos la venta anual, que es la suma de las ventas mensuales:

```
print("\nVentas anuales son:",meses_ventas.Ventas.sum())
```

```
Total anual

Ventas anuales son:

 737916
```

MESES CON MAYORES VENTAS

Usaremos la tabla de meses con mayores ventas:

```
print("Meses con mayores ventas\n")
```

la ordenaremos de mayor a menor:

```
print("Ventas mensuales son:\n\n",meses_ventas.sort_values('Ventas',ascending=False))
```

	Ventas	Cantidad_de_produtos	Venta_promedio_producto
Abril	191066	74	2581.97
Marzo	162931	49	3325.12
Enero	117738	52	2264.19
Febrero	107270	40	2681.75
Mayo	91936	34	2704.00
Junio	36949	11	3359.00
Julio	26949	11	2449.91
Agosto	3077	3	1025.67
Septiembre	0	0	NaN
Octubre	0	0	NaN
Noviembre	0	0	NaN
Diciembre	0	0	NaN

5. CONCLUSION

La empresa de LifeStore tiene una variedad en cuanto a sus productos, teniendo en cuenta que algunos meses tienen mas ventas que otros, como nos dimos cuenta de que varios meses no tuvieron. Así mismo que noviembre pertenecía al año de 2019, así como varios productos no cuentan con stock sin embargo los siguen vendiendo, teniendo como consecuencia un stock negativo. Como en toda empresa también cuenta con devoluciones en este caso únicamente hubo 9 en todo este tiempo.

6. ANEXO

El usuario para ingresar al menú es: Fher

Contraseña= contrasena