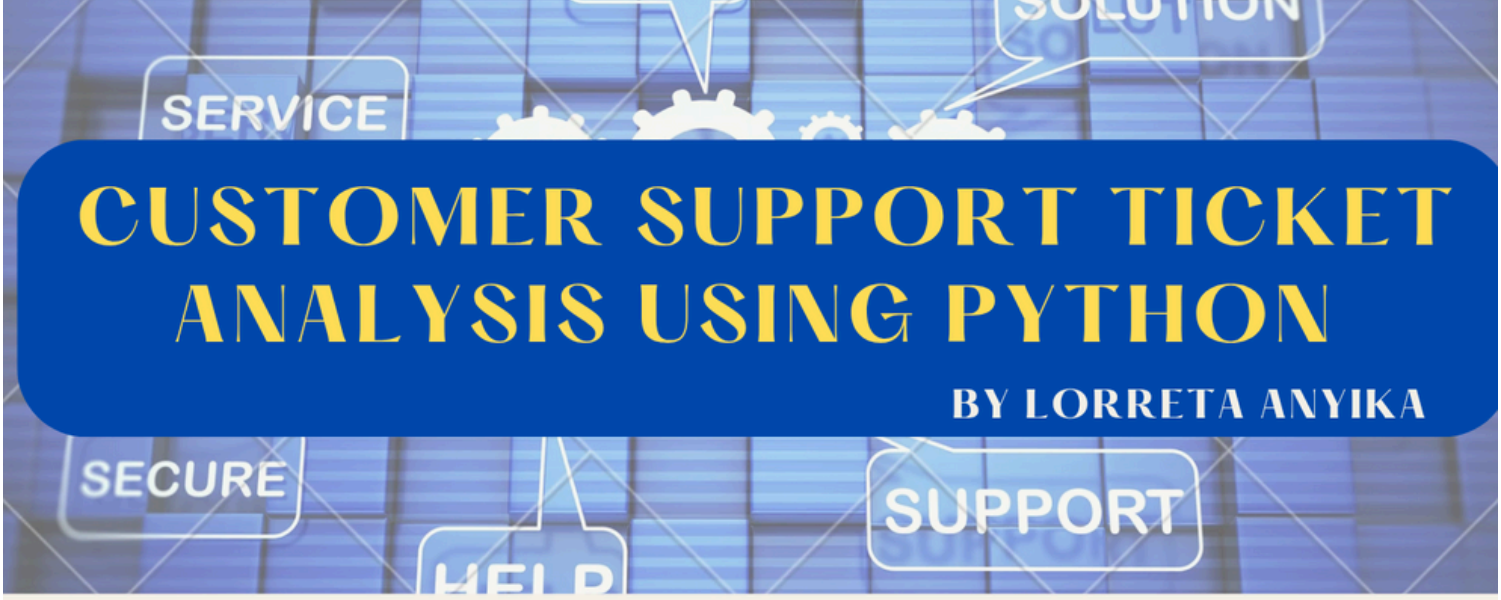


CUSTOMER SUPPORT TICKET ANALYSIS USING PYTHON

BY LORRETA ANYIKA

DATASET

Customer support ticket: 8469 tickets

TOOLS & SKILLS

Tools Used: Python, Pandas, Seaborn, matplotlib

Skills Applied: Data wrangling, formatting, type conversion, EDA

KEY OBJECTIVES

- Clean and preprocess raw customer support ticket data for accurate analysis using pandas
- Extract and engineer meaningful date time features from date of purchase
- Handle missing and inconsistency values
- Calculate key KPIs such as customer satisfaction resolution delay, first response delay, tickets by recency of purchase
- Segment customers based on gender, age, rating and correlation between all three
- Spot operational inefficiencies in support system
- Identify recurring pain points of customer
- Deliver a clean, exportable dataset ready for Visualization and BI reporting

Customer Insight analysis

May 27, 2025

[1]:
```
import pandas as pd
```

1 Import Dataset

[3]:
```
customer_ticket = pd.read_csv(r'C:\Users\user\Downloads\customer_support_tickets.csv')
```

2 Initial Exploration

[23]:
```
#extract first 5 rows
customer_ticket.head(5)
```

[23]:

	TicketID	CustomerName	CustomerEmail	CustomerAge	\
0	1	Marisa Obrien	carrollallison@example.com	32	
1	2	JessicaRios	clarkeashley@example.com	42	
2	3	Christopher Robbins	gonzalestracy@example.com	48	
3	4	Christina Dillon	bradleyolson@example.org	27	
4	5	Alexander Carroll	bradleymark@example.com	67	

	Customer Gender	Product Purchased	Date of Purchase	Ticket Type	\
0	Other	GoProHero	2021-03-22	Technicalissue	
1	Female	LGSmart TV	2021-05-22	Technicalissue	
2	Other	DellXPS	2020-07-14	Technicalissue	
3	Female	Microsoft Office	2020-11-13	Billinginquiry	
4	Female	AutodeskAutoCAD	2020-02-04	Billinginquiry	

	Ticket Subject	\
0	Productsetup	
1	Peripheral compatibility	
2	Networkproblem	
3	Accountaccess	
4	Dataloss	

	Ticket Description	\
0	I'm having an issue with the {product_purchase...	
1	I'm having an issue with the {product_purchase...	

2 I'm facing a problem with my {product_purchase...
3 I'm having an issue with the {product_purchase... I'm
4 having an issue with the {product_purchase...

	Ticket	Status		Resolution	\
0	PendingCustomer	Response		NaN	
1	PendingCustomer	Response		NaN	
2		Closed	Casemaybe	showrecentlymycomputerfollow.	
3		Closed	Try capital	clearlynevercolortowardstory.	
4		Closed		Westdecisionevidencebit.	

	Ticket Priority	Ticket Channel	First Response Time	Time to Resolution	\
0	Critical	Social media	6/1/202312:15	NaN	
1	Critical	Social Chat	6/1/202316:45	NaN	
2	Low	media	6/1/202311:14	6/1/202318:05	
3	Low	media	6/1/20237:29	6/1/20231:57	
4	Low	Email	6/1/20230:12	6/1/202319:53	

	Customer Satisfaction Rating
0	NaN
1	NaN
2	3.0
3	3.0
4	1.0

[24]:
```
# Extract last 5 rows
customer_ticket.tail(5)
```

[24]:

	TicketID	CustomerName	CustomerEmail	CustomerAge	\
8464	8465	David Todd	adam28@example.net	22	
8465	8466	Lori Davis	russell68@example.com	27	
8466	8467	Michelle Kelley	ashley83@example.org	57	
8467	8468	Steven Rodriguez	fpowell@example.org	54	
8468	8469	Steven Davis MD	lori20@example.net	53	

	CustomerGender	Product Purchased	DateofPurchase	\
8464	Female	OLED TV	2021-12-08	
8465	Female	Bose SoundLink Speaker	2020-02-22	
8466	Female	GoPro ActionCamera	2021-08-17	
8467	Male	PlayStation	2021-10-16	
8468	Other	Philips Hue Lights	2020-06-01	

