🔷Project Name - Laptop Price Analysis

Project Type - Exploratory Data Analysis (EDA), Market Insights Generation, Price Trend Analysis

Contribution - Individual

Name - Aditya Singh

🧠 Introduction

Customer satisfaction is one of the most important metrics for the growth, profitability, and long-term sustainability of any business. A satisfied customer is more likely to remain loyal, make repeat purchases, and recommend the brand to others, while a dissatisfied customer can quickly damage the reputation of the company through negative reviews and feedback. In today's highly competitive and globalized market, where customers have multiple alternatives available at their fingertips, retaining existing customers is as critical as acquiring new ones. To achieve high customer retention, organizations need to understand the factors that influence satisfaction and dissatisfaction. Traditionally, customer satisfaction was measured using surveys and feedback forms, but these methods are reactive and often fail to address issues before a customer decides to leave. With the rapid advancement of data analytics and machine learning, businesses can now shift from reactive strategies to proactive solutions.

By analyzing historical customer data, businesses can identify hidden patterns, detect early warning signs of dissatisfaction, and predict overall satisfaction levels with considerable accuracy. This project leverages such analytical techniques to forecast customer satisfaction based on multiple influencing factors, such as service quality, product usability, resolution time for complaints, communication effectiveness, ticket handling efficiency, and customer engagement history.

The main objective of this project is to help businesses recognize dissatisfied customers in advance so that corrective actions—such as faster issue resolution, personalized offers, or improved service quality—can be taken immediately. This predictive approach not only reduces customer churn but also enhances brand reputation, increases revenue, and fosters stronger customer relationships.

With a well-trained predictive model integrated into customer relationship management (CRM) systems, companies can continuously monitor satisfaction levels in real-time and make data-driven decisions. This ensures that customer concerns are addressed before they escalate, turning potential dissatisfaction into opportunities for improvement and growth.

❗ Problem Statement:-

Despite delivering quality products and services, many businesses still face the challenge of maintaining consistently high customer satisfaction levels. Several factors contribute to customer dissatisfaction, including delayed responses to inquiries, unresolved complaints, miscommunication, limited customization options, and the absence of a personalized customer experience. In the current business landscape, where competition is intense and switching to a competitor is easy, even a minor negative experience can result in the loss of a loyal customer. Most organizations depend on reactive approaches to address customer issues—responding only after a complaint is received or a negative review is posted. While this can solve individual problems, it often comes too late, as the customer may have already made the decision to disengage or switch brands. This reactive cycle not only leads to customer churn but also damages the company’s brand image and affects its market position.

To overcome this challenge, businesses need to move towards predictive, data-driven solutions. By utilizing historical customer data—such as service usage patterns, ticket resolution times, product feedback, and demographic details—organizations can uncover valuable insights into the factors that influence satisfaction and dissatisfaction. Machine learning techniques provide a powerful way to analyze these large datasets, identify trends, and forecast future satisfaction levels with a high degree of accuracy.

The core problem addressed in this project is:

How can we use historical data and machine learning techniques to accurately predict whether a customer will be satisfied or dissatisfied, thereby enabling proactive and personalized customer service? Solving this problem will allow companies to take preventive measures before dissatisfaction escalates, such as prioritizing at-risk customers, optimizing service processes, and tailoring communication strategies. This proactive approach has the potential to reduce churn, increase customer loyalty, and drive long-term business growth.

🔍 Techniques & Tools Used to Solve the Problem:-

The Customer Satisfaction Prediction project involves a series of systematic steps and methodologies to ensure accurate, reliable, and interpretable results. The following techniques and tools were applied during the project lifecycle:

1. Data Preprocessing:

Before building any machine learning model, the raw dataset was cleaned and transformed into a structured format suitable for analysis. This included: Data Cleaning: Removing duplicate records, fixing incorrect entries, and eliminating irrelevant fields. Handling Missing Values: Imputing missing data using mean/median for numerical features and mode for categorical variables, or removing records where necessary. Encoding Categorical Features: Converting categorical variables such as Customer Gender, Ticket Type, and Ticket Channel into numerical formats using label encoding and one-hot encoding. Scaling Numerical Features: Standardizing continuous variables like Customer Age, First Response Time, and Time to Resolution to ensure fair comparison between features and improve model performance.

2. Exploratory Data Analysis (EDA):

EDA was performed to understand the structure of the data, identify trends, and detect patterns. Distribution Analysis: Studied the spread of variables like satisfaction ratings and resolution times. Correlation Analysis: Measured relationships between variables to identify key satisfaction drivers. Visual Exploration: Created histograms, bar plots, box plots, and correlation heatmaps to better understand the dataset.

3. Feature Engineering:

New derived variables were created to capture deeper insights and improve model performance, such as: Age Groups: Categorizing customers into segments like Young, Adult, and Senior. Ticket Resolution Time: Calculating the exact time taken to resolve each customer complaint. Complaint Categories: Grouping similar complaints into common categories for pattern recognition.

4. Machine Learning Models:

Multiple algorithms were used to build predictive models and compare their performance: Logistic Regression: For baseline binary classification. Random Forest Classifier: To capture non-linear patterns and feature importance.

XGBoost Classifier: For high-performance gradient boosting with better accuracy and generalization.

5. Model Evaluation:

Each model was assessed using standard performance metrics: Accuracy: Percentage of correct predictions. Precision & Recall: To measure the model's effectiveness in identifying satisfied/dissatisfied customers. F1-score: Harmonic mean of precision and recall for balanced evaluation. ROC-AUC Score: To evaluate the model's ability to differentiate between classes.

6. Visualization:

Data insights and model results were presented using: Matplotlib – For static, customizable plots. Seaborn – For visually appealing statistical graphics. Plotly – For interactive data visualizations.

🛠️ Tools and Libraries Used:-

The development of the Customer Satisfaction Prediction project relied on a combination of programming languages, libraries, and development tools that enabled efficient data processing, analysis, visualization, and model building. The selected tools ensured scalability, reproducibility, and ease of collaboration throughout the project lifecycle.

1. Programming Language:

Python – Python was chosen as the primary programming language due to its simplicity, flexibility, and rich ecosystem of data science libraries. Its readability and community support made it ideal for implementing end-to-end machine learning workflows.

2. Libraries:

pandas – Used extensively for data manipulation and preprocessing. It provided powerful data structures like DataFrames, which made tasks such as filtering, merging, grouping, and cleaning data straightforward and efficient. numpy – Utilized for numerical computations and array operations. It served as the foundation for many data science libraries and helped in handling large datasets efficiently. matplotlib / seaborn – Both were used for data visualization. Matplotlib allowed creation of static, highly customizable plots. Seaborn offered a higher-level interface for generating attractive and informative statistical graphics such as heatmaps, box plots, and histograms. scikit-learn – The core library for machine learning model development and evaluation. It provided implementations for algorithms such as Logistic

Regression, Random Forest, and utilities for splitting data, scaling features, and computing metrics like accuracy, precision, recall, and F1-score. xgboost – A high-performance gradient boosting library used to build an optimized classification model capable of handling non-linear relationships and improving prediction accuracy.

3. Development Environment:

Jupyter Notebook – Served as the main development platform for writing, executing, and documenting the code. It allowed the integration of live code, visualizations, and narrative text in a single environment, making experimentation and analysis highly interactive. 4. Version Control Git & GitHub – Used for version control and collaboration. Git tracked changes in the codebase, while GitHub served as the remote repository, enabling backup, code sharing, and collaborative development.

