Problem Statement : **Data Driven Approach to Payment Fraud Detection**

Fraudulent transactions are a major concern; they result in significant financial losses and, more importantly, a loss of consumer trust. The purpose of this research is to thoroughly analyze a huge dataset for patterns and correlations between transaction variables and the possibility of fraud. This research will primarily focus on developing a robust predictive model capable of detecting fraudulent online transactions. It would classify the transactions as fraudulent or not based on numerous attributes in the Transaction Dataset.

```
# importing required libraries
import zipfile
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Use the below command only if you face error during unipping data.

```
!rm -rf /content/.kaggle/
!rm -rf /.kaggle/
!rm -rf /root/.kaggle/
```

The below code does the work of downloading a dataset from Kaggle, which is on fraudulent transactions, extracting, and preparing them for analysis. This includes the creation of necessary authentication with the Kaggle API, downloading of the dataset, and loading into a Python environment where further analysis or processing will take place. These steps are absolutely key to the successful analysis of fraudulent transactions in developing insights or models which could be used in fraud detection and mitigation within payments.

```
import os
os.makedirs("/content/.kaggle/")

import json

token = {"username":"adityaashokthakare","key":"637d87331a545d565a6a00a70cd1a9d6"}
with open('/content/.kaggle/kaggle.json', 'a+') as file:
    json.dump(token, file)

import shutil
os.makedirs("/.kaggle/")
src="/content/.kaggle/kaggle.json"
des="/.kaggle/kaggle.json"
shutil.copy(src,des)


os.makedirs("/root/.kaggle/")
!cp /content/.kaggle/kaggle.json ~/.kaggle/kaggle.json

!kaggle config set -n path -v /content

!kaggle datasets download -d shriyashjagtap/fraudulent-e-commerce-transactions
```

```
Warning: Your Kaggle API key is readable by other users on this system! To fix this, you can run 'chmod 600 /root/.kaggle/kaggle.jsc
- path is now set to: /content
Warning: Your Kaggle API key is readable by other users on this system! To fix this, you can run 'chmod 600 /root/.kaggle/kaggle.jsc
Dataset URL: https://www.kaggle.com/datasets/shriyashjagtap/fraudulent-e-commerce-transactions
License(s): MIT
fraudulent-e-commerce-transactions.zip: Skipping, found more recently modified local copy (use --force to force download)
```

Unzipping the Downloaded Dataset

```
# Unzip the dataset
zip_path = "/content/datasets/shriyashjagtap/fraudulent-e-commerce-transactions/fraudulent-e-commerce-transactions.zip"
with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall("/content/")

# List extracted files
extracted_files = os.listdir("/content/")
print(extracted_files)

# Load the dataset into a pandas dataframe (replace with the correct file name)
dataset_path = "/content/Fraudulent_E-Commerce_Transaction_Data.csv"  # Use the actual file name from the extracted files
df = pd.read_csv(dataset_path)
df3 = df
```

```
# Display the first few rows of the dataset
df.head()
```

```
['.config', 'datasets', 'Fraudulent_E-Commerce_Transaction_Data_2.csv', '.kaggle', 'Fraudulent_E-Commerce_Transaction_Data.csv', 'sa
```

	Transaction ID	Customer ID	Transaction Amount	Transaction Date	Payment Method	Product Category	Quantity	Customer Age	Customer Location	Device Used	IP Addre
0	15d2e414-8735-46fc-9e02-80b472b2580f	d1b87f62-51b2-493b-ad6a-77e0fe13e785	58.09	2024-02-20 05:58:41	bank transfer	electronics	1	17	Amandaborough	tablet	212.195.49.1
1	0bfee1a0-6d5e-40da-a446-d04e73b1b177	37de64d5-e901-4a56-9ea0-af0c24c069cf	389.96	2024-02-25 08:09:45	debit card	electronics	2	40	East Timothy	desktop	208.106.249.1
2	e588eef4-b754-468e-9d90-d0e0abfc1af0	1bac88d6-4b22-409a-a06b-425119c57225	134.19	2024-03-18 03:42:55	PayPal	home & garden	2	22	Davismouth	tablet	76.63.88.2
3	4de46e52-60c3-49d9-be39-636681009789	2357c76e-9253-4ceb-b44e-ef4b71cb7d4d	226.17	2024-03-16 20:41:31	bank transfer	clothing	5	31	Lynnberg	desktop	207.208.171.
4	074a76de-fe2d-443e-a00c-f044cdb68e21	45071bc5-9588-43ea-8093-023caec8ea1c	121.53	2024-01-15 05:08:17	bank transfer	clothing	2	51	South Nicole	tablet	190.172.14.1

Question 1 Onkar Ramade (50604538) -

1. How does transaction behaviour-as represented by amount, frequency, and time of day-relate to the incidence of fraud in e-commerce transactions?

Significance : This question focuses on transaction behaviours, crucial in ascertaining fraud dynamics. Knowing how specific characteristics of a transaction relate to fraud might help guide the design in fraud detection systems that flag suspicious activities.

Possible Hypothesis: The higher the amount of money transacted, the greater the likelihood of fraud.

Question 2 Onkar Ramade (50604538) -

2. What are demographic factors, including but not limited to age, location, and method of payment, that signal fraudulent e-commerce transactions?

Importance: By searching out the demographic influences, teams can find patterns in subsets of customers that could elude fraud detection efforts in a more effective and specific manner.

Potential Hypotheses: Younger customers are most likely to be perpetrators of fraudulent transactions when compared to older customers.

Question 1 Sourabh Kodag (50606796) -

1. Is there fraud transaction in uneven hours ?

Rationale Behind the Hypothesis:

Behavioural Patterns: Customers' behaviour also depends on the time of day. Overnight transactions, for example, have less oversight and more anonymity and therefore hold greater potential for fraudulent activities.

Operational Factors: Most businesses will be having a skeleton crew during very late or very early hours. These periods will therefore mean that transactions are not monitored as carefully and there is less support to react quickly to suspicious activities.

Importance:

Identifying Behavioural Patterns: Analyzing the time of fraud transactions will relate to the behavioural patterns of fraudsters. Understanding when fraud is most likely to occur means there are familiar tactics against which organizations can adapt their defenses.

Operational Improvement: Peak hours of fraud help the organization in optimizing its resources. For instance, if data reflected that most frauds happened after midnight, a company would increase monitoring and fraud detection at such hours to avoid any possible loss.

Question 2 Sourabh Kodag (50606796) -

2. Is there a relation between account age and fraud ?

Hypothesis Rationale

Lack of Transaction History: New accounts lack transaction history, and no pattern can be established to indicate a trend in legitimate behavior. Fraudsters are normally taking advantage of the lack of history since there are no prior behaviors to which one could compare when assessing legitimacy.

Vulnerability to Exploitation: In general, fraudsters may target new accounts since they are less monitored. And most probably, they would have been opened without strict identity verification processes in place. This makes newer accounts the favorite target for fraudsters. likecopy

Importance:

Changes in Business Practice: The findings have many implications for wider business practices, including marketing strategies and customer engagement. For instance, organizations can make promotional offers that incentivize customers to engage when they are on the site, but security measures will be in place.

Supporting Regulatory Compliance:Many industries have certain regulations that call for them to put in place methods for fraud prevention. It would also be of significance to an organization in case there are risks related to new accounts to also note them to ensure compliance with the set regulations to avoid probable penalties.

Question 1 Aditya Thakare (50608812) -

Question 1: "Is there a correlation between the customer age and the likelihood of fraud?"

Why This Question is significant and leading to our object: Fraud Detection: Understanding the relationship between customer age and fraud can inform better risk assessment models. If fraudulent activities are detected among a population with younger age groups more frequently, then a business could institute additional verification steps for these transactions. Feature Engineering: This customer age can be a critical feature in fraud detection algorithms, especially by enabling the algorithm to create risk profiles. Market Strategies: Knowledge of the age-related pattern of fraud can help organizations in framing appropriate marketing strategies and fraud prevention policy.

Question 2 Aditya Thakare (50608812) -

**Question 2: **"Is there a correlation between the payment method used and the likelihood of fraud?"

How It Leads to Our Objective: Feature Engineering: Knowing the correlations between fraud and means of payment helps decide which features are most appropriate for fraud detection algorithms. For instance, if credit cards bear the brunt of fraud, then that feature would be amplified in the model. Fraud Prevention: The ability to identify the most risky forms of payments will allow businesses to focus fraud prevention measures on those forms of payments and reduce the overall incidence of fraud. Significance of the Question: Security Measures: The associations between the mode of payments and fraud assist firms in implementing extra security measures around the risky payment types. **Cost Efficiency:** In spotting fraud-related modes of payments, the companies can effectively allocate their resources to further the fraud detection and prevention programs. **Customer Trust:** This enhances customer trust as, with greater clarity on fraudulent ways of making payments, businesses can advise on the use of safer alternatives like PayPal or bank transfers.

Data Cleaning

Handle missing values: In this step, we check for missing values and remove them if found.