	TicketType	TicketSubject	\
8464	Product inquiry	Installation support	
8465	Technical issue	Refundrequest	
8466	Technical issue	Accountaccess	
8467	Product inquiry	Payment issue	

```
8468  Billing    inquiry              Hardwareissue

                                                  Ticket Description Ticket Status  \
8464  My{product_purchased} ismaking      strangenoise...          Open
8465  I'mhaving   an issue  with  the  {product_purchase...          Open
8466  I'mhaving   an issue  with  the  {product_purchase...        Closed
8467  I'mhaving   an issue  with  the  {product_purchase...        Closed
8468    There seems to be a hardware problem with my {...          Open

                                                  Resolution Ticket Priority Ticket Channel  \
8464                                                     NaN             Low          Phone
8465                                                     NaN        Critical          Email
8466  Eight  account  century  nature  kitchen.             High   Socialmedia
8467                    Weseat  culture   plan.             Medium          Email
8468                                                     NaN            High          Phone

     First Response Time Time to Resolution  Customer Satisfaction Rating
8464                 NaN                NaN                          NaN
8465                 NaN                NaN                          NaN
8466      6/1/2023 9:44    6/1/2023 4:31                              3.0
8467      6/1/202318:28    6/1/2023 5:32                              3.0
8468                 NaN                NaN                          NaN
```

[25]:
```
#How many rows and columns
customer_ticket.shape
```

[25]: (8469, 17)

[27]:
```
#Display the columns, number of records per column, datatypes of column
customer_ticket.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8469 entries, 0 to 8468
Data columns (total 17 columns):
 #   Column                        Non-NullCount    Dtype
---  ------                        --------------   -----
 0   Ticket ID                     8469non-null     int64
 1   Customer Name                 8469non-null     object
 2   Customer Email                8469non-null     object
 3   Customer Age                  8469non-null     int64
 4   Customer Gender               8469non-null     object
 5   Product Purchased             8469non-null     object
 6   Dateof Purchase               8469non-null     datetime64[ns]
 7   Ticket Type                   8469non-null     object
 8   Ticket Subject                8469non-null     object
 9   Ticket Description            8469non-null     object
 10  Ticket Status                 8469non-null     object
 11  Resolution                    2769non-null     object
```

```
 12  Ticket  Priority                    8469non-null     object
 13  Ticket  Channel                     8469non-null     object
 14  First   ResponseTime                5650non-null     object
 15  TimetoResolution                    2769non-null     object
 16  Customer Satisfaction Rating        2769 non-null    float64
dtypes: datetime64[ns](1), float64(1), int64(2), object(13)
memory usage: 1.1+ MB
```

2.0.1 observation: there are missing values

[29]:
```
#Summarization of data
customer_ticket.describe()
```

[29]:

	Ticket ID	CustomerAge	DateofPurchase	\
count	8469.000000	8469.000000		8469
mean	4235.000000	44.026804	2020-12-30	01:35:13.071201024
min	1.000000	18.000000		2020-01-0100:00:00
25%	2118.000000	31.000000		2020-07-0200:00:00
50%	4235.000000	44.000000		2020-12-3100:00:00
75%	6352.000000	57.000000		2021-07-0100:00:00
max	8469.000000	70.000000		2021-12-3000:00:00
std	2444.934048	15.296112		NaN

	Customer Satisfaction Rating
count	2769.000000
mean	2.991333
min	1.000000
25%	2.000000
50%	3.000000
75%	4.000000
max	5.000000
std	1.407016

3 HandlingMissingValues

[30]:
```
customer_ticket.isnull().sum()
```

[30]:

Ticket ID	0
Customer Name	0
Customer Email	0
Customer Age	0
Customer Gender	0
Product Purchased	0
Date of Purchase	0
Ticket Type	0
Ticket Subject	0
Ticket Description	0

Ticket Status	0
Resolution	5700
Ticket Priority	0
Ticket Channel	0
First Response Time	2819
Time to Resolution	5700
Customer Satisfaction Rating	5700

dtype: int64

A. Visualize and Confirm Patterns Before filling or dropping, confirm if missing values are related to ticket status:

[31]:
```
# Check missing values by Ticket Status
missing_summary = customer_ticket.groupby('Ticket Status')[['Resolution', 'Time to Resolution', 'Customer Satisfaction Rating', 'First Response Time']].apply(lambda x:x.isnull().sum())
print(missing_summary)
```

Ticket Status	Resolution	Time to Resolution	\
Closed	0	0	
Open	2819	2819	
PendingCustomer Response	2881	2881	

Ticket Status	Customer Satisfaction Rating	First Response Time
Closed	0	0
Open	2819	2819
PendingCustomer Response	2881	0

Interpretation: Missingvaluesarenotrandom:Theydependontheticketstatus.

Closed tickets are complete in these fields.

Open and Pending tickets are missing resolution-related fields, because they are not finished yet.

First Response Time is present for tickets where support has replied at least once (Closed or Pending Customer Response), but missing for tickets never responded to (Open).

Inference Whenanalyzingresolution,satisfaction,ortimemetrics,onlyuseclosedtickets.

To analyze response times, use both closed and pending customer response tickets.

Do not fill these missing values—they are logically missing, not due to data error.

[34]:
```
# Filtering for Analysis
resolved_ticket = customer_ticket[customer_ticket['Ticket Status'] == 'Closed']
resolved_ticket.shape
```

[34]: (2769, 17)

[40]:
```
#avg satisfaction rate
avg_satisfaction= resolved_ticket['Customer Satisfaction Rating'].mean()
avg_satisfaction= round(avg_satisfaction, 0)
print(f"Average satisfaction rating (closed tickets) is {avg_satisfaction}")
```

Average satisfaction rating (closed tickets) is 3.0

[]:
```

```

4 Converttimecolumn

[97]:
```
import pandas as pd

# Convert to datetime, handling missing values
customer_ticket['First Response Time'] = pd.to_datetime(customer_ticket['First Response Time'],errors= 'coerce')

customer_ticket['TimetoResolution']=pd.to_datetime(customer_ticket['Timeto Resolution'],errors= 'coerce')
```

5 ConvertDateColumn

[41]:
```
customer_ticket['Date of Purchase'] = pd.to_datetime(customer_ticket['Date of Purchase'])

customer_ticket.head(3)
```

[41]:

	TicketID	CustomerName	CustomerEmail	CustomerAge	\
0	1	Marisa Obrien	carrollallison@example.com	32	
1	2	JessicaRios	clarkeashley@example.com	42	
2	3	Christopher Robbins	gonzalestracy@example.com	48	

	Customer Gender	Product Purchased	Date of Purchase	Ticket Type	\
0	Other	GoProHero	2021-03-22	Technicalissue	
1	Female	LGSmart TV	2021-05-22	Technicalissue	
2	Other	DellXPS	2020-07-14	Technicalissue	

	Ticket Subject	\
0	Productsetup	
1	Peripheral compatibility	
2	Networkproblem	

	Ticket Description	\
0	I'm having an issue with the {product_purchase...	
1	I'm having an issue with the {product_purchase...	
2	I'm facing a problem with my {product_purchase...	