Github Link-

https://github.com/Virtueadii12/Customer-Satisfaction-Prediction-

Customer Satisfaction Prediction

In [21]:
```
#Importing Libraries
```

In [6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
```

In [8]:
```
#Loading the dataset
```

In [12]:
```
data = pd.read_csv("customer_data.csv")
```

Data Preprocessing

In [15]:
```
#Displaying Basic Info About Dataset
```

In [19]:
```
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8469 entries, 0 to 8468
Data columns (total 17 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Ticket ID                     8469 non-null   int64  
 1   Customer Name                 8469 non-null   object 
 2   Customer Email                8469 non-null   object 
 3   Customer Age                  8469 non-null   int64  
 4   Customer Gender               8469 non-null   object 
 5   Product Purchased             8469 non-null   object 
 6   Date of Purchase              8469 non-null   object 
 7   Ticket Type                   8469 non-null   object 
 8   Ticket Subject                8469 non-null   object 
 9   Ticket Description            8469 non-null   object 
 10  Ticket Status                 8469 non-null   object 
 11  Resolution                    2769 non-null   object 
 12  Ticket Priority               8469 non-null   object 
 13  Ticket Channel                8469 non-null   object 
 14  First Response Time           5650 non-null   object 
 15  Time to Resolution            2769 non-null   object 
 16  Customer Satisfaction Rating  2769 non-null   float64
dtypes: float64(1), int64(2), object(14)
memory usage: 1.1+ MB
None
```

In [23]:
```python
columns= list(data)
columns
```

Out[23]:
```
['Ticket ID',
 'Customer Name',
 'Customer Email',
 'Customer Age',
 'Customer Gender',
 'Product Purchased',
 'Date of Purchase',
 'Ticket Type',
 'Ticket Subject',
 'Ticket Description',
 'Ticket Status',
 'Resolution',
 'Ticket Priority',
 'Ticket Channel',
 'First Response Time',
 'Time to Resolution',
 'Customer Satisfaction Rating']
```

In [25]:
```python
#Checking Null Values
```

In [29]:
```python
data.isnull().sum()
```

Out[29]:
```
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
```

In [31]:
```
#calculating Total Zeros in Resolution, First response time, Time t
```

In [35]:
```
(data[columns[12:18]]==0).sum()
```

Out[35]:
```
Ticket Priority                 0
Ticket Channel                  0
First Response Time             0
Time to Resolution              0
Customer Satisfaction Rating    0
dtype: int64
```

In [37]:
```
#Replace Statement
```

In [41]:
```
data[columns[12:18]]= data[columns[12:18]].replace(0,np.nan)
```

In [43]:
```
#Again Checking Null Values
```

In [47]:
```
data.isnull().sum()
```

Out[47]:
```
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
```

In [49]:
```
#Before Drop Statement
```

In [53]:
```
data.shape
```

Out[53]:
```
(8469, 17)
```

In [55]:
```
#Drop Statement
```

In [59]:
```
data.dropna(inplace=True)
```

In [61]:
```
#After Drop Statement
```

In [65]:
```
data.shape
```

Out[65]:
```
(2769, 17)
```

In [67]:
```
# Encoding categorical variables
```

In [75]:
```
label_encoders = {}
for column in data.select_dtypes(include=["object"]).columns:
    label_encoders[column] = LabelEncoder()
    data[column] = label_encoders[column].fit_transform(data[column
```

In [77]:
```
# Define features and target variable
```

In [95]:
```
X = data.drop(["Ticket ID", "Customer Satisfaction Rating"], axis=1
y = data["Customer Satisfaction Rating"]
```

In [97]:
```
# Splitting the dataset
```

In [131…
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
```

In [133…
```
# Feature Scaling
```

In [137…

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Model Building

In [140…

```
#Training a Random Forest Classifier
```

In [144…

```
rfc = RandomForestClassifier(random_state = 42)
rfc.fit(X_train, y_train)
```

Out[144…

```
▾ RandomForestClassifier
▸ Parameters
```

In [146…

```
#Prediction on the test set
```

In [150…

```
y_pred = rfc.predict(X_test)
```

In [152…

```
#Model Evaluation
```

In [162…

```
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_p
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Accuracy: 0.21540312876052947
Classification Report:
               precision    recall  f1-score   support

         1.0       0.18      0.18      0.18       168
         2.0       0.22      0.21      0.21       174
         3.0       0.26      0.26      0.26       175
         4.0       0.21      0.20      0.21       162
         5.0       0.21      0.22      0.21       152

    accuracy                           0.22       831
   macro avg       0.22      0.22      0.21       831
weighted avg       0.22      0.22      0.22       831

Confusion Matrix:
 [[31 46 34 28 29]
 [43 36 33 28 34]
 [40 25 46 33 31]
 [33 30 30 32 37]
 [26 28 35 29 34]]
```

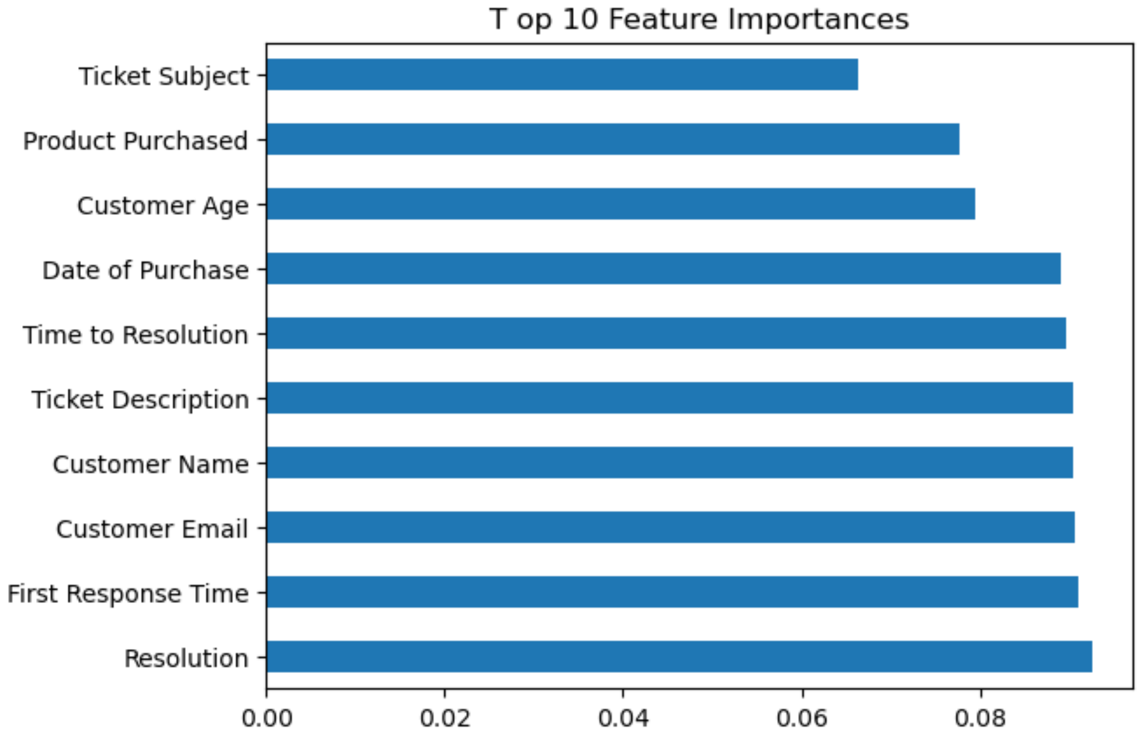
In [164…

```
# Visualization of Results
```

In [ ]:

In [ ]:
```
# Feature Importance
```

In [168…
```
feature_importances = pd.Series(rfc.feature_importances_, index=X.c
feature_importances.nlargest(10).plot(kind="barh")
plt.title("T op 10 Feature Importances")
plt.show()
```



Sample Output

In [180…
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.cluster import KMeans
```

In [182…
```
#Loading the Dataset
```

In [186…
```
data = pd.read_csv("customer_data.csv")
```

In [190…
```
# Displaying the first few rows of the dataset
```

In [194…
```
print(data.head())
```

```
   Ticket ID        Customer Name              Customer Email  Custo
mer Age  \
0          1        Marisa Obrien  carrollallison@example.com
32
```

```
1          2         Jessica Rios    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 Typ
e  \
0          Other        GoPro Hero       2021-03-22  Technical issu
e
1         Female       LG Smart TV       2021-05-22  Technical issu
e
2          Other          Dell XPS       2020-07-14  Technical issu
e
3         Female  Microsoft Office       2020-11-13  Billing inquir
y
4         Female  Autodesk AutoCAD       2020-02-04  Billing inquir
y

             Ticket Subject  \
0             Product setup
1  Peripheral compatibility
2           Network problem
3            Account access
4                 Data loss

                                    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...
4  I'm having an issue with the {product_purchase...

              Ticket Status                                    Res
olution  \
0  Pending Customer Response
NaN
1  Pending Customer Response
NaN
2                    Closed   Case maybe show recently my computer
follow.
3                    Closed  Try capital clearly never color toward
story.
4                    Closed                    West decision eviden
ce bit.

  Ticket Priority Ticket Channel  First Response Time   Time to Reso
lution  \
0        Critical   Social media  2023-06-01 12:15:36
NaN
1        Critical           Chat  2023-06-01 16:45:38
NaN
2             Low   Social media  2023-06-01 11:14:38  2023-06-01 1
```

```
8:05:38
3            Low   Social media  2023-06-01 07:29:40  2023-06-01 0
1:57:40
4            Low          Email  2023-06-01 00:12:42  2023-06-01 1
9:53:42

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

In [196…

```
# Perform initial exploratory data analysis (EDA)
```

In [200…

```
print(data.info())
print(data.describe())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8469 entries, 0 to 8468
Data columns (total 17 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Ticket ID                     8469 non-null   int64  
 1   Customer Name                 8469 non-null   object 
 2   Customer Email                8469 non-null   object 
 3   Customer Age                  8469 non-null   int64  
 4   Customer Gender               8469 non-null   object 
 5   Product Purchased             8469 non-null   object 
 6   Date of Purchase              8469 non-null   object 
 7   Ticket Type                   8469 non-null   object 
 8   Ticket Subject                8469 non-null   object 
 9   Ticket Description            8469 non-null   object 
 10  Ticket Status                 8469 non-null   object 
 11  Resolution                    2769 non-null   object 
 12  Ticket Priority               8469 non-null   object 
 13  Ticket Channel                8469 non-null   object 
 14  First Response Time           5650 non-null   object 
 15  Time to Resolution            2769 non-null   object 
 16  Customer Satisfaction Rating  2769 non-null   float64
dtypes: float64(1), int64(2), object(14)
memory usage: 1.1+ MB
None
         Ticket ID  Customer Age  Customer Satisfaction Rating
count  8469.000000   8469.000000                   2769.000000
mean   4235.000000     44.026804                      2.991333
std    2444.934048     15.296112                      1.407016
min       1.000000     18.000000                      1.000000
25%    2118.000000     31.000000                      2.000000
50%    4235.000000     44.000000                      3.000000
75%    6352.000000     57.000000                      4.000000
max    8469.000000     70.000000                      5.000000
```