```
df.isnull().sum()
df=df.dropna()       #removes rows with null values
df1 = df
```

```
df.info()            #metadata
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1472952 entries, 0 to 1472951
Data columns (total 16 columns):
 #   Column              Non-Null Count    Dtype  
---  ------              --------------    -----  
 0   Transaction ID      1472952 non-null  object 
 1   Customer ID         1472952 non-null  object 
 2   Transaction Amount  1472952 non-null  float64
 3   Transaction Date    1472952 non-null  object 
 4   Payment Method      1472952 non-null  object 
 5   Product Category    1472952 non-null  object 
 6   Quantity            1472952 non-null  int64  
 7   Customer Age        1472952 non-null  int64  
 8   Customer Location   1472952 non-null  object 
 9   Device Used         1472952 non-null  object
 10  IP Address          1472952 non-null  object
 11  Shipping Address    1472952 non-null  object
 12  Billing Address     1472952 non-null  object
 13  Is Fraudulent       1472952 non-null  int64
 14  Account Age Days    1472952 non-null  int64
 15  Transaction Hour    1472952 non-null  int64
dtypes: float64(1), int64(5), object(10)
memory usage: 179.8+ MB
```

Correct Data Types: In this step for the date column, we convert it to datetime format if not already.

```
df['Transaction Date'] = pd.to_datetime(df['Transaction Date'])
```

Removing undesired duplicate entries: Transactions should be unique as duplicate transactions could skew fraud detection Checking for duplicates based on Transaction ID to ensure data integrity.

```
df.duplicated(subset=['Transaction ID']).sum() #checking duplicate Transaction IDs
```

```
0
```

Sometimes, addresses have slight variations (like different abbreviations). A string standardization function can help clean up Shipping Address and Billing Address.

```
#converting to lower-case
df['Shipping Address'] = df['Shipping Address'].str.lower().str.strip()
df['Billing Address'] = df['Billing Address'].str.lower().str.strip()
```

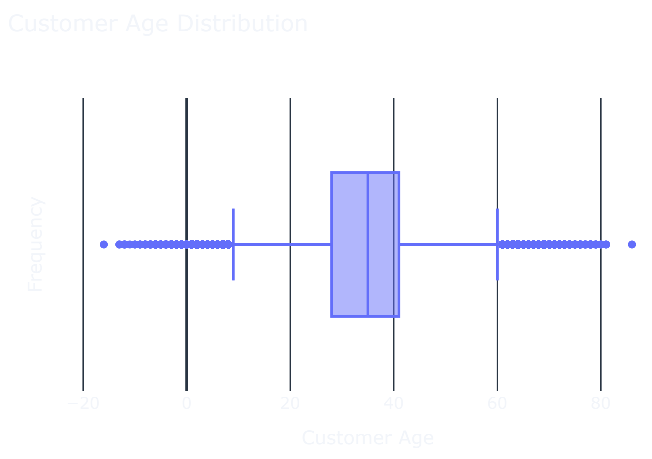
Adding necessary features: The transaction date can be broken doen into day of the week which may be useful for detecting fraud patterns.

```
df['Transaction Day'] = df['Transaction Date'].dt.weekday
df.head()
```

	Transaction ID	Customer ID	Transaction Amount	Transaction Date	Payment Method	Product Category	Quantity	Customer Age	Customer Location	Device Used	IP Addre
0	15d2e414-8735-46fc-9e02-80b472b2580f	d1b87f62-51b2-493b-ad6a-77e0fe13e785	58.09	2024-02-20 05:58:41	bank transfer	electronics	1	17	Amandaborough	tablet	212.195.49.1
1	0bfee1a0-6d5e-40da-a446-d04e73b1b177	37de64d5-e901-4a56-9ea0-af0c24c069cf	389.96	2024-02-25 08:09:45	debit card	electronics	2	40	East Timothy	desktop	208.106.249.1
2	e588eef4-b754-468e-9d90-d0e0abfc1af0	1bac88d6-4b22-409a-a06b-425119c57225	134.19	2024-03-18 03:42:55	PayPal	home & garden	2	22	Davismouth	tablet	76.63.88.2
3	4de46e52-60c3-49d9-be39-636681009789	2357c76e-9253-4ceb-b44e-ef4b71cb7d4d	226.17	2024-03-16 20:41:31	bank transfer	clothing	5	31	Lynnberg	desktop	207.208.171.
4	074a76de-fe2d-443e-a00c-f044cdb68e21	45071bc5-9588-43ea-8093-023caec8ea1c	121.53	2024-01-15 05:08:17	bank transfer	clothing	2	51	South Nicole	tablet	190.172.14.1

We identify observed irregularities in the customer age column:

```
import plotly.express as px

fig = px.box(data_frame=df,
             x="Customer Age",
             title="Customer Age Distribution",
             width=600, height=400,
             template="plotly_dark")

fig.update_layout(
    xaxis_title="Customer Age",
    yaxis_title="Frequency",
    showlegend=False
)
fig.show()
```



We observe there are some negative values. Assuming them as mistakes we replace them with their absolute values as below:

```
df['Customer Age'] = np.where(df['Customer Age'] < 0, np.abs(df['Customer Age']), df['Customer Age'])
```

We check if the shipping address and billing address are same, to detect possible fraudulent behaviour:

```
df["Is Address Match"] = (df["Shipping Address"] == df["Billing Address"]).astype(int) #marking 1 for same address and 0 for different
```

Reducing dataset size by downcasting: We reduce the dataset size by downcasting all integer and float values. Downcasting helps in reducing the dataset size without actually changing the original values. bold text

```
integer_cols = df.select_dtypes(include="int").columns   #selecting integer columns
float_cols = df.select_dtypes(include="float").columns   #selecting float columns

#downcasting
df[integer_cols] = df[integer_cols].apply(pd.to_numeric, downcast='integer')
df[float_cols] = df[float_cols].apply(pd.to_numeric, downcast='float')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1472952 entries, 0 to 1472951
Data columns (total 18 columns):
 #   Column              Non-Null Count    Dtype
---  ------              --------------    -----
 0   Transaction ID      1472952 non-null  object
 1   Customer ID         1472952 non-null  object
 2   Transaction Amount  1472952 non-null  float32
 3   Transaction Date    1472952 non-null  datetime64[ns]
 4   Payment Method      1472952 non-null  object
 5   Product Category    1472952 non-null  object
 6   Quantity            1472952 non-null  int8
 7   Customer Age        1472952 non-null  int8
 8   Customer Location   1472952 non-null  object
 9   Device Used         1472952 non-null  object
 10  IP Address          1472952 non-null  object
 11  Shipping Address    1472952 non-null  object
 12  Billing Address     1472952 non-null  object
 13  Is Fraudulent       1472952 non-null  int8
 14  Account Age Days    1472952 non-null  int16
 15  Transaction Hour    1472952 non-null  int8
 16  Transaction Day     1472952 non-null  int8
 17  Is Address Match    1472952 non-null  int8
dtypes: datetime64[ns](1), float32(1), int16(1), int8(6), object(9)
memory usage: 129.2+ MB
```

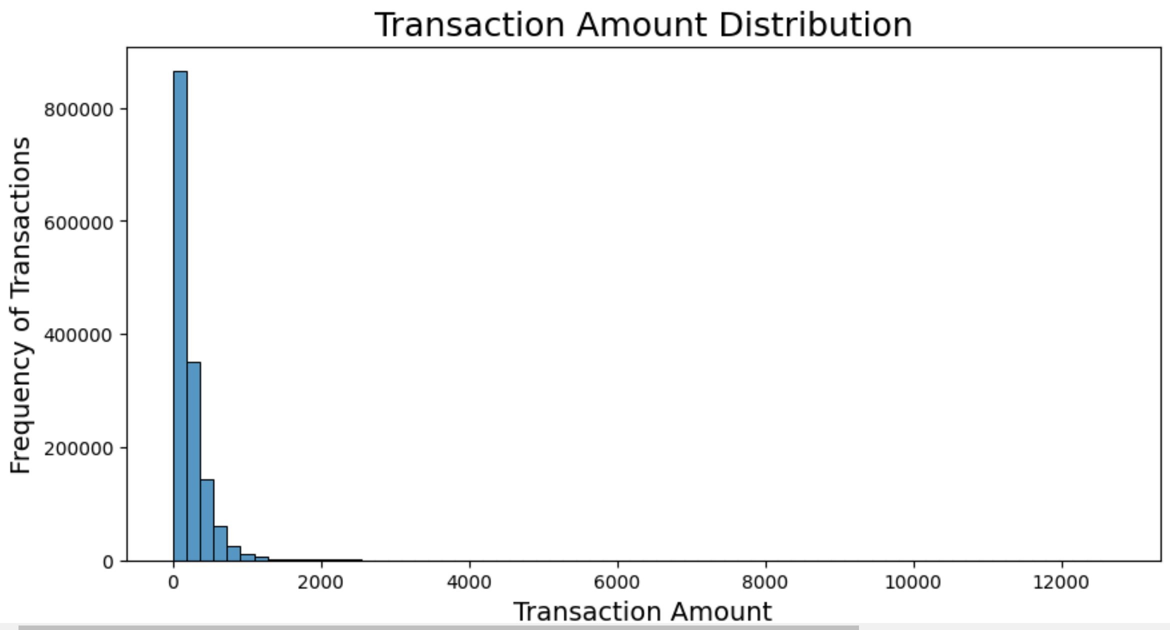
Thus, we observe our dataset size has significantly reduced by about 130MBs.

Hypothesis 1 (Onkar : 50604538): Does value of tranasaction increase the likelihood of fraudulent transactions ?