	Ticket Status	Resolution	\

```
0  PendingCustomer Response                                          NaN
1  PendingCustomer Response                                          NaN
2                  Closed   Case  maybeshowrecentlymycomputerfollow.
                                                                      \
  Ticket Priority Ticket Channel First Response Time Time to Resolution
0        Critical  Social media       6/1/202312:15                NaN
1        Critical  Social   Chat       6/1/202316:45                NaN
2             Low          media       6/1/202311:14  6/1/202318:05

      Customer Satisfaction Rating
0                              NaN
1                              NaN
2                              3.0
```

6 Extractdatefeature

[43]:
```
customer_ticket["Year"] = customer_ticket["Date of Purchase"].dt.year
customer_ticket["Month"] = customer_ticket["Date of Purchase"].dt.month
customer_ticket["Monthname"] = customer_ticket["Date of Purchase"].dt.
month_name()

customer_ticket["Day"]=customer_ticket["DateofPurchase"].dt.day
customer_ticket["DayofWeek"] = customer_ticket["Date of Purchase"].dt.dayofweek
```

[45]:
```
customer_ticket.head(3)
```

[45]:
```
   TicketID          CustomerName                CustomerEmail  CustomerAge  \
0         1          Marisa Obrien  carrollallison@example.com           32
1         2           JessicaRios   clarkeashley@example.com           42
2         3  Christopher Robbins  gonzalestracy@example.com           48

  Customer Gender Product Purchased Date of Purchase     Ticket Type  \
0           Other       GoProHero       2021-03-22  Technicalissue
1          Female     LGSmart TV       2021-05-22  Technicalissue
2           Other         DellXPS       2020-07-14  Technicalissue

            Ticket Subject  \
0            Productsetup
1  Peripheral compatibility
2          Networkproblem

                                  Ticket Description  ... Ticket Priority  \
0  I'mhaving an issuewith the {product_purchase...  ...          Critical
1  I'mhaving an issue with the{product_purchase...  ...          Critical
2  I'mfacing a problemwith my{product_purchase...  ...               Low

  Ticket Channel First Response Time Time to Resolution  \
```

```
0    Social media      6/1/2023 12:15                    NaN
1    Social  Chat       6/1/2023 16:45                    NaN
2            media      6/1/2023 11:14  6/1/2023 18:05
```

Customer Satisfaction Rating	Year	Month	Monthname	Day	DayofWeek
0 1 2 [3 rows x 22 columns]	2021	3	March	22	0
	2021	5	May	22	5
	2020	7	July	14	1
	NaN				
	NaN				
	3.0				

7 HandleDuplicatevalue

[14]:
```
customer_ticket.duplicated().sum()
```

[14]: 0

8 ExploratoryDataAnalysis(EDA)

8.0.1 UnivariateAnalysis

[47]:
```
# 1. Ticket Status Distribution
customer_ticket['Ticket Status'].value_counts()
```

[47]:
```
Ticket Status
PendingCustomer  Response    2881
Open                         2819
Closed                       2769
Name: count, dtype: int64
```

[48]:
```
# 2. Ticket Priority Distribution
customer_ticket['Ticket Priority'].value_counts()
```

[48]:
```
Ticket Priority
Medium      2192
Critical    2129
High        2085
Low         2063
Name: count, dtype: int64
```

[49]:
```
# 3. Ticket Channel Distribution
customer_ticket['Ticket Channel'].value_counts()
```

[49]:
```
Ticket Channel
Email          2143
Phone          2132
Socialmedia    2121
```

```
Chat                2073
Name: count, dtype: int64
```

[54]:
```
# 4. Customer Demographics
customer_ticket['Customer Gender'].value_counts()
```

[54]:
```
Customer Gender
Male      2896
Female    2887
Other     2686
Name: count, dtype: int64
```

[55]:
```
customer_ticket['Customer Age'].describe
```

[55]:
```
<boundmethodNDFrame.describe of 0          32
1          42
2          48
3          27
4          67
          ..
8464       22
8465       27
8466       57
8467       54
8468       53
Name: Customer Age, Length: 8469, dtype: int64>
```

[57]:
```
# 5. Product Popularity
customer_ticket['Product Purchased'].value_counts().head(10)
```

[57]:
```
Product Purchased
Canon EOS                240
GoPro Hero               228
NestThermostat           225
PhilipsHue Lights        221
AmazonEcho               221
LGSmartTV                219
SonyXperia               217
RoombaRobotVacuum        216
Apple AirPods            213
LGOLED                   213
Name: count, dtype: int64
```

8.0.2 Time-BasedAnalysis

[59]:
```
# 1. Tickets by Month
customer_ticket['Monthname'].value_counts()
```

[59]:
```
Monthname
January      736
October      735
July         727
April        718
February     715
November     704
May          701
September    696
December     696
August       691
June         678
March        672
Name: count, dtype: int64
```

[61]:
```
# 2. Tickets by Day of Week
customer_ticket['DayofWeek'].value_counts()
```

[61]:
```
DayofWeek
1    1263
3    1246
6    1198
5    1196
2    1196
0    1188
4    1182
Name: count, dtype: int64
```

8.0.3 BivariateAnalysis

[64]:
```
# 1. Ticket Status by Channel
pd.crosstab(customer_ticket['Ticket Channel'], customer_ticket['Ticket Status'])
```

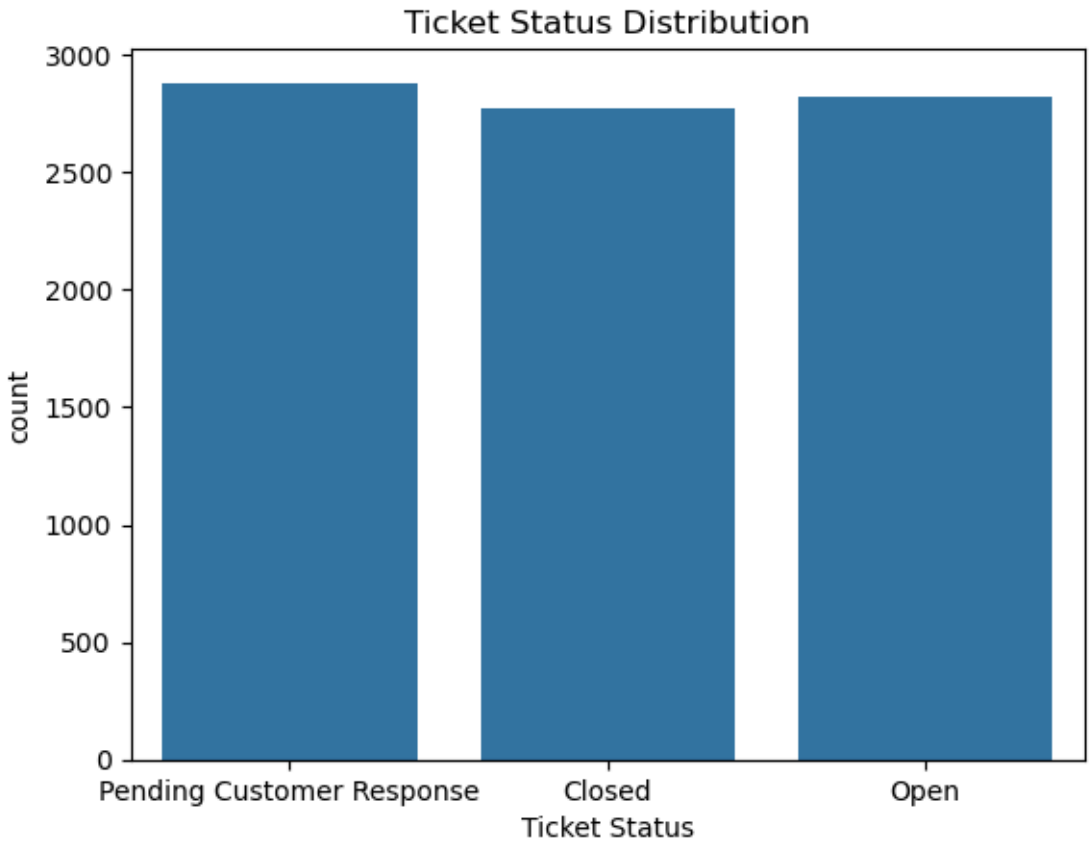
[64]:

Ticket Status	Closed	Open	Pending Customer Response
Ticket Channel			
Chat	674	685	714
Email	720	701	722
Phone	691	736	705
Socialmedia	684	697	740

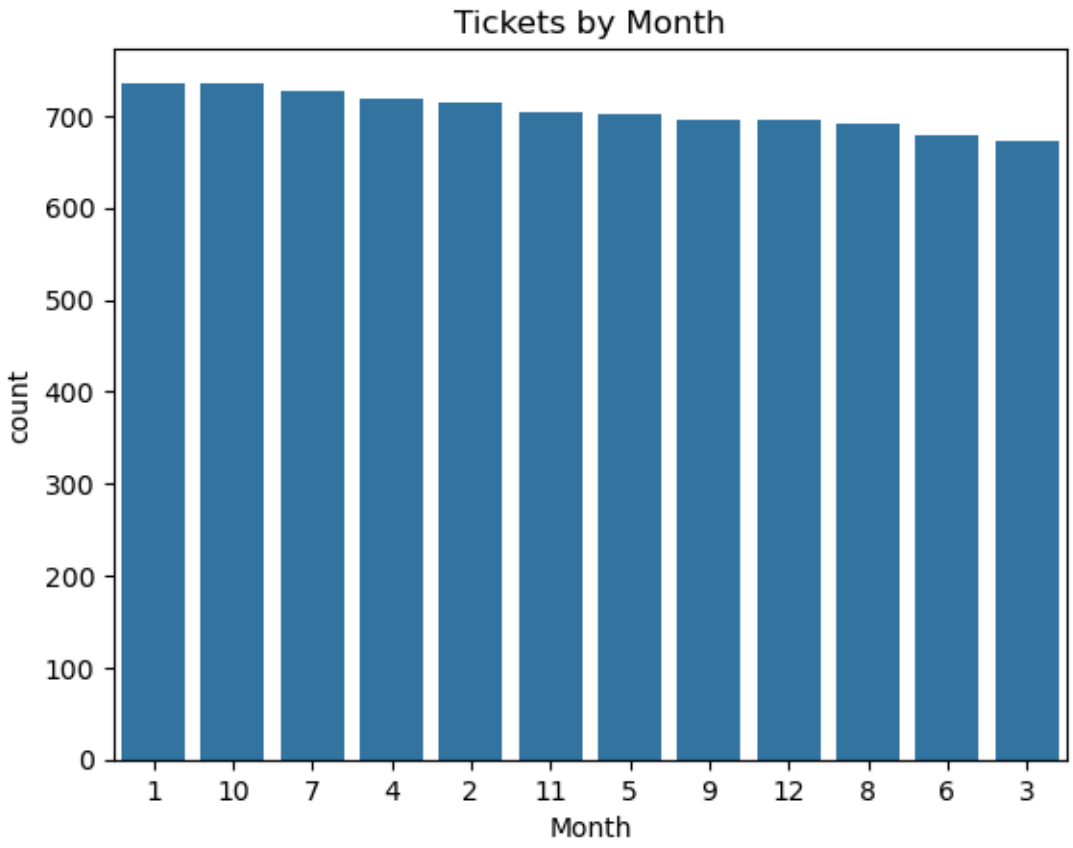
8.0.4 Visualization

[69]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

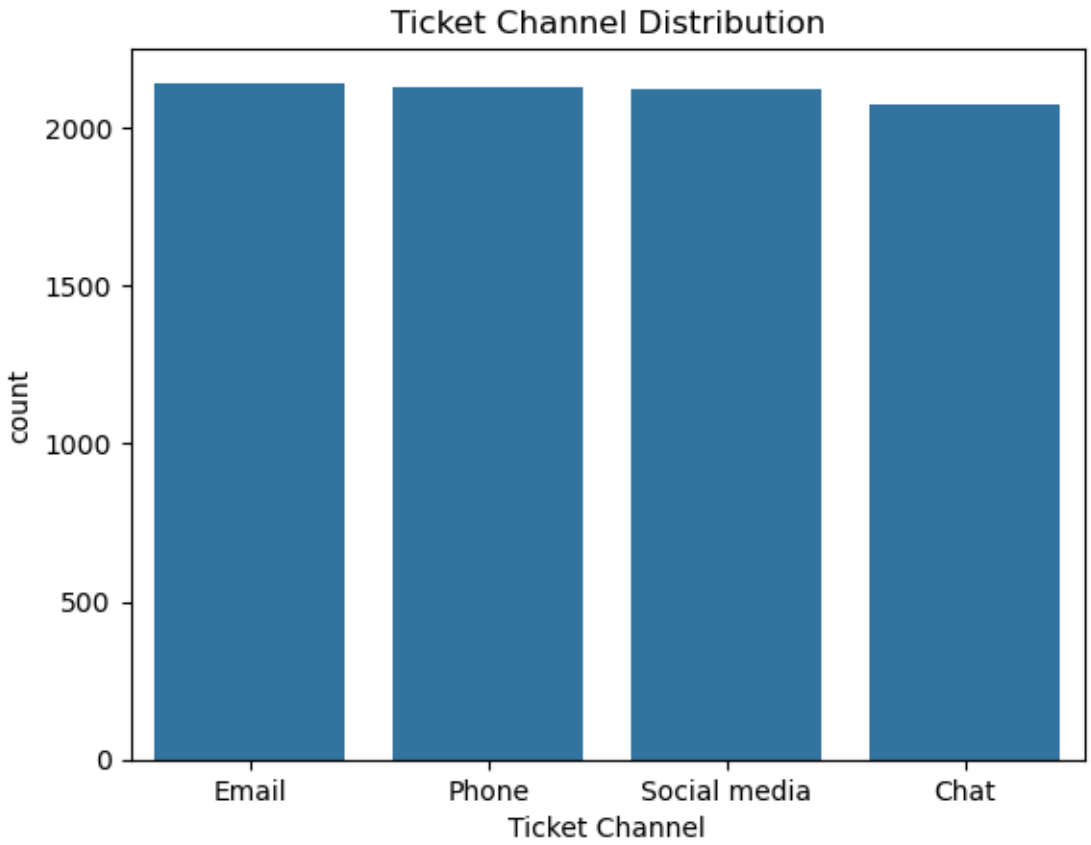
[70]:
```
sns.countplot(data=customer_ticket, x= 'Ticket Status')
plt.title('Ticket Status Distribution')
plt.show()
```