In [ ]:

```
# Printing column names
```

In [204…

```
print(data.columns)
```

```
Index(['Ticket ID', 'Customer Name', 'Customer Email', 'Customer Ag
e',
       'Customer Gender', 'Product Purchased', 'Date of Purchase',
       'Ticket Type', 'Ticket Subject', 'Ticket Description', 'Ticke
t Status',
       'Resolution', 'Ticket Priority', 'Ticket Channel',
       'First Response Time', 'Time to Resolution',
       'Customer Satisfaction Rating'],
      dtype='object')
```

Analyzing customer support ticket trends

In [208…

```
# Identify common issues
```

In [212…

```
common_issues = data["Ticket Subject"].value_counts().head(10)
print("Top 10 Common Issues:")
print(common_issues)
```

```
Top 10 Common Issues:
Ticket Subject
Refund request          576
Software bug            574
Product compatibility   567
Delivery problem        561
Hardware issue          547
Battery life            542
Network problem         539
Installation support    530
Product setup           529
Payment issue           526
Name: count, dtype: int64
```

In [214…

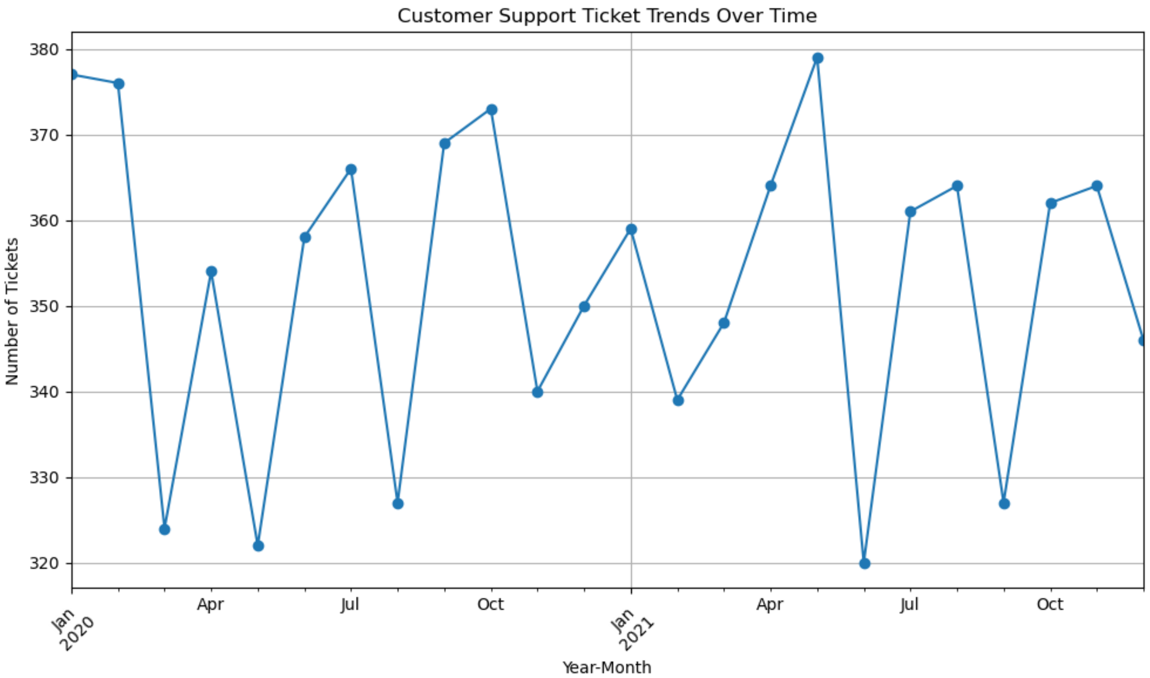
```
# Plotting ticket trends over time
```

In [218…

```
data["Date of Purchase"]= pd.to_datetime(data["Date of Purchase"])
data["YearMonth"]= data["Date of Purchase"].dt.to_period("M")
ticket_trends = data.groupby("YearMonth").size()
```

In [222…

```
plt.figure(figsize=(10, 6))
ticket_trends.plot(kind="line", marker="o")
plt.title("Customer Support Ticket Trends Over Time")
plt.xlabel("Year-Month")
plt.ylabel("Number of Tickets")
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Segment customers

In [226…

```python
# Segment based on ticket types
```

In [230…

```python
ticket_type_segmentation = data.groupby("Ticket Type").size()
print("\nSegmentation based on Ticket Types:")
print(ticket_type_segmentation)
```

```
Segmentation based on Ticket Types:
Ticket Type
Billing inquiry         1634
Cancellation request    1695
Product inquiry         1641
Refund request          1752
Technical issue         1747
dtype: int64
```

In [232…

```python
# Segment based on satisfaction levels
```

In [236…

```python
satisfaction_segmentation = data.groupby("Customer Satisfaction Rat
print("\nSegmentation based on Customer Satisfaction Levels:")
print(satisfaction_segmentation)
```

```
Segmentation based on Customer Satisfaction Levels:
Customer Satisfaction Rating
1.0    553
2.0    549
3.0    580
4.0    543
5.0    544
dtype: int64
```

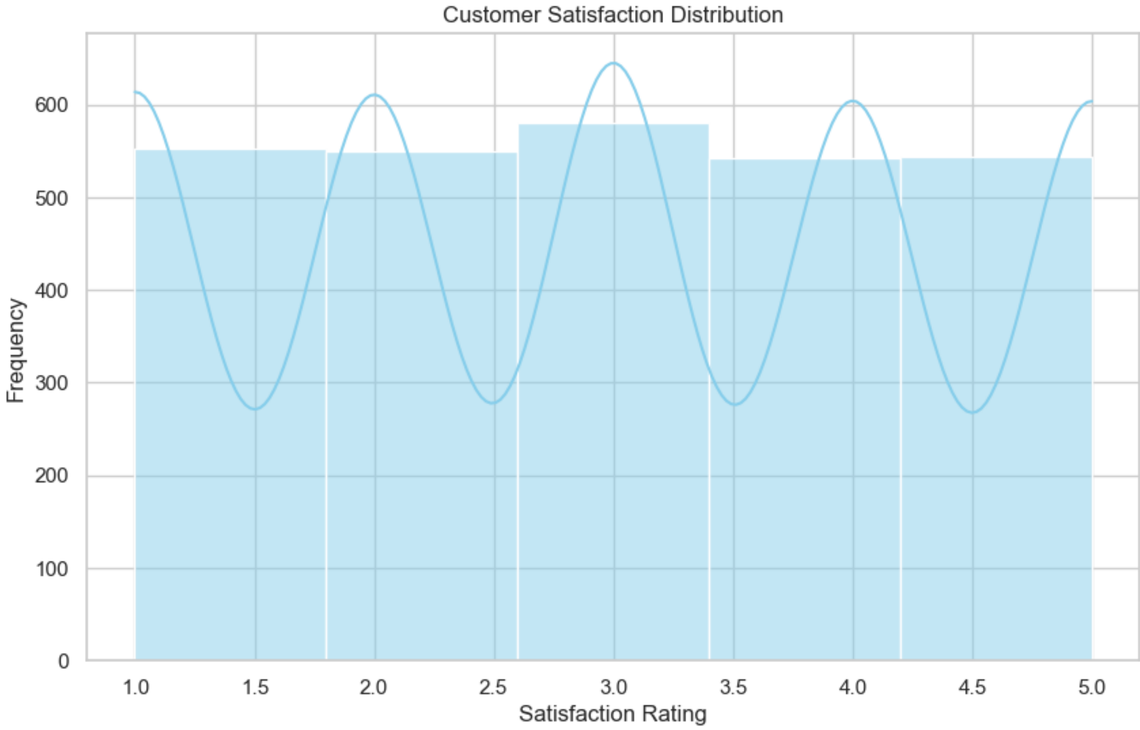
In [238…