```
df['Transaction Amount'].describe()  # Checking for extreme values
```

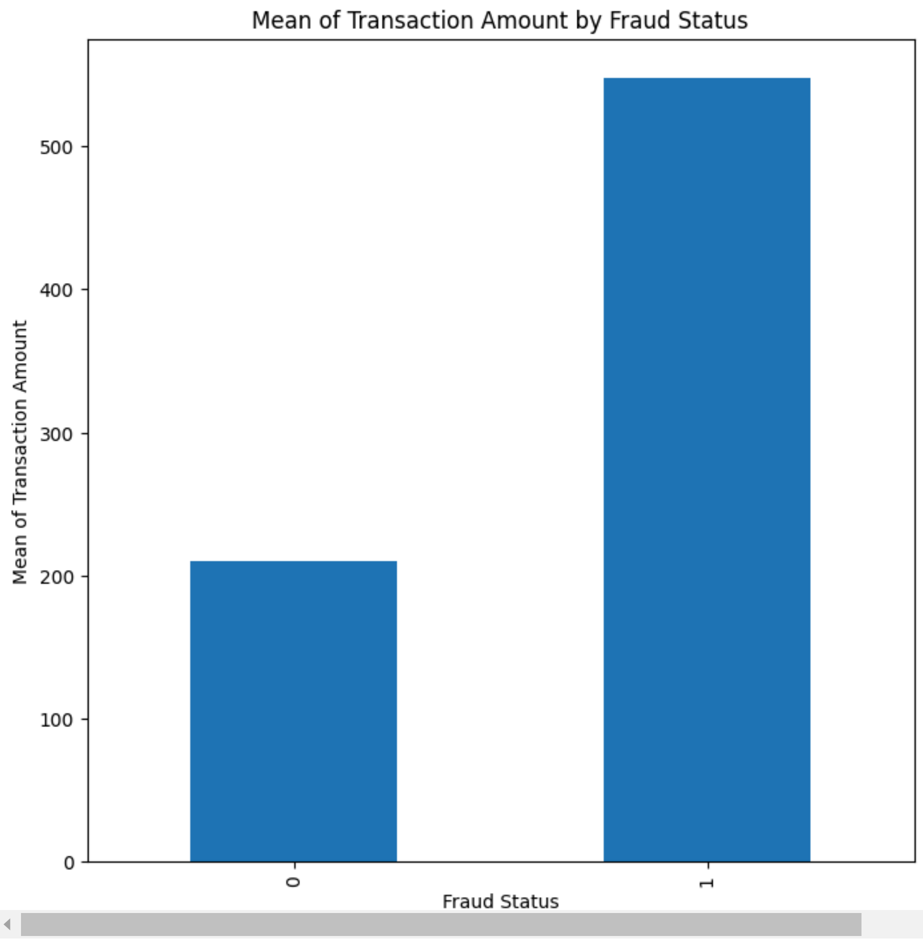
```
       Transaction Amount
count      1.472952e+06
mean       2.267682e+02
std        2.702478e+02
min        1.000000e+01
25%        6.861000e+01
50%        1.517600e+02
75%        2.960500e+02
max        1.270175e+04
```

```
plt.figure(figsize=(10, 5))

sns.histplot(df['Transaction Amount'], bins=70)

plt.title('Transaction Amount Distribution', fontsize=18)
plt.xlabel('Transaction Amount', fontsize=14)
plt.ylabel('Frequency of Transactions', fontsize=14)

plt.show()
```



The transaction amount bins between 0 to 1000 has the highest frequency

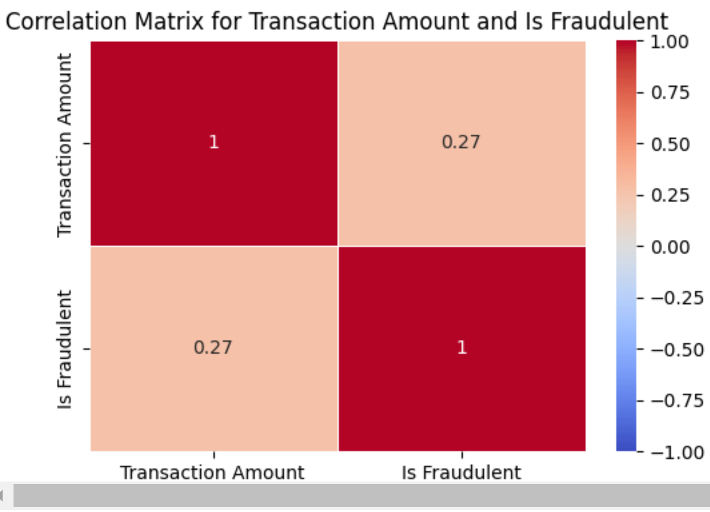
```
fraud_groups = df.groupby('Is Fraudulent')
feature_mean = fraud_groups['Transaction Amount'].mean()
plt.figure(figsize=(8, 8))
feature_mean.plot(kind='bar')
plt.xlabel('Fraud Status')
plt.ylabel('Mean of Transaction Amount')
plt.title('Mean of Transaction Amount by Fraud Status')
plt.show()
```



Mean Transaction Amount of Fraudulent Transaction is higher compared to legitimate transactions, which supports our hypothesis.

```
fraud_corr = df[['Transaction Amount', 'Is Fraudulent']].corr()
print(fraud_corr)

plt.figure(figsize=(6, 4))
sns.heatmap(fraud_corr, annot=True, cmap='coolwarm', linewidths=0.5, vmin=-1, vmax=1)
plt.title('Correlation Matrix for Transaction Amount and Is Fraudulent')
plt.show()
```

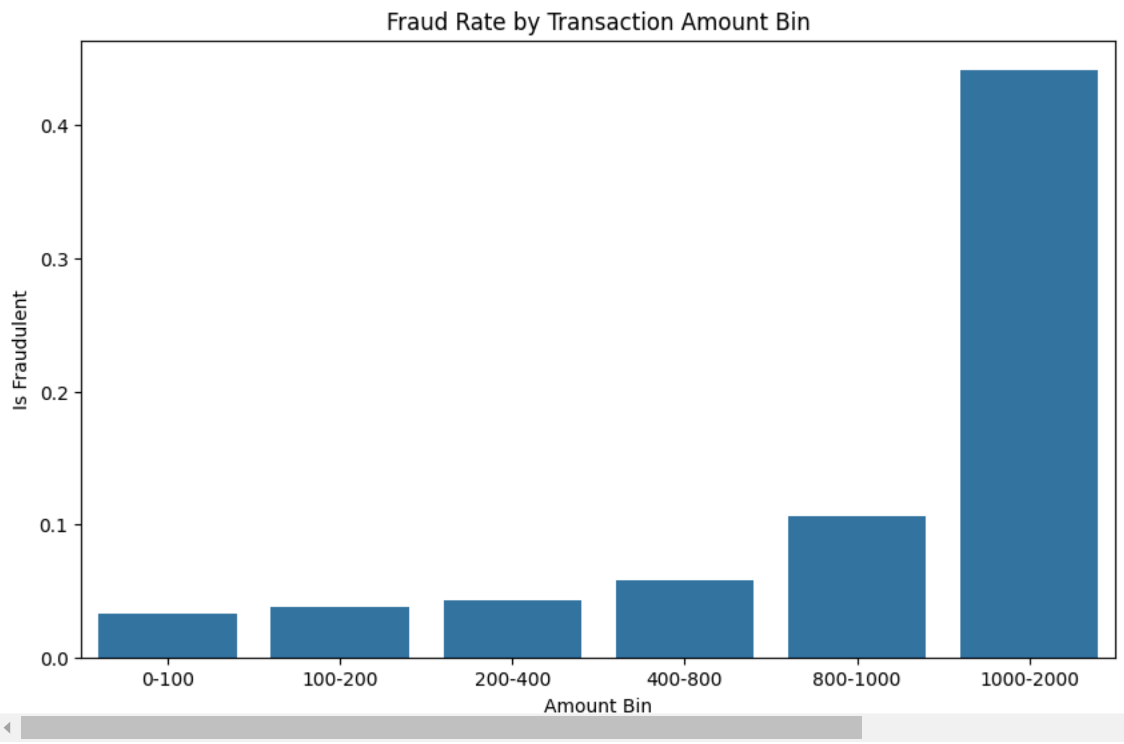
```
                    Transaction Amount  Is Fraudulent
Transaction Amount            1.000000       0.272766
Is Fraudulent                 0.272766       1.000000
```



The correlation output between Transaction Amount and Is Fraudulent shows a positive but weak correlation coefficient of 0.272766. The relationship suggests that higher transaction amounts are more likely to be fraudulent but the strength of the correlation is not very high.

It would still be helpful to plot by relationship between Transaction Amount and Fraud likelihood. We aanalyze this further by binning the transaction amount in multiple bins of transaction amount.