[75]:
```
#2. Ticket by month distribution
sns.countplot(data=customer_ticket, x='Month', order=customer_ticket['Month'].value_counts().index)

plt.title('TicketsbyMonth')
plt.show()
```



[79]:
```
# # Ticket Channel Distribution
sns.countplot(data=customer_ticket, x='Ticket Channel', order = customer_ticket['TicketChannel'].value_counts().index)

plt.title('TicketChannelDistribution')
plt.show()
```



8.0.5 Observation

Most Tickets come in January via Email. Again, more tickets are pending, yet to be resolved or closed.

[]:

9 1. Customer Experience Analysis

9.0.1 a. What are the main pain points and recurring issues reported by customers?

[82]:
```
# Top ticket types
print(customer_ticket['Ticket Type'].value_counts())
```

```
Ticket Type
Refundrequest              1752
Technical issue            1747
Cancellation request       1695
Productinquiry             1641
Billinginquiry             1634
Name: count, dtype: int64
```

[83]:
```
# Top ticket subjects
print(customer_ticket['Ticket Subject'].value_counts().head(10))
```

```
Ticket Subject
Refundrequest                576
Softwarebug                  574
Productcompatibility         567
Deliveryproblem              561
Hardwareissue                547
Batterylife                  542
Networkproblem               539
Installation support         530
Productsetup                 529
Paymentissue                 526
Name: count, dtype: int64
```

9.0.2 b. Which products or services generate the most support tickets?

[86]:
```
print(customer_ticket['Product Purchased'].value_counts().head(10))
```

```
Product Purchased
Canon EOS                    240
GoPro Hero                   228
NestThermostat               225
PhilipsHue Lights            221
AmazonEcho                   221
LGSmartTV                    219
SonyXperia                   217
RoombaRobotVacuum            216
Apple AirPods                213
LGOLED                       213
Name: count, dtype: int64
```

9.0.3 c. How do response and resolution times impact customer satisfaction?

[88]:
```
customer_ticket.head(2)
```

[88]:

	TicketID	CustomerName	CustomerEmail	CustomerAge	\
0	1	Marisa Obrien	carrollallison@example.com	32	
1	2	JessicaRios	clarkeashley@example.com	42	

	Customer Gender	Product Purchased	Date of Purchase	Ticket Type	\
0	Other	GoProHero	2021-03-22	Technicalissue	
1	Female	LG Smart TV	2021-05-22	Technicalissue	

	Ticket Subject	\
0	Productsetup	
1	Peripheral compatibility	

```
                                                Ticket Description  ... Ticket Priority  \
0  I'mhaving  a  issue  with  the{product_purchase...  ...          Critical
1  I'mhaving  n  issue  with  the{product_purchase...  ...          Critical
              a
   Ticket Channel First Response Time Time to Resolution  \
0          Social media     6/1/2023 12:15                    NaN
1                Chat       6/1/2023 16:45                    NaN

   Customer Satisfaction Rating  Year  Month  Monthname  Day  DayofWeek
0                           NaN  2021      3      March   22          0
1                           NaN  2021      5        May   22          5

[2 rows x 22 columns]
```

[98]:
```
closed.dtypes
```

[98]:
```
TicketID                          int64
CustomerName                     object
CustomerEmail                    object
CustomerAge                       int64
CustomerGender                   object
ProductPurchased                 object
DateofPurchase           datetime64[ns]
TicketType                       object
TicketSubject                    object
TicketDescription                object
TicketStatus                     object
Resolution                       object
TicketPriority                   object
TicketChannel                    object
First  Response Time             object
TimetoResolution                 object
CustomerSatisfaction            float64
Rating                            int32
Year                              int32
Month                            object
Monthname                         int32
Day                               int32
DayofWeek
dtype: object
```

[99]:
```
import pandas as pd

# Convert to datetime, handling missing values
customer_ticket['First Response Time'] = pd.to_datetime(customer_ticket['First
```

```
customer_ticket['Time to Resolution'] = pd.to_datetime(customer_ticket['Time to Resolution'],errors='coerce')
```

[100]:
```
closed = customer_ticket[customer_ticket['Ticket Status']== "Closed"]
print(closed[['First Response Time', 'Time to Resolution', 'Customer Satisfaction Rating']].corr())
```

```
                           First Response Time  Time to Resolution  \
First Response Time                   1.000000            0.056236
Time to Resolution                    0.056236            1.000000
Customer Satisfaction Rating         -0.037189           -0.010084

                           Customer Satisfaction Rating
First Response Time                           -0.037189
Time to Resolution                            -0.010084
Customer Satisfaction Rating                   1.000000
```

Customer satisfaction in this dataset is not strongly driven by how quickly support responds or resolves tickets.

10 2. Operational Efficiency

10.0.1 a. Which ticket channels are most used?

[101]:
```
customer_ticket['Ticket Channel'].value_counts()
```

[101]:
```
Ticket Channel
Email          2143
Phone          2132
Socialmedia    2121
Chat           2073
Name: count, dtype: int64
```

Email is the most used ticket channel

10.0.2 b. Aretherebottlenecksinthesupportprocess?

A bottleneck in support means certain channels, priorities, or types of tickets are experiencing significantly slower response or resolution times compared to others.

[110]:
```
# Calculate delays (in hours)
#  Calculate  Timedelta
customer_ticket['First Response Delay'] = customer_ticket['First Response Time'] - customer_ticket['Date of Purchase']


customer_ticket['First Response Delay (hrs)'] = customer_ticket['First Response Delay'].dt.total_seconds() / 3600


customer_ticket['First Response Delay (hrs)']
```

[110]:
```
0 1 2 3 4 8464 8465 8466 8467 8468 Name: First Response Delay (hrs),
Length: 8469, dtype: float64
        17776.75000
        0
        25259.23333
        3     ...
        22327.48333
        3          NaN
        29682.20000
        0
        4250.466667
                   NaN
```

[112]:
```
# Calculate resolution delay
customer_ticket['ResolutionDelay'] = customer_ticket['Time to Resolution'] -
    customer_ticket['DateOfPurchase']

customer_ticket['ResolutionDelay(hrs)']=customer_ticket['ResolutionDelay'].
    dt.total_seconds()/3600

customer_ticket['ResolutionDelay(hrs)']
```

[112]:
```
0                  NaN
1                  NaN
2         25266.083333
3         22321.950000
4 8464    29131.883333
8465           ...
8466               NaN
8467               NaN
8468      15676.516667
          14287.533333
Name: Re                  NaN
```

[124]:
```
# Bottleneck by channel
first_response_by_channel = customer_ticket.groupby('Ticket Channel')['First
    Response Delay(hrs)'].mean()

print(first_response_by_channel)
```

```
Ticket Channel
Chat                21344.023631
Email               21188.641100
Phone               21348.850096
Socialmedia         21055.905887
Name: First Response Delay (hrs), dtype: float64
```

[127]:
```
# Bottleneck by channel: resolution delay
resolution_delay_channel = customer_ticket.groupby('Ticket
    Channel')['Resolution Delay (hrs)'].mean()
```

```
resolution_delay_channel
```

[127]:
```
Ticket Channel
Chat            21245.929723
Email           21426.768102
Phone           21353.168548
Socialmedia     20900.374342
Name: Resolution Delay (hrs), dtype: float64
```

Since our dataset does not include the actual ticket creation time, all delay calculations are based on the time from product purchase to support response or resolution. This reflects customer behavior (how long after purchase they seek help), not the support team's speed.