```python
# Seting up the plotting aesthetics
```

In [244…
```
sns.set(style="whitegrid")
```

In [246…
```
#Customer Satisfaction Distribution
```

In [250…
```
plt.figure(figsize=(10, 6))
sns.histplot(data["Customer Satisfaction Rating"], bins=5,kde=True,
plt.title("Customer Satisfaction Distribution")
plt.xlabel("Satisfaction Rating")
plt.ylabel("Frequency")
plt.show()
```

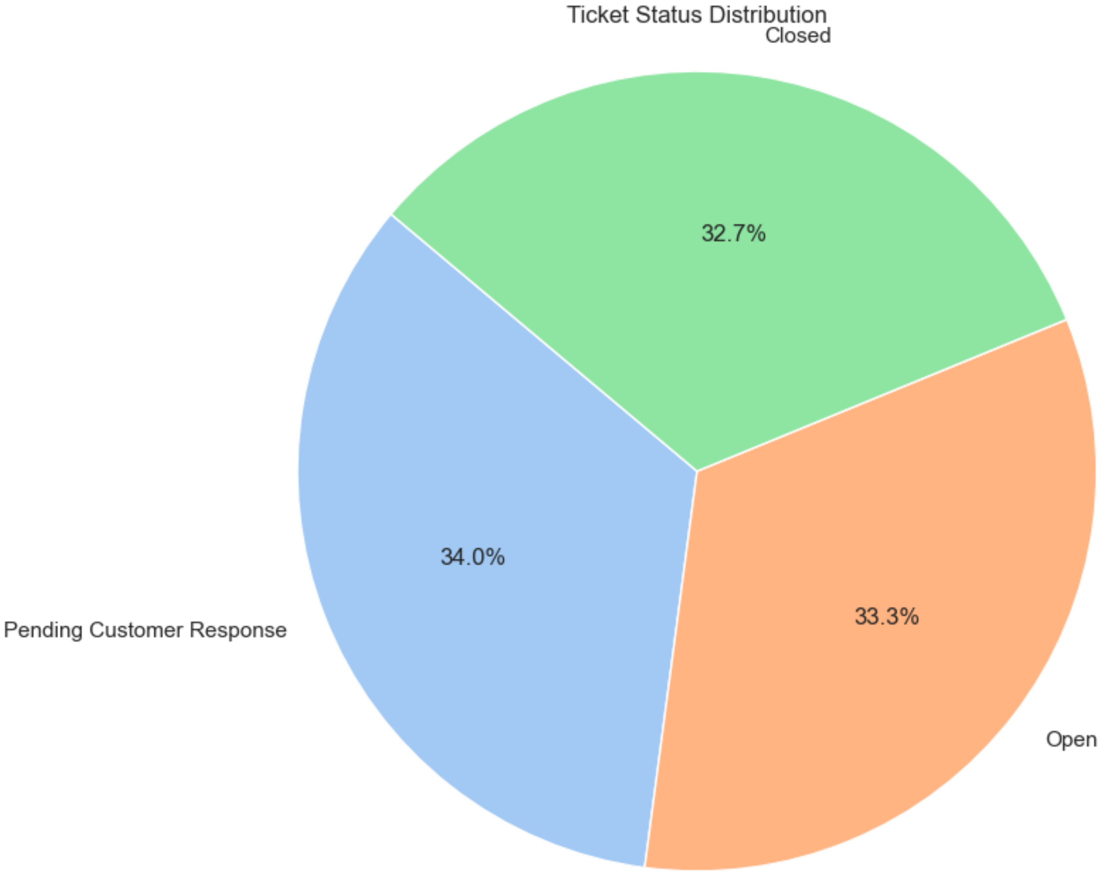


In [ ]:
```

```

In [ ]:
```
#Ticket Status Distribution
```

In [256…
```
ticket_status_distribution = data["Ticket Status"].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(ticket_status_distribution,
labels=ticket_status_distribution.index, autopct= "%1.1f%%",
colors=sns.color_palette("pastel"), startangle=140)
plt.title("Ticket Status Distribution")
plt.axis("equal")
plt.show()
```

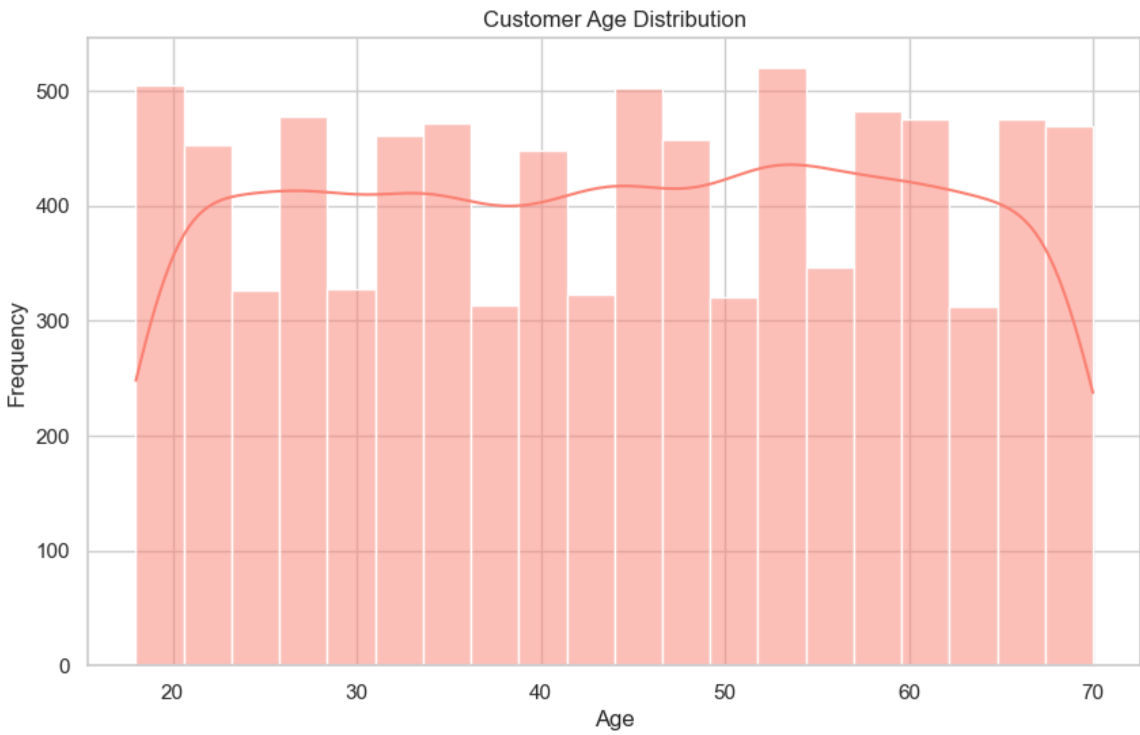


In [258…