```
df['Amount Bin'] = pd.cut(df['Transaction Amount'], bins=[0, 100, 200, 400, 600, 1000, 2000], labels=['0-100', '100-200', '200-400', '4(
fraud_rate_by_amount_bin = df.groupby('Amount Bin')['Is Fraudulent'].mean().reset_index()

plt.figure(figsize=(10, 6))
sns.barplot(x='Amount Bin', y='Is Fraudulent', data=fraud_rate_by_amount_bin)
plt.title('Fraud Rate by Transaction Amount Bin')
plt.show()
```



From the graph we interpret that high value transaction bins have very high chances of fraud, compared to low and medium range bins. This supports our hypothesis that high-value transactions are more susceptible to fraud, likely because they offer higher potential rewards for the fraudster.

Handling the outliers in the Transaction Amount feature

```
Q1 = df['Transaction Amount'].quantile(0.25)
Q3 = df['Transaction Amount'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = df[(df['Transaction Amount'] < lower_bound) | (df['Transaction Amount'] > upper_bound)]
print("Number of outliers detected:", outliers.shape[0])
```

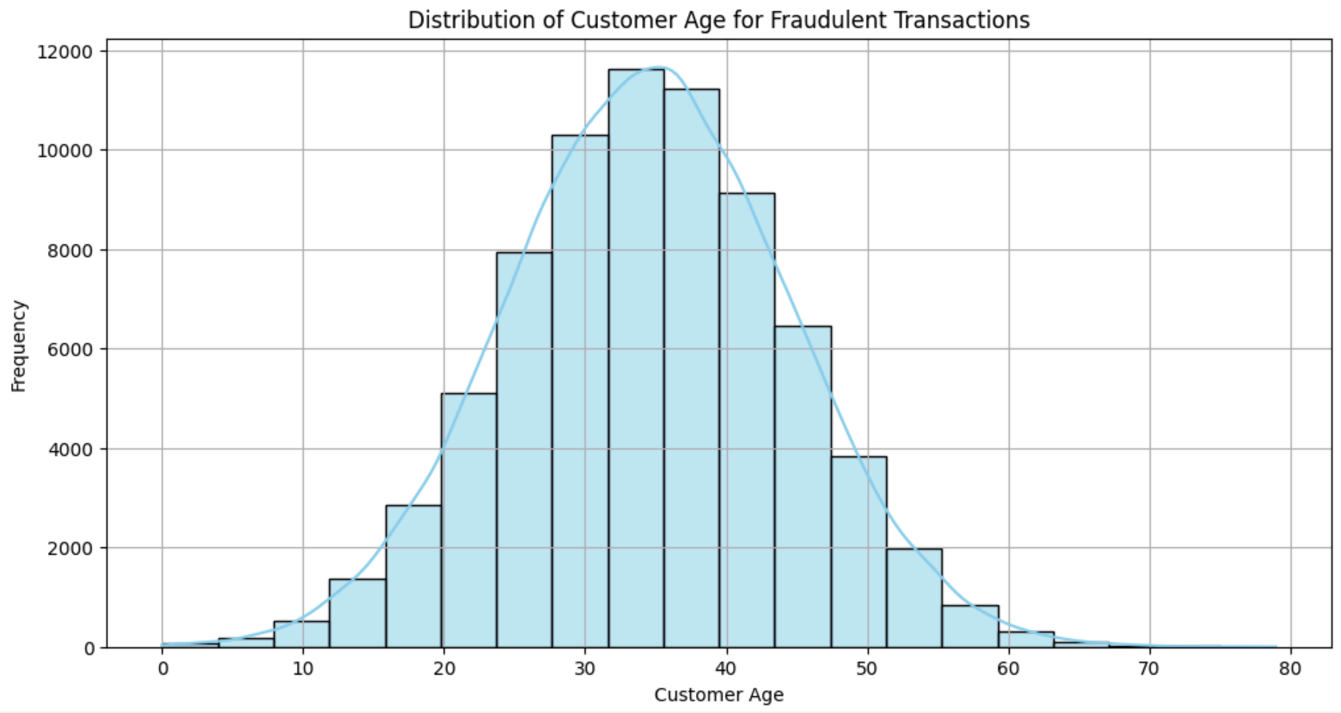
```
Number of outliers detected: 79180
```

Capping the outliers to upper and lower bound to limit their impact.

```
df['Transaction Amount'] = np.where(df['Transaction Amount'] > upper_bound, upper_bound, df['Transaction Amount'])
df['Transaction Amount'] = np.where(df['Transaction Amount'] < lower_bound, lower_bound, df['Transaction Amount'])
```

Hypothesis 2 : Do younger customers have a higher chance of commiting fraud ?