Among all channels, customers who use Social Media tend to contact support and receive resolutions sooner after purchase compared to other channels. However, these figures do not reflect internal support process efficiency.

[129]:
```
pd.crosstab(customer_ticket['Ticket Priority'], customer_ticket['Ticket
Status'])
```

10.0.3 What is the distribution of ticket priorities and how are they handled?

[129]:

TicketStatus	Closed	Open	Pending	CustomerResponse
Ticket Priority				
Critical	726	692		711
High	705	704		676
Low	644	696		723
Medium	694	727		771

Interpretaion Analysisofticketstatusbyprioryshowsthattheresolutionratesaresimilar across all priority levels. Critical tickets are not being closed at a higher rate than others, which suggests that the current process does not prioritize urgent issues for faster handling. This could impact customer satisfaction and business outcomes.

Recommendation Werecommendreviewingtheticettriageprocesstoensurethatcriticaland high-priority tickets are resolved more quickly.

11 3. Customer Segmentation

11.0.1 a. Howdocustomerdemographics(age,gender)relatetotypesofissuesor satisfaction ratings?

i. Satistfaction by Gender

[133]:
```
# convert satisfaction rating to numeric
customer_ticket['Customer Satisfaction Rating'] = pd.to_numeric(customer_ticket['Customer Satisfaction Rating'])


# find the avearge satisfaction by gender
```

```
satisfaction_rating_gender = customer_ticket.groupby('Customer Gender')['Customer Satisfaction Rating'].mean()

satisfaction_rating_gender
```

[133]:
```
Customer Gender
Female    2.971545
Male      3.028384
Other     2.974684
Name: Customer Satisfaction Rating, dtype: float64
```

Males give relatively higher rating

ii. Satisfaction by Age

[136]:
```
# find the avearge satisfaction by age
satisfaction_rating_age = customer_ticket.groupby('Customer Age')['Customer Satisfaction Rating'].mean()

satisfaction_rating_age=satisfaction_rating_age.sort_values(ascending=False)
satisfaction_rating_age.head(10)
```

[136]:
```
Customer Age
34    3.430769
50    3.352941
32    3.348837
27    3.301587
38    3.290909
44    3.246154
51    3.237288
49    3.218182
60    3.200000
20    3.163636
Name: Customer Satisfaction Rating, dtype: float64
```

There is no trend in the age and rating

iii. Types of issues by gender

[139]:
```
issue_type_by_gender = customer_ticket.groupby(['Customer Gender', 'Ticket Type']).size().unstack(fill_value=0)

print(issue_type_by_gender)
```

```
TicketType       Billinginquiry  Cancellationrequest  Productinquiry  \
Customer Gender
Female                      561                  557             554
Male                        560                  591             558
Other                       513                  547             529

TicketType       Refundrequest  Technicalissue
Customer Gender
Female                     617             598
```

Male	61	57
Other	2	5
	52	57
	3	4

The numbers are in close range. Meaning, gender does not affect the the ticket type

12 4. Product Insights

12.0.1 Whichproductsaremostassociatedwithtechnical,billing,orrefundissues?

[152]:
```
product_Issue = customer_ticket.groupby(['Product Purchased','Ticket Type']).
size().unstack(fill_value = 0).sort_values(by = 'Technical Issue', ascending
=False)
product_Issue
```

[152]:

TicketType	Billinginquiry	Cancellationrequest	\
Product Purchased			
GoPro Hero	38	42	
AmazonEcho	43	40	
GarminForerunner	45	36	
Canon EOS	46	44	
LGWashingMachine	37	40	
AmazonKindle	36	37	
AppleAirPods	55	39	
NestThermostat	38	43	
PhilipsHue Lights	45	49	
LenovoThinkPad	29	37	
NikonD	37	35	
DysonVacuumCleaner	35	39	
PlayStation Vacuum	46	35	
RoombaRobot	45	35	
FitbitCharge	32	41	
NintendoSwitch Pro Controller	32	42	
HPPavilion	37	42	
iPhone	51	45	
SamsungGalaxy	39	26	
MicrosoftXbox Controller	38	53	
MicrosoftOffice	39	41	
Sony4KHDRTV	45	34	
LGSmartTV	37	53	
Xbox	32	40	
MicrosoftSurface	29	38	
GoProAction Camera	30	40	
BoseQuietComfort	49	27	
SonyXperia	45	48	
AsusROG	31	45	
CanonDSLR Camera	41	41	
FitbitVersa Smartwatch	33	40	
AutodeskAutoCAD	35	37	

Bose SoundLink Speaker	37		4
Samsung Soundbar LG OLED	35		5
Google Pixel Nintendo Switch	43		3
Adobe Photoshop MacBook Pro	32		9
Google Nest Dell XPS Sony	36		5
PlayStation Ticket Type Product	38		2
Purchased GoPro Hero Amazon	45		5
Echo Garmin Forerunner Canon	49		0
EOS LG Washing Machine	33		3
Amazon Kindle Apple AirPods	36		7
Nest Thermostat Philips Hue	Product inquiry		3
Lights Lenovo ThinkPad Nikon D	41	Refund request	\ 9
Dyson Vacuum Cleaner	39	50	2
PlayStation Roomba Robot	36	51	5
Vacuum Fitbit Charge Nintendo	52	43	3
Switch Pro Controller HP Pavilion	42	50	8
iPhone Samsung Galaxy	35	43	3
Microsoft Xbox Controller	40	44	7
Microsoft Office Sony 4K HDR TV	50	33	4
LG Smart TV Xbox Microsoft	41	49	9
Surface GoPro Action Camera	35	41	
Bose QuietComfort Sony Xperia	37	38	
Asus ROG Canon DSLR Camera	47	51	
Fitbit Versa Smartwatch Autodesk	42	33	
AutoCAD Bose SoundLink	44	25	
Speaker Samsung Soundbar	40	48	
	46	46	
	36	40	
	31	42	
	41	42	
	31	46	
	38	32	
	45	40	
	49	44	
	40	38	
	37	34	
	33	45	
	37	39	
	32	37	
	40	52	
	35	31	
	37	50	
	46	42	
	34	39	
	41	42	
		36	