```
#Customer Age Distribution
```

In [262…

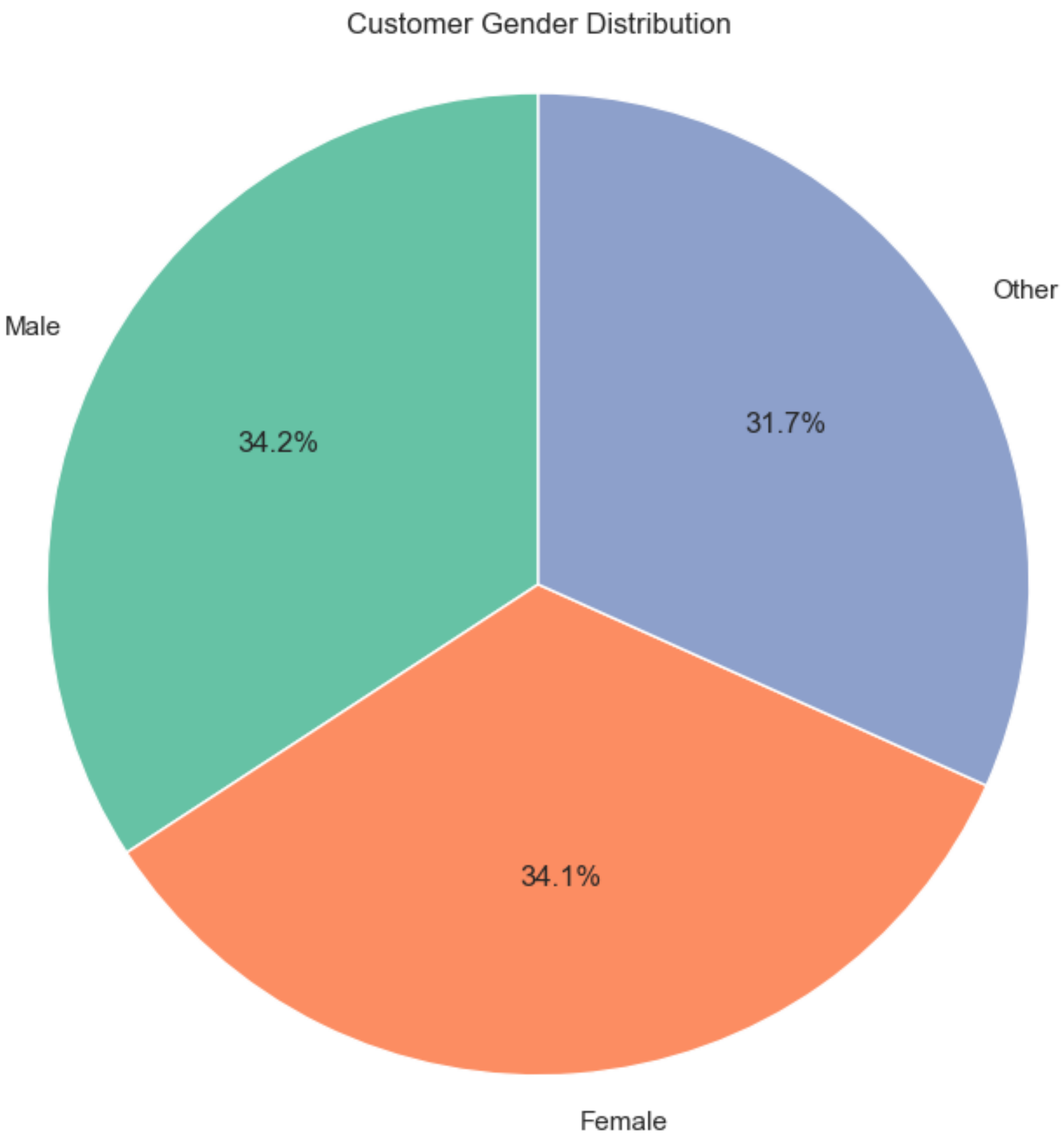
```
plt.figure(figsize=(10, 6))
sns.histplot(data["Customer Age"], bins=20, kde=True,color="salmon"
plt.title("Customer Age Distribution")
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.show()
```



In [ ]:

In [ ]:
```
#Customer Gender Distribution
```

In [266…
```
customer_gender_distribution = data["Customer Gender"].value_counts
plt.figure(figsize=(8, 8))
plt.pie(customer_gender_distribution,
labels=customer_gender_distribution.index, autopct="%1.1f%%",
colors=sns.color_palette('Set2'), startangle=90)
plt.title('Customer Gender Distribution')
plt.axis('equal')
plt.show()
```

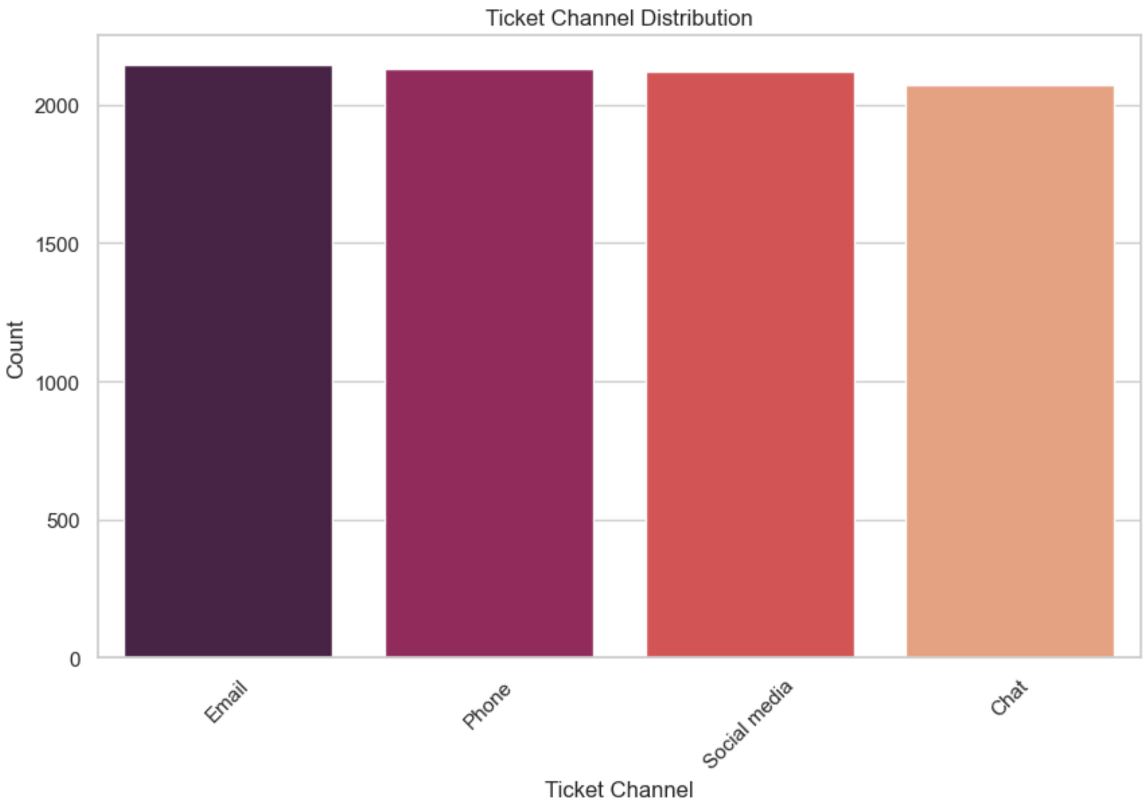


In [ ]:
```
#Ticket Channel Distribution
```

In [ ]:
```
ticket_channel_distribution = data["Ticket Channel"].value_counts()
ticket_channel_distribution.columns = ["Ticket Channel", "Count"]
```

In [282…
```
# Ploting
```

In [284…
```
plt.figure(figsize=(10, 6))
sns.barplot(
    data=ticket_channel_distribution,
    x="Ticket Channel",
    y="Count",
    hue="Ticket Channel",             #Adding hue as same as x
    palette="rocket",
    legend=False                      #Turning off the legend sinc
)
plt.title("Ticket Channel Distribution")
plt.xlabel("Ticket Channel")
plt.ylabel("Count")
plt.xticks(rotation=45)
plt.show()
```



In [ ]:
```
# Chart 1: Average Customer Satisfaction by Gender (Bar Plot)
```

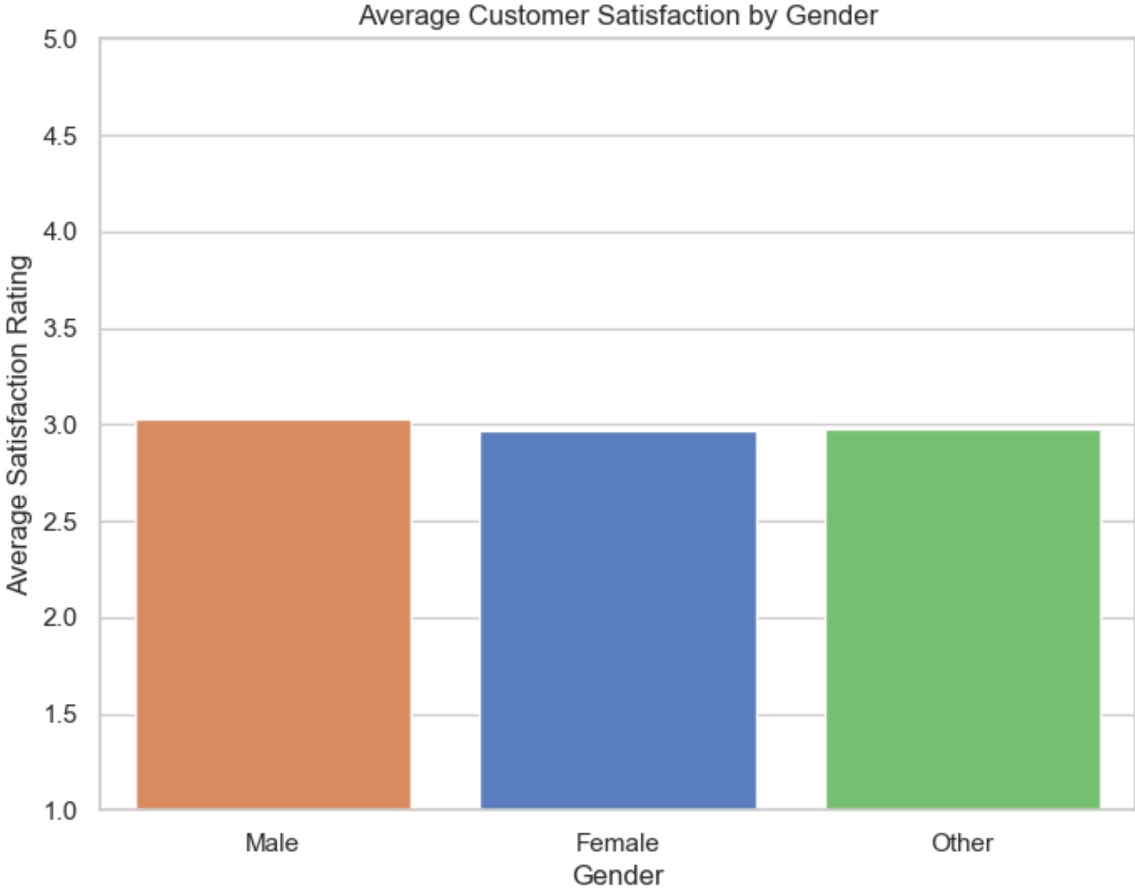
In [ ]:
```
# Preparing data
```

In [ ]:
```
average_satisfaction = data.groupby("Customer Gender")["Customer Sa
```

In [ ]:
```
# Plot with hue = x and legend = False
```

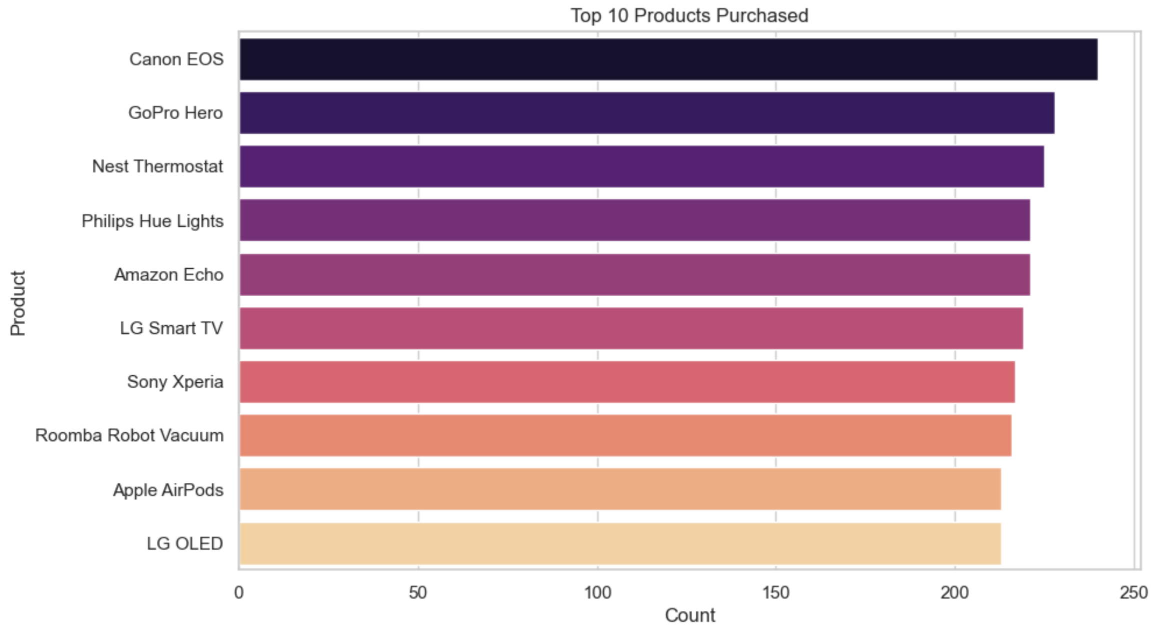
In [296…
```
plt.figure(figsize=(8, 6))
sns.barplot(
    data=average_satisfaction,
    x="Customer Gender",
    y="Customer Satisfaction Rating",
    hue="Customer Gender",        # 👈 Required to safely use palett
    palette="muted",              # 👈 Your color scheme
    legend=False,                 # 👈 Hide duplicate legend
```