```
fraudulent_transactions = df[df['Is Fraudulent'] == 1]

plt.figure(figsize=(12, 6))
sns.histplot(fraudulent_transactions['Customer Age'], bins=20, kde=True, color='skyblue')

plt.title('Distribution of Customer Age for Fraudulent Transactions')
plt.xlabel('Customer Age')
plt.ylabel('Frequency')
plt.grid()
plt.show()
```

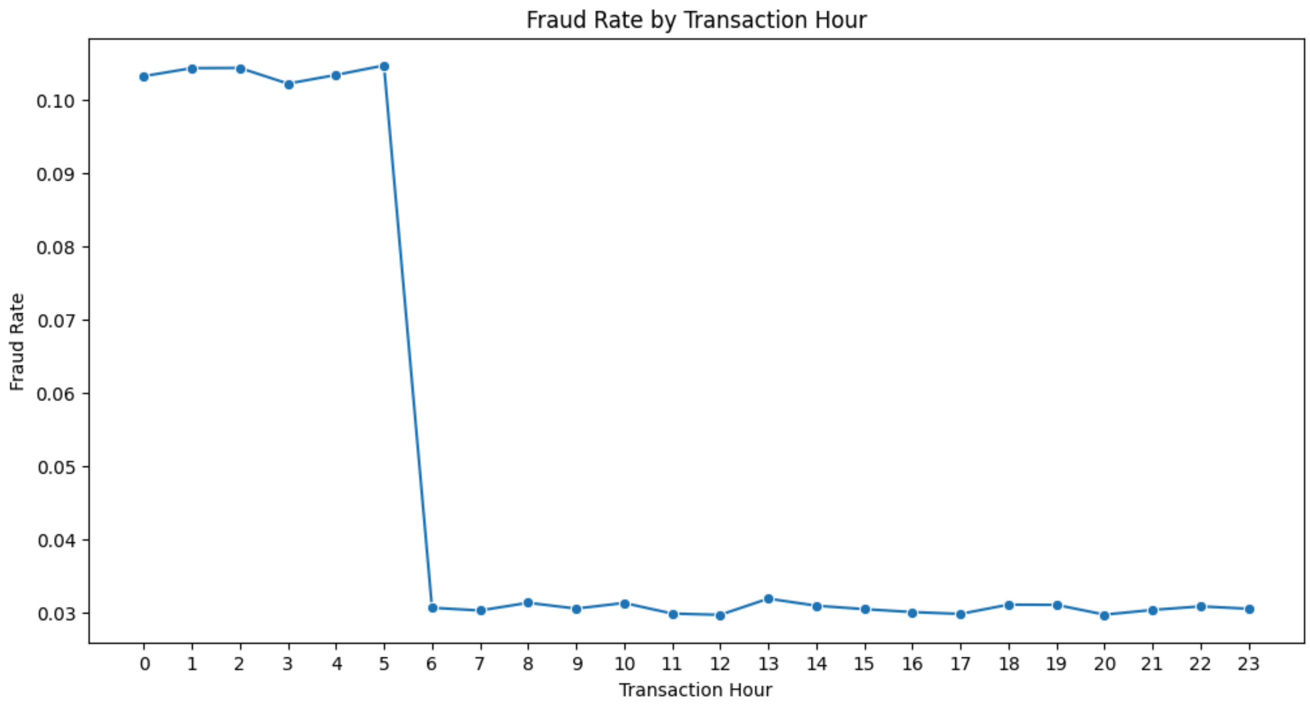


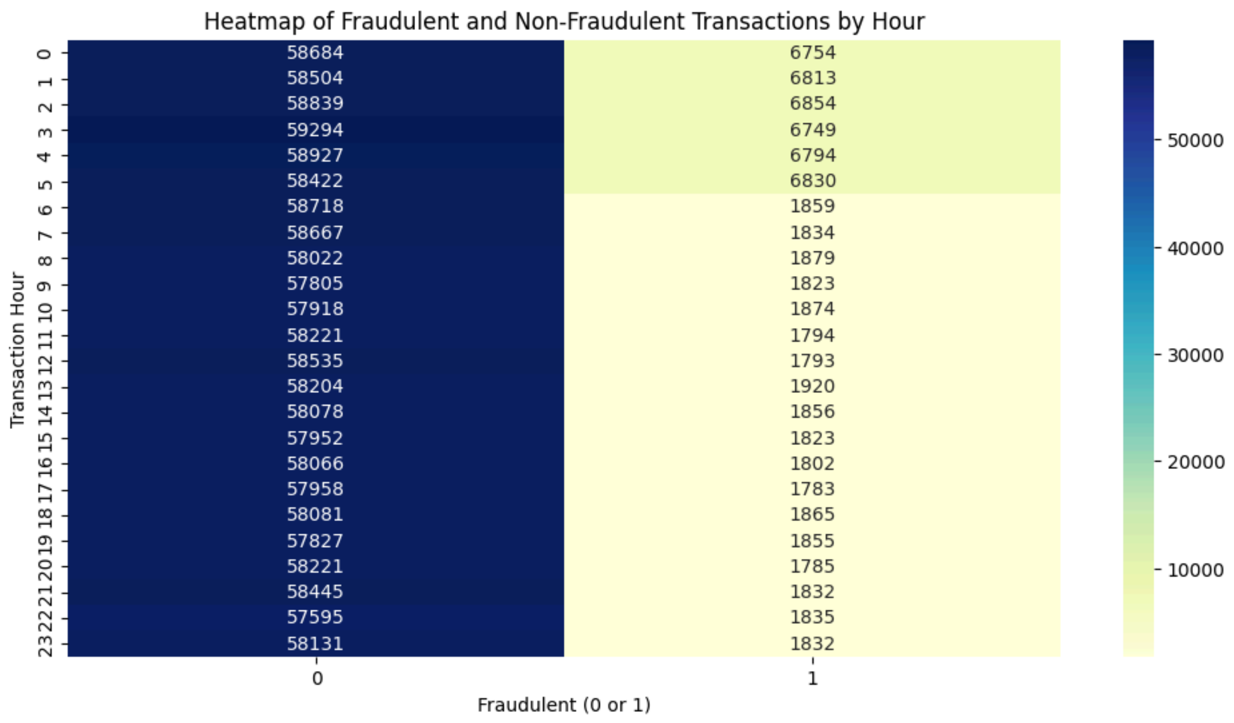
Fraudulent transaction are normally distributed across customers of all ages.

```
Start coding or generate with AI.
```

Hypothesis 3 : Sourabh Kodag (50606796) - The hypothesis "Fraudulent transactions vary by hour" assumes that time could be a factor for fraud. This hypothesis postulates that segments based on the time of day may be vulnerable to fraudulent activities. This analysis will help an organization understand patterns that could indicate the likelihood of fraud at specific times.