LG OLED Google Pixel Nintendo	40	4
Switch Adobe Photoshop	48	1
MacBook Pro Google Nest Dell	32	3
XPS Sony PlayStation Ticket Type	33	6
Product Purchased GoPro Hero	38	3
Amazon Echo Garmin Forerunner	33	7
Canon EOS LG Washing Machine	32	3
Amazon Kindle Apple AirPods	35	5
Nest Thermostat Philips Hue	Technical issue	4
Lights Lenovo ThinkPad Nikon D	57	3
Dyson Vacuum Cleaner	48	4
PlayStation Roomba Robot	48	3
Vacuum Fitbit Charge Nintendo	48	4
Switch Pro Controller HP Pavilion	46	9
iPhone Samsung Galaxy	46	5
Microsoft Xbox Controller	46	2
Microsoft Office Sony 4K HDR TV	45	
LG Smart TV Xbox Microsoft	45	
Surface GoPro Action Camera	44	
Bose QuietComfort Sony Xperia	44	
Asus ROG Canon DSLR Camera	44	
Fitbit Versa Smartwatch Autodesk	44	
AutoCAD Bose SoundLink	44	
Speaker Samsung Soundbar LG	43	
OLED Google Pixel	43	
	43	
	43	
	42	
	42	
	42	
	42	
	42	
	41	
	41	
	41	
	40	
	40	
	40	
	39	
	39	
	39	
	39	
	37	
	37	
	37	

Nintendo Switch	3
Adobe Photoshop	6
MacBook Pro	3
Google Nest	6
Dell XPS	3
Sony PlayStation	5
	3
	5
	3
	4
	3
	0

12.0.2 b. By Recency of Purchase

[157]:
```
#Issue by Year
issue_Year = customer_ticket.groupby(['Year', 'Ticket Type']).size().
    unstack(fill_value=0)

issue_Year
```

[157]:

Ticket Type	Billing inquiry	Cancellation request	Product inquiry	\
Year				
2020	802	850	847	
2021	832	845	794	

TicketType	Refund request	Technical issue
Year		
2020	898	839
2021	854	908

[162]:
```
#Issue by Month
issue_month_name = customer_ticket.groupby(['Monthname', 'Ticket Type']).size().
    unstack(fill_value = 0).sort_values(by = "Technical issue", ascending =
    False)
issue_month_name
```

[162]:

Ticket Type	Billing inquiry	Cancellation request	Product inquiry	\
Monthname				
October	125	155	142	
January	137	145	148	
December	139	132	130	
February	156	133	140	
July	151	142	130	
May	128	142	134	
June	129	132	121	
August	139	124	140	
November	104	160	148	
September	130	142	153	
March	136	141	120	
April	160	147	135	

TicketType	Refund request	Technical issue
Monthname		

October	15	16	
January	2	1	
Decembe	14	15	
r	8	8	
February	14	15	
July May	2	3	
June	13	15	
August	3	3	
Novembe	15	15	
r	3	1	
Septembe	14	15	
r March	7	0	
April	14	14	
	8	8	
	14	14	
	3	5	
	15	13	
	4	8	
	13	13	
	6	5	
	14	13	
	5	0	
	15	12	
	1	5	

12.0.3 Issues by Product Line

[164]:
```
customer_ticket['Product Line'] = customer_ticket['Product Purchased'].str.split().str[0]

print(customer_ticket.groupby(['ProductLine','TicketType']).size().unstack(fill_value=0).sort_values(by = 'Billing inquiry', ascending = False)
```

TicketType	Billing inquiry	Cancellation request	Product inquiry	\
Product Line				
Adobe	38	39	33	
Amazon	79	77	74	
Apple	55	39	40	
Asus	31	45	40	
Autodesk	35	37	46	
Bose	86	72	71	
Canon	87	85	87	
Dell	33	37	32	
Dyson	35	39	47	
Fitbit	65	81	77	
Garmin	45	36	36	
GoPro	68	82	74	
Google	81	88	81	
HP	37	42	36	
LG	117	145	131	
Lenovo	29	37	35	
MacBook	45	25	38	
Microsoft	106	132	106	
Nest	38	43	50	
Nikon	37	35	37	
Nintendo	68	79	78	
Philips	45	49	41	
PlayStation	46	35	42	
Roomba	45	35	44	
Samsung	74	65	82	
Sony	126	131	112	

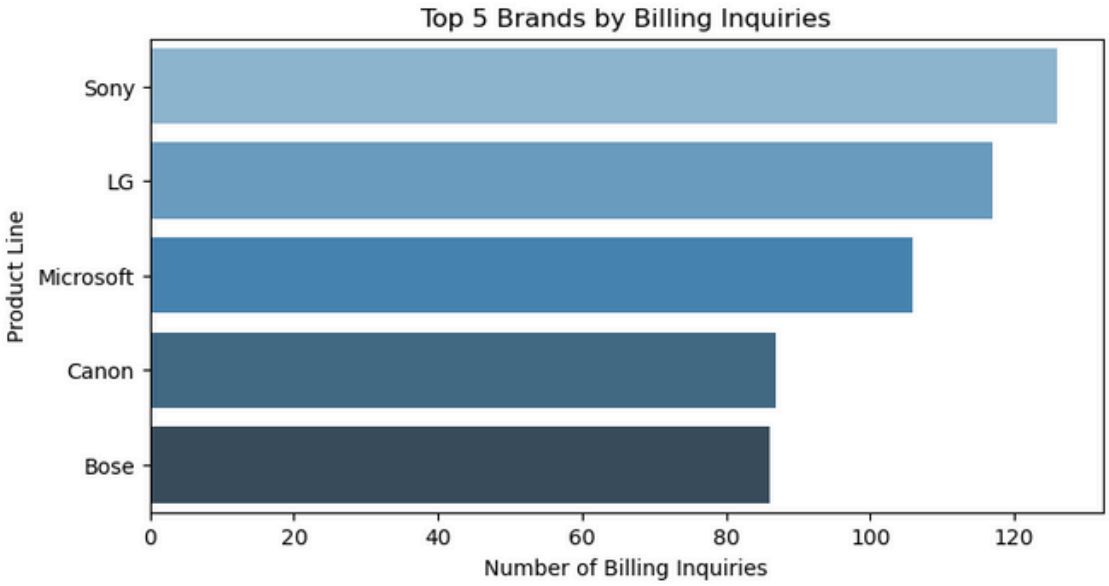
Xbox iPhone	3		4	4
Ticket Type	2		0	0
Product Line	5		4	3
Adobe	Refund request	Technical issue	5	1
Amazon		36		
Apple Asus	3	94		
Autodesk	5	46		
Bose Canon	9	40		
Dell Dyson	5	39		
Fitbit Garmin	3	79		
GoPro	3	87		
Google HP	130	34		
LG Lenovo	4	44		
MacBook	9	82		
Microsoft	9	48		
Nest Nikon	7	98		
Nintendo	8	72		
Philips	8	43		
PlayStation	4	125		
Roomba	122	44		
Samsung	8	35		
Sony Xbox	8	125		
iPhone	147	45		
	4	44		
	9	79		
	3	45		
	1	44		
	7	44		
	7	79		
	4	112		
	148	41		
	2	43		
	5			
	4			
	8			
	8			
	2			