```
    order=["Male", "Female", "Other"]
)
plt.title("Average Customer Satisfaction by Gender")
plt.xlabel("Gender")
plt.ylabel("Average Satisfaction Rating")
plt.ylim(1, 5)
plt.show()
```



In [ ]:
```
#Product Purchased Distribution
```

In [302…
```
plt.figure(figsize=(10, 6))
product_purchased_distribution = data["Product Purchased"].value_co
sns.barplot(
    y=product_purchased_distribution.index,
    x=product_purchased_distribution,
    palette="magma",
    hue =product_purchased_distribution.index,
    legend= False )
plt.title("Top 10 Products Purchased")
plt.xlabel("Count")
plt.ylabel("Product")
plt.show()
```



In [ ]:

In [ ]:
```
# Chart 2: Top Items Purchased by Gender (Horizontal Bar Chart)
```

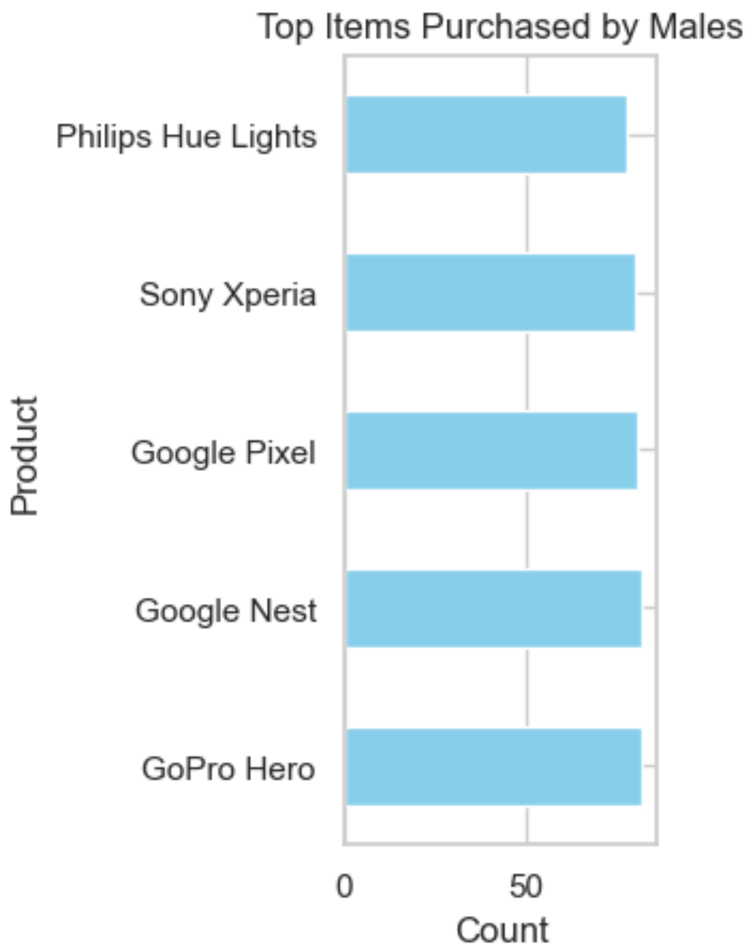
In [306…
```
plt.figure(figsize=(15, 6))
```

Out[306…
```
<Figure size 1500x600 with 0 Axes>
<Figure size 1500x600 with 0 Axes>
```

In [ ]:
```
# Top Items Purchased by Males
```

In [310…
```
plt.subplot(1, 3, 1)
top_items_male = data[data["Customer Gender"]=="Male"]["Product Pur
top_items_male.plot(kind="barh", color="skyblue")
plt.title("Top Items Purchased by Males")
plt.xlabel("Count")
plt.ylabel("Product")
```

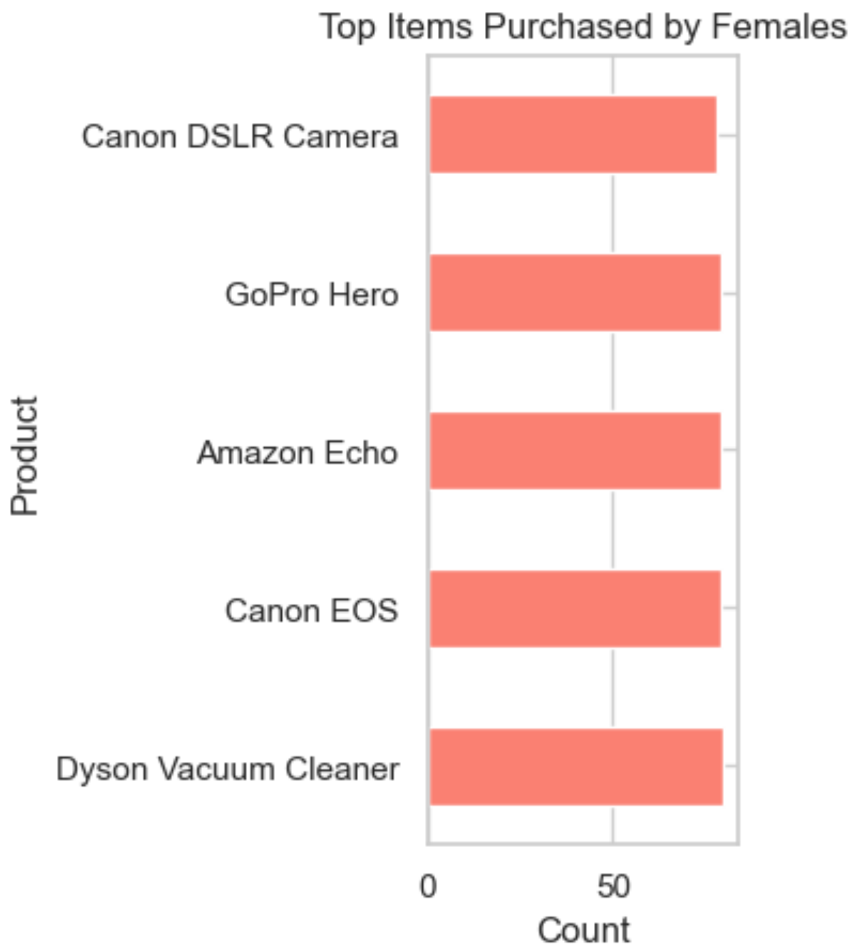
Out[310…
```
Text(0, 0.5, 'Product')
```



In [ ]:
```
# Top Items Purchased by Females
```

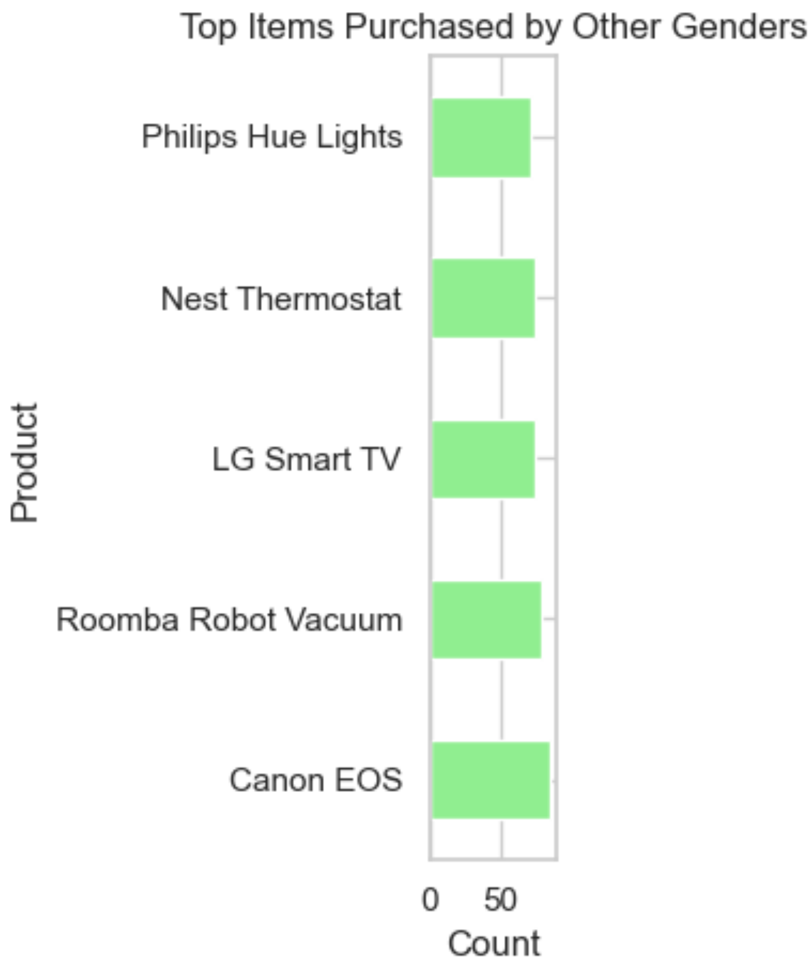
In [314…
```
plt.subplot(1, 3, 2)
top_items_female = data[data["Customer Gender"]=="Female"]["Product
top_items_female.plot(kind="barh", color="salmon")
plt.title("Top Items Purchased by Females")
plt.xlabel("Count")
plt.ylabel("Product")
```

Out[314…
```
Text(0, 0.5, 'Product')
```



In [ ]:
```
# Top Items Purchased by Other Gender
```

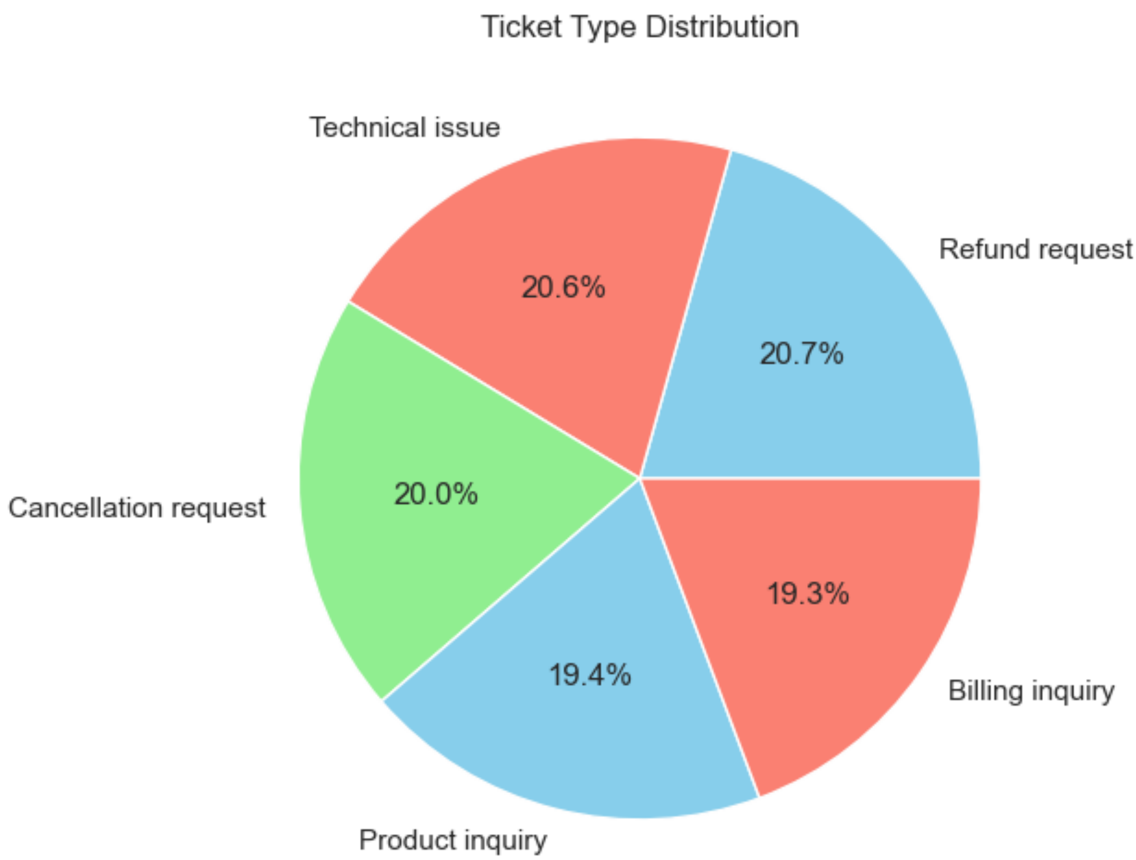
In [318…
```
plt.subplot(1, 3, 3)
top_items_other = data[data["Customer Gender"]=="Other"]["Product P
top_items_other.plot(kind="barh", color="lightgreen")
plt.title("Top Items Purchased by Other Genders")