```
# Group by Transaction Hour and calculate the fraud rate
fraud_hour = df.groupby('Transaction Hour')['Is Fraudulent'].mean().reset_index()

# Plot 1: Line plot of Fraud Rate by Transaction Hour
plt.figure(figsize=(12, 6))
sns.lineplot(x='Transaction Hour', y='Is Fraudulent', data=fraud_hour, marker='o')
plt.title('Fraud Rate by Transaction Hour')
plt.xlabel('Transaction Hour')
plt.ylabel('Fraud Rate')
plt.xticks(range(0, 24))
plt.show()



# Create a pivot table to count fraudulent and non-fraudulent transactions by hour
hour_fraud_matrix = df.pivot_table(index='Transaction Hour',
                                   columns='Is Fraudulent',
                                   aggfunc='size',
                                   fill_value=0)

# Plot the heatmap
plt.figure(figsize=(12, 6))
sns.heatmap(hour_fraud_matrix, annot=True, cmap='YlGnBu', fmt='d')

plt.title('Heatmap of Fraudulent and Non-Fraudulent Transactions by Hour')
plt.xlabel('Fraudulent (0 or 1)')
plt.ylabel('Transaction Hour')
plt.show()
```



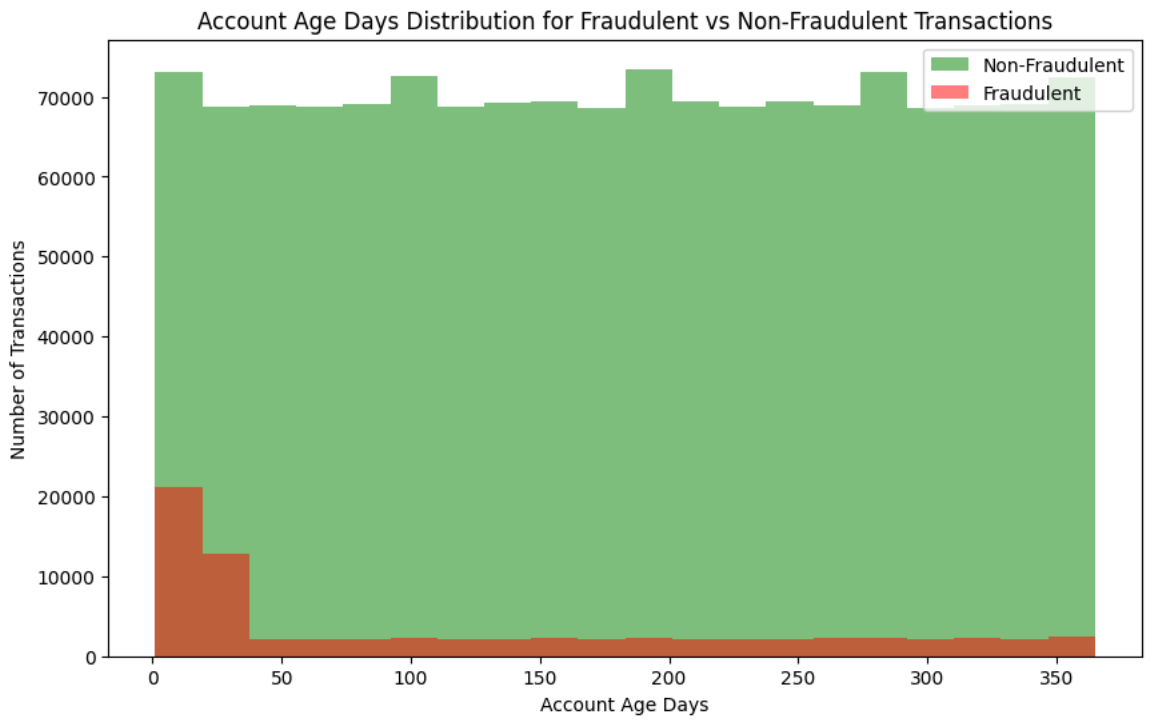


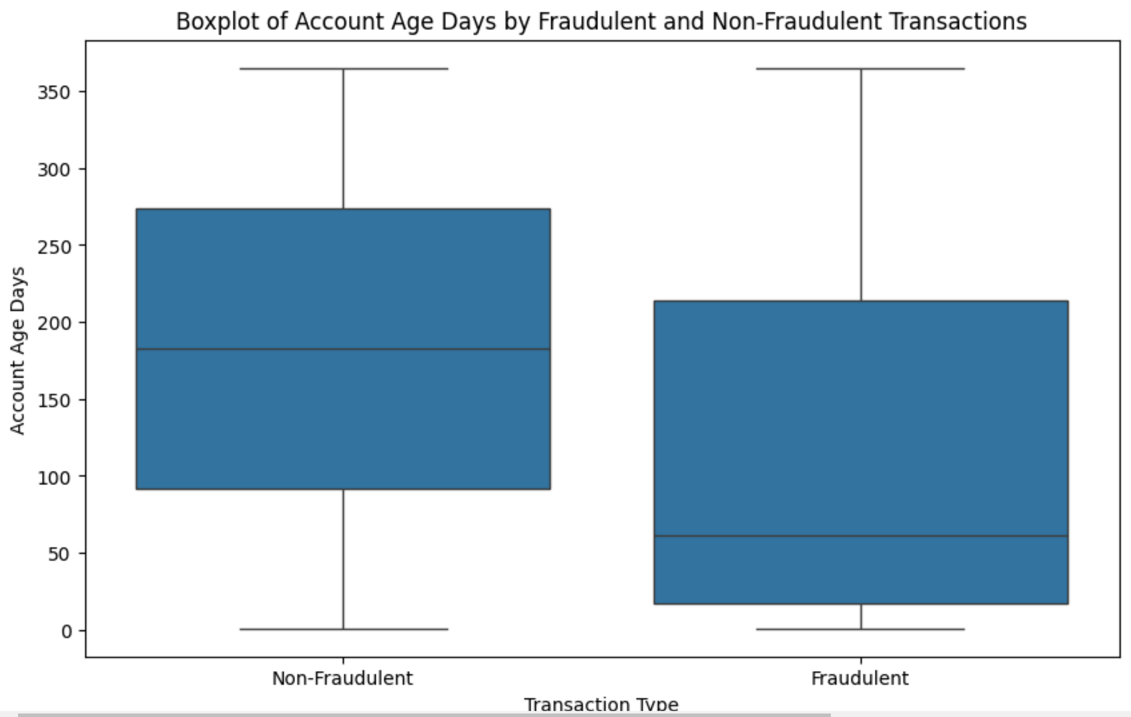
Hour	0	1
0	58684	6754
1	58504	6813
2	58839	6854
3	59294	6749
4	58927	6794
5	58422	6830
6	58718	1859
7	58667	1834
8	58022	1879
9	57805	1823
10	57918	1874
11	58221	1794
12	58535	1793
13	58204	1920
14	58078	1856
15	57952	1823
16	58066	1802
17	57958	1783
18	58081	1865
19	57827	1855
20	58221	1785
21	58445	1832
22	57595	1835
23	58131	1832

Hypothesis 4 : Sourabh Kodag (50606796) - This hypothesis therefore assumes that the newer the account, the more likely it is to be fraudulent compared to older, well-established accounts. A detailed explanation of this hypothesis and its importance is provided below.