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[164], line 2
      1 customer_ticket['Product Line'] = customer_ticket['Product Purchased'].
      2 str.split().str[0]
----> 2print(customer_ticket.groupby(['ProductLine','TicketType']).size().
      3 unstack(fill_value=0).sort_values(by = "Billing inquiry", ascending = False)

AttributeError : 'NoneType' object has no attribute 'sort_values'
```

[166]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

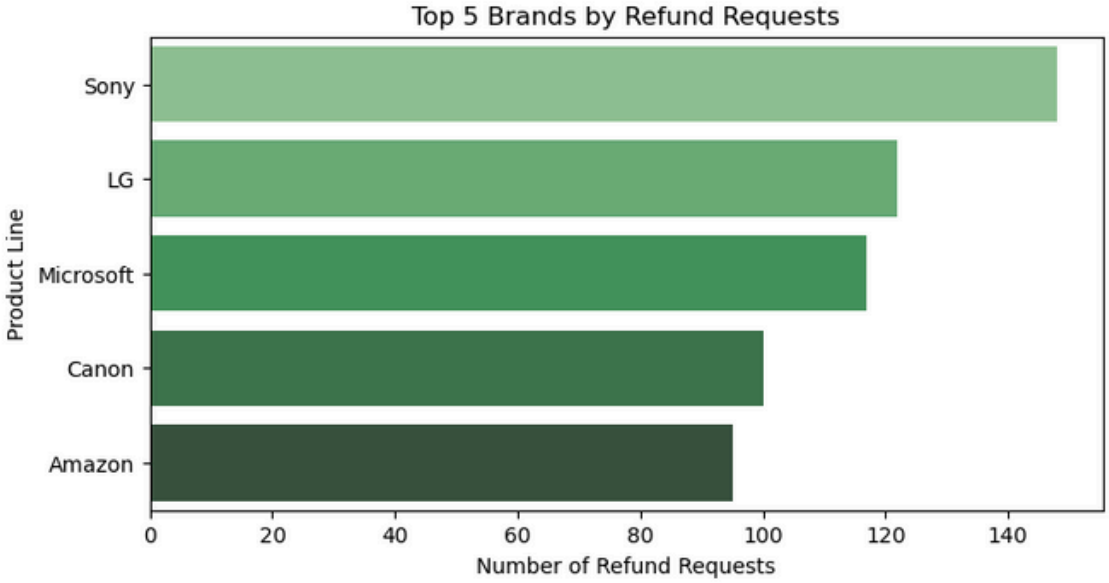
```
# Group by product line and ticket type, count occurrences
issue_counts = customer_ticket.groupby(['Product Line', 'Ticket Type']).size().
  unstack(fill_value=0)

# --- Billing Inquiry ---
top_billing = issue_counts['Billing inquiry'].sort_values(ascending=False).
  head(5)

plt.figure(figsize=(8,4))
sns.barplot(x=top_billing.values, y=top_billing.index, palette="Blues_d")
plt.title('Top 5 Brands by Billing Inquiries')
plt.xlabel('Number of Billing Inquiries')
plt.ylabel('Product Line')
plt.show();
# --- Refund Request ---
top_refund = issue_counts['Refund request'].sort_values(ascending=False).head(5)
plt.figure(figsize=(8,4))
sns.barplot(x=top_refund.values, y=top_refund.index, palette="Greens_d")
plt.title('Top 5 Brands by Refund Requests')
plt.xlabel('Number of Refund Requests')
plt.ylabel('Product Line')
plt.show();
# --- Technical Issue ---
top_technical = issue_counts['Technical issue'].sort_values(ascending=False).

  head(5)

plt.figure(figsize=(8,4))
sns.barplot(x=top_technical.values, y=top_technical.index, palette="Reds_d")
plt.title('Top 5 Brands by Technical Issues')
plt.xlabel('Number of Technical Issues')
plt.ylabel('Product Line')
plt.show();
```

```
C:\Users\user\AppData\Local\Temp\ipykernel_12512\1633705761.py:10:
FutureWarning:
Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=top_billing.values, y=top_billing.index, palette="Blues_d")
```



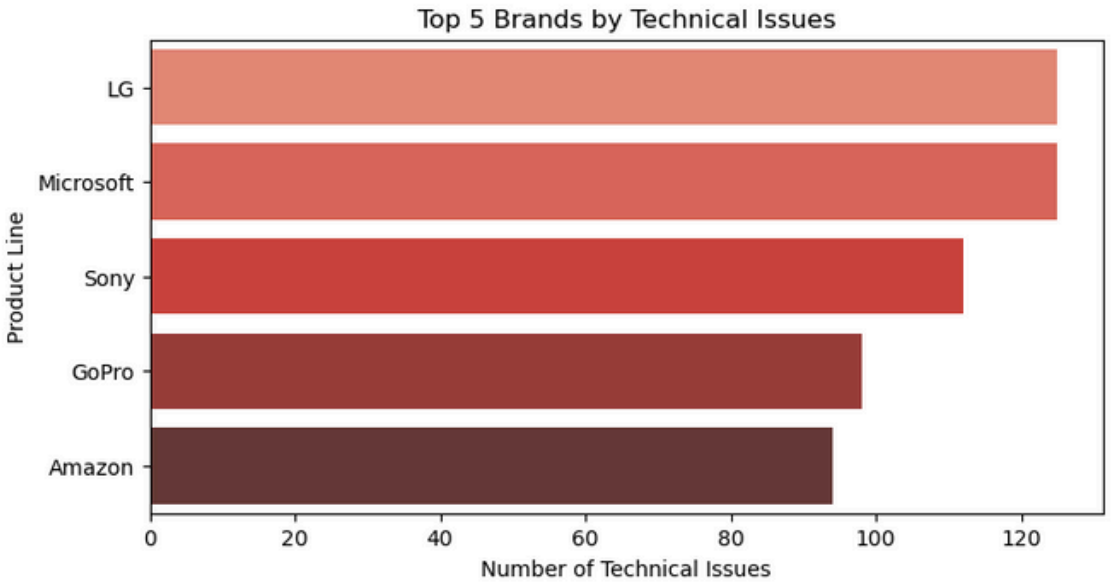
C:\Users\user\AppData\Local\Temp\ipykernel_12512\1633705761.py:19:
FutureWarning:
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

```
sns.barplot(x=top_refund.values, y=top_refund.index, palette="Greens_d")
```



C:\Users\user\AppData\Local\Temp\ipykernel_12512\1633705761.py:28:
FutureWarning:
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

```
sns.barplot(x=top_technical.values, y=top_technical.index, palette="Reds_d")
```



[]:

Subscribe to our Newsletter @ The ALU DATATOK Brief on LinkedIn

Thank you!

Connect on Linkedin:
Uchechukwu Lorreta Anyika.

Open to roles in customer & data analytics.

~💜