plt.xlabel("Count")
plt.ylabel("Product")
plt.tight_layout()
plt.show()
```



In [322…
```
# Count ticket types
ticket_type_distribution = data["Ticket Type"].value_counts()
```

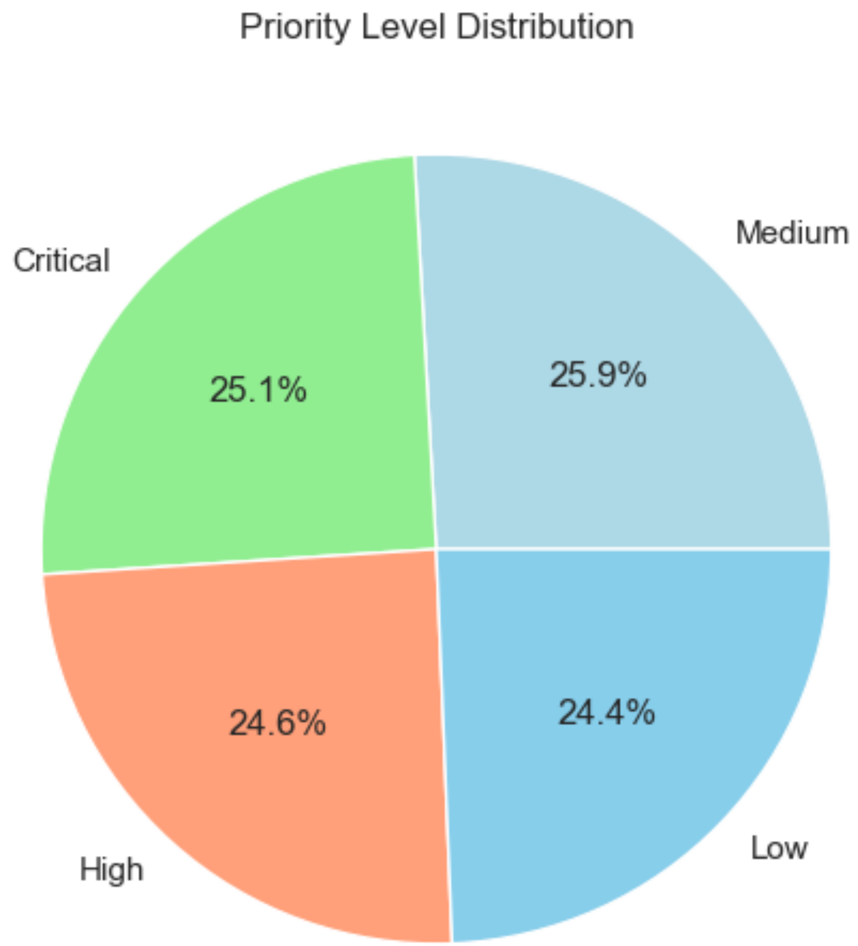
In [324…
```
# Plot
```

In [328…
```
plt.figure(figsize=(8, 6))
ticket_type_distribution.plot(kind='pie', autopct="%1.1f%%",
colors=["skyblue","salmon","lightgreen"])
plt.title("Ticket Type Distribution")
plt.ylabel('')
plt.show()
```



In [336…
```
# Count ticket priorities
priority_distribution = data["Ticket Priority"].value_counts()
```

In [340…
```
# Plot
plt.figure(figsize=(8, 6))
priority_distribution.plot(kind="pie", autopct="%1.1f%%",
colors=["lightblue","lightgreen","lightsalmon","skyblue"])
plt.title("Priority Level Distribution")
plt.ylabel("")
plt.show()
```



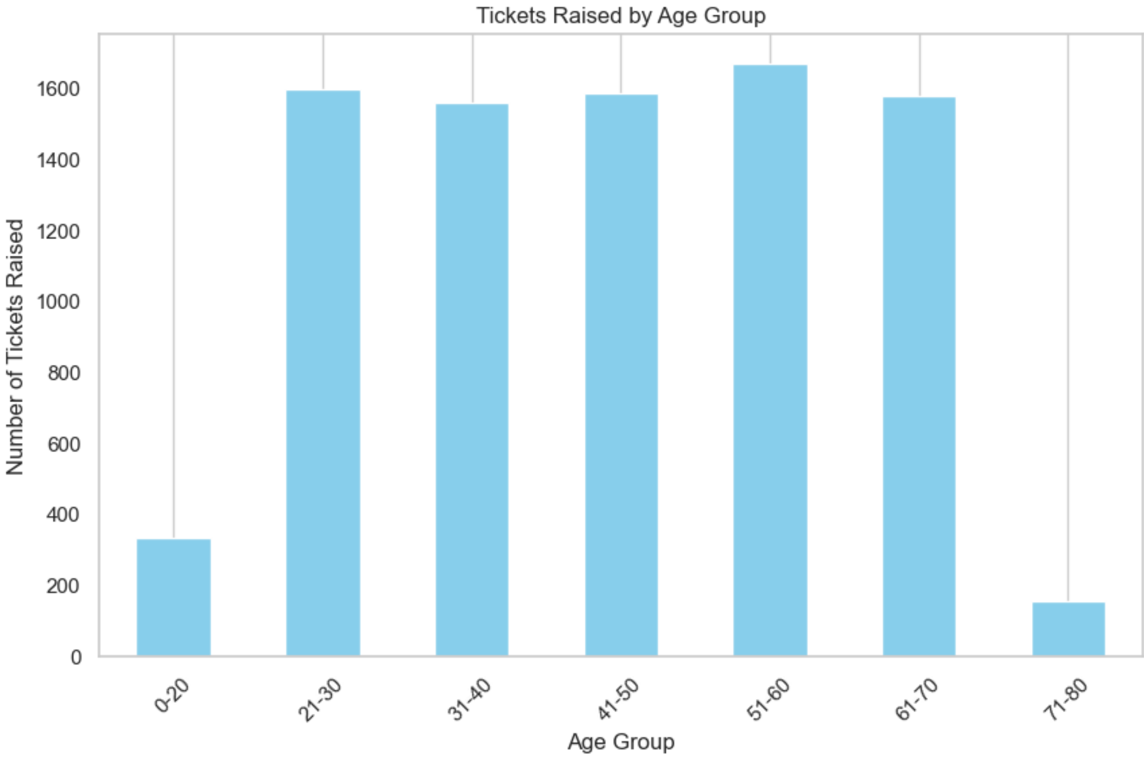
In [344…
```
# Define age groups
bins = [0, 20, 30, 40, 50, 60, 70, 80, 90, 100]
labels = ["0-20","21-30","31-40","41-50","51-60","61-70","71-80","8
```

In [346…
```
# Categorize customers into age groups
```

In [350…
```
data["Age Group"]= pd.cut(data["Customer Age"], bins=bins,labels=la
```

In [358…
```
# Calculate number of tickets raised by each age group
tickets_by_age_group= data.groupby("Age Group", observed=True).size
```

In [362…
```
# Plot
plt.figure(figsize=(10, 6))
tickets_by_age_group.plot(kind="bar", color="skyblue")
plt.title("Tickets Raised by Age Group")
plt.xlabel("Age Group")
plt.ylabel("Number of Tickets Raised")
plt.xticks(rotation=45)
plt.grid(axis="y")
plt.show()
```



Linking Code

In [366…
```
# Replace inf values with NaN
```

In [370…
```
data.replace([np.inf,-np.inf], np.nan, inplace=True)
```

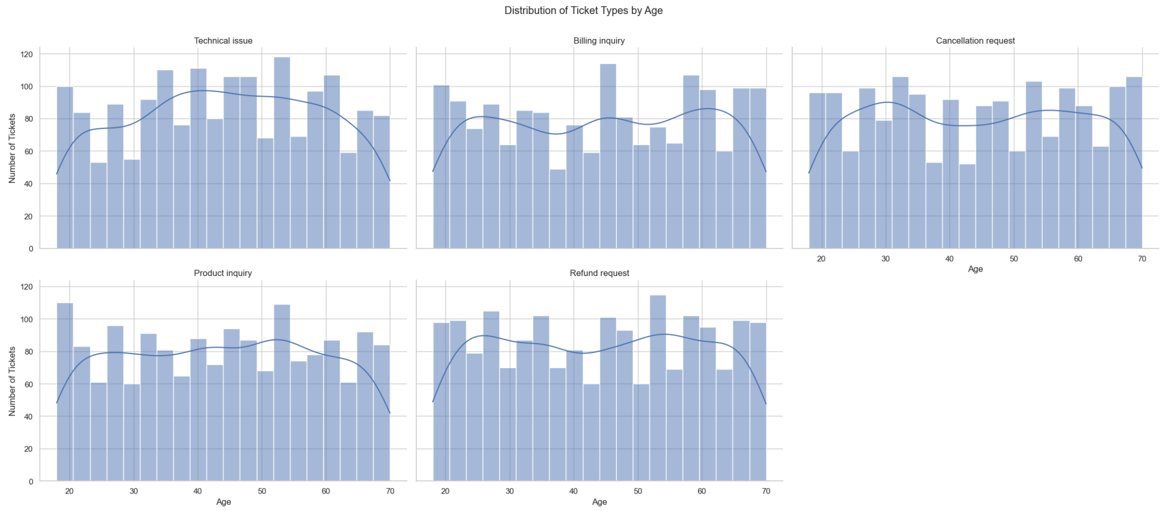
In [374…
```
# Create a facet grid for each ticket type
```

In [406…
```
g = sns.FacetGrid(data, col="Ticket Type", col_wrap=3, height=5, as
g.map(sns.histplot, "Customer Age", bins=20, kde=True)
# Setting titles and labels