```
df['Is Fraudulent'] = df['Is Fraudulent'].astype(bool)

fraudulent_transactions = df[df['Is Fraudulent'] == True]
non_fraudulent_transactions = df[df['Is Fraudulent'] == False]

print("Fraudulent Transactions Account Age Stats:")
print(fraudulent_transactions['Account Age Days'].describe())

print("\nNon-Fraudulent Transactions Account Age Stats:")
print(non_fraudulent_transactions['Account Age Days'].describe())
plt.figure(figsize=(10,6))


plt.hist(non_fraudulent_transactions['Account Age Days'], bins=20, alpha=0.5, label='Non-Fraudulent', color='green')
plt.hist(fraudulent_transactions['Account Age Days'], bins=20, alpha=0.5, label='Fraudulent', color='red')

plt.title('Account Age Days Distribution for Fraudulent vs Non-Fraudulent Transactions')
plt.xlabel('Account Age Days')
plt.ylabel('Number of Transactions')
plt.legend()

plt.show()


plt.figure(figsize=(10,6))
df['Fraudulent Label'] = df['Is Fraudulent'].apply(lambda x: 'Fraudulent' if x else 'Non-Fraudulent')

sns.boxplot(x='Fraudulent Label', y='Account Age Days', data=df)
plt.title('Boxplot of Account Age Days by Fraudulent and Non-Fraudulent Transactions')
plt.xlabel('Transaction Type')
plt.ylabel('Account Age Days')
plt.show()
```

```
Fraudulent Transactions Account Age Stats:
count    73838.000000
mean       116.295024
std        116.100774
min          1.000000
25%         17.000000
50%         61.000000
75%        214.000000
max        365.000000
Name: Account Age Days, dtype: float64

Non-Fraudulent Transactions Account Age Stats:
count    1.399114e+06
mean     1.829898e+02
std      1.053010e+02
min      1.000000e+00
25%      9.200000e+01
50%      1.830000e+02
75%      2.740000e+02
max      3.650000e+02
Name: Account Age Days, dtype: float64
```





Hypothesis 5 : Aditya Thakare (50608812) - "Is there a correlation between the payment method used and the likelihood of fraud?"

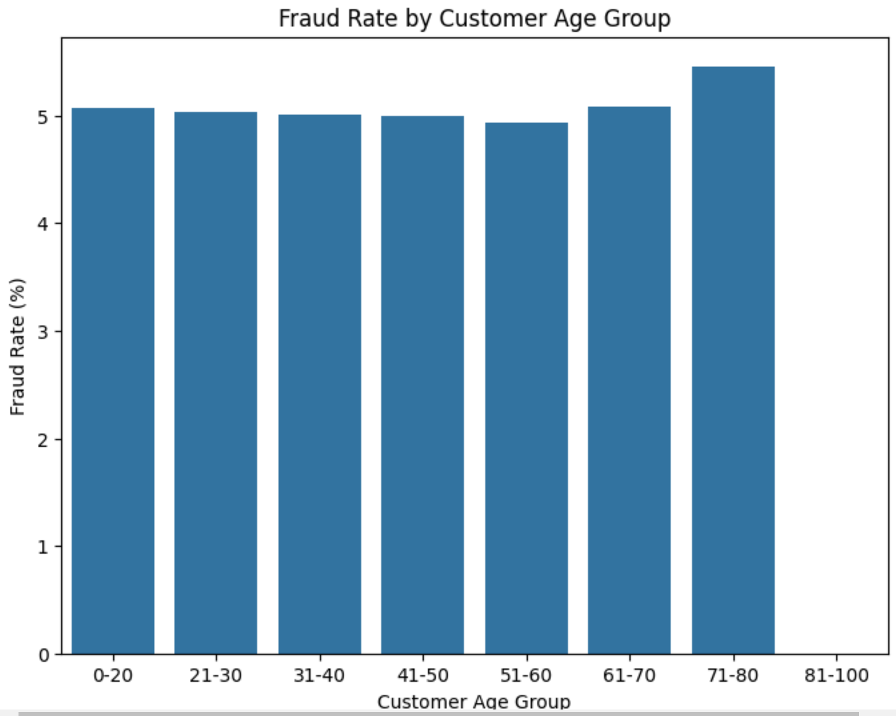
Why This Question is significant and leading to our object: Fraud Detection: Understanding the relationship between customer age and fraud can inform better risk assessment models. If fraudulent activities are detected among a population with younger age groups more frequently, then a business could institute additional verification steps for these transactions. Feature Engineering: This customer age can be a critical feature in fraud detection algorithms, especially by enabling the algorithm to create risk profiles. Market Strategies: Knowledge of the age-related pattern of fraud can help organizations in framing appropriate marketing strategies and fraud prevention policy.

Task 5. for Question1 (Aditya-50608812) Hypothesis 5: Older customers (above 60) are more likely to engage in fraudulent transactions.

EDA Operation 1: Fraud Rate by Age Group Objective: to find the fraud rate across different age groups and get the variation of fraud likelihood with customer age.

Steps: Divide into groups according to the age groups. Next, divide the data into age groups and calculate the rate of fraud in each group.

```
bins = [0, 20, 30, 40, 50, 60, 70, 80, 100]
labels = ['0-20', '21-30', '31-40', '41-50', '51-60', '61-70', '71-80', '81-100']
df['Age_Group'] = pd.cut(df['Customer Age'], bins=bins, labels=labels, right=False)

age_group_fraud_rate = df.groupby('Age_Group')['Is Fraudulent'].mean() * 100

plt.figure(figsize=(8, 6))
sns.barplot(x=age_group_fraud_rate.index, y=age_group_fraud_rate.values)
plt.title('Fraud Rate by Customer Age Group')
plt.ylabel('Fraud Rate (%)')
plt.xlabel('Customer Age Group')
plt.show()
```



The bar chart displays the fraud rates across different age groups. A higher fraud rate in the older age group (>60) supports the hypothesis that older customers are more likely to engage in fraud.

Task 5. Question 1(Aditya-50608812) Hypothesis 6: Working younger customers (e.g., between 25-45 years old) are more likely to engage in fraudulent transactions.

EDA operation 2: Transaction Amount Distribution per Age Group: transaction amount for different age categories for fraudulent transactions.

Steps: Filter the dataset for fraudulent transactions. Create a Boxplot to Compare Transaction Amount across the defined age groupings:

```
fraudulent_data_1 = df[df['Is Fraudulent'] == 1]

plt.figure(figsize=(8, 6))
sns.boxplot(x='Age_Group', y='Transaction Amount', data=fraudulent_data_1)
plt.title('Transaction Amount Distribution for Fraudulent Transactions by Customer Age Group')
plt.ylabel('Transaction Amount')
plt.xlabel('Customer Age Group')
plt.show()
```