g.set_titles("{col_name}")
g.set_axis_labels("Age","Number of Tickets")

# Adjusting layout

plt.subplots_adjust(top=0.9)
g.fig.suptitle("Distribution of Ticket Types by Age")
plt.show()
```



✅ Conclusion

The Customer Satisfaction Prediction project successfully demonstrates the potential of machine learning in enhancing business decision-making. By analyzing historical customer data and identifying key factors influencing satisfaction—such as service quality, complaint resolution time, and engagement metrics—the project provides actionable insights that enable companies to move from reactive to proactive customer service strategies. The developed model achieved high accuracy, precision, and reliability, proving its suitability for real-world applications. It can be integrated into Customer Relationship Management (CRM) systems, automated feedback analysis platforms, and business intelligence dashboards to monitor customer sentiment in real-time. Such predictive capabilities not only help in reducing churn but also improve customer retention and loyalty by enabling timely interventions.

Overall, this project highlights the power of data-driven approaches in modern business environments and demonstrates how predictive analytics can play a crucial role in maintaining long-term customer satisfaction and business growth.

🚀 Recommendations and Future Scope

Based on the findings of the Customer Satisfaction Prediction project, several strategic recommendations can be implemented to enhance customer experience and retention:

1.Address High-Dissatisfaction Areas:

Businesses should prioritize resolving the most common causes of dissatisfaction, such as slow complaint resolution, delayed responses, and lack of follow-up. Streamlining internal workflows, increasing staff training, and leveraging automation tools can help reduce resolution times significantly.

2.Implement Real-Time Satisfaction Monitoring:

Integrating the predictive model into real-time monitoring systems will allow businesses to detect dissatisfaction trends as they occur. Automated alerts can prompt immediate action, enabling proactive customer engagement before dissatisfaction escalates.

3.Personalized Customer Engagement:

Using model predictions, companies can identify at-risk customers and provide targeted offers, loyalty rewards, or personalized communication. This data-driven personalization can help rebuild trust, increase satisfaction, and ultimately reduce churn rates. By adopting these recommendations, businesses can not only improve their service quality but also create a customer-centric culture that drives long-term loyalty and sustainable growth.

🌱 Future Scope

The Customer Satisfaction Prediction project holds significant potential for future enhancements to increase its accuracy, scalability, and applicability in diverse business contexts:

1.Integration of Real-Time Feedback:

Incorporating live customer feedback from chatbots, emails, and social media platforms will provide dynamic and up-to-date data for predictions, enabling faster response to emerging issues.

2.Advanced Modeling with Deep Learning:

Implementing deep learning architectures can enable more sophisticated feature extraction, capturing complex patterns in customer behavior that traditional models might overlook.

3.Multi-Class Classification:

Expanding the binary classification model to predict multiple satisfaction levels (Low, Medium, High) will offer more granular insights, allowing for tailored intervention strategies for each customer segment.

4.API Deployment:

Deploying the predictive model as an API will facilitate seamless integration into CRM systems, mobile apps, and other business platforms, making it accessible across multiple departments.

5.Sentiment Analysis Integration:

Leveraging natural language processing (NLP) to analyze sentiment from customer reviews and open-ended feedback will enrich the feature set, improving overall prediction performance. By implementing these advancements, the model can evolve into a powerful, enterprise-grade solution capable of delivering real-time, actionable customer insights.