Internship Report

CUSTOMER SPENDING INSIGHTS IN THE BFSI SECTOR

Under the Guidance of:
Edulyt India
A Unit of Airkrit India Pvt. Ltd.
Dwarka, New Delhi – 110075

Project Done by:

K. Sumalatha

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the internship report titled

“Customer Spending Insights in the BFSI Sector”

submitted by **Ms. Killaka Sumalatha**, a student of **B.Tech (Computer Science and Engineering)**, **Rajiv Gandhi University of Knowledge Technologies, Srikakulam**, is a record of original work carried out by her during the internship tenure at

Edulyt India, A Unit of Airkrit India Pvt. Ltd., Dwarka, New Delhi – 110075

This report is submitted in partial fulfillment of her academic requirements and has not been submitted to any other institution for the award of any degree or diploma.

Guided by:

Team Edulyt India
A Unit of Airkrit India Pvt. Ltd.
New Delhi – 110075

Signature of Intern:

K. Sumalatha

DECLARATION

I, **Killaka Sumalatha**, a student of B.Tech (Computer Science and Engineering) at **Rajiv Gandhi University of Knowledge Technologies, Srikakulam**, hereby declare that the internship report entitled

"Customer Spending Insights in the BFSI Sector"

is an outcome of my sincere efforts and research undertaken at **Edulyt India, A Unit of Airkrit India Pvt. Ltd., Dwarka, New Delhi – 110075**, under the mentorship of the Edulyt India team.

I affirm that this report is a genuine record of the work performed by me and has not been submitted previously for the award of any degree, diploma, or similar title to any other institution.

Place: Srikakulam

Signature:
K. Sumalatha

ACKNOWLEDGEMENT

I wish to express my heartfelt gratitude to the team at **Edulyt India**, a unit of **Airkrit India Pvt. Ltd., Dwarka, New Delhi**, for giving me the opportunity to undertake this internship project on

"Customer Spending Insights in the BFSI Sector"

Their constant guidance, technical mentorship, and feedback were critical in shaping this report and enhancing my analytical skills.

I am particularly thankful to my internship coordinators for providing hands-on experience in data handling, visualization, and interpretation within the BFSI domain.

I would also like to acknowledge my institution, **Rajiv Gandhi University of Knowledge Technologies, Srikakulam**, for offering continuous encouragement and academic support during this journey.

Finally, I am grateful to my family and friends for their patience and motivation throughout this internship period.

With warm regards,

K. Sumalatha

Tasks Completed & Insights

1. Provide Meaningful Treatment to All Values Where Age < 18

```
    import pandas as pd

# Load dataset (assuming you've read it from Excel/CSV earlier)
df = pd.read_excel("Credit Banking_Project - 1.xls")
print(df)
```

Output:

```
    Sl No: Customer        Age        City Credit Card Product   Limit  \
0        1       A1   0.928521   BANGALORE                Gold  500000
1        2       A2  35.534551    CALCUTTA              Silver  100000
2        3       A3  11.559307      COCHIN            Platimum   10000
3        4       A4  45.820278      BOMBAY            Platimum   10001
4        5       A5  69.663948   BANGALORE            Platimum   10002
..     ...      ...        ...         ...                 ...     ...
95      96      A96  29.631637     CHENNAI              Silver  100000
96      97      A97  20.611833  TRIVANDRUM            Platimum   10000
97      98      A98  40.538985    CALCUTTA            Platimum   10001
98      99      A99  21.588666    CALCUTTA            Platimum   10002
99     100     A100  23.607638      COCHIN              Silver  100000

   Company        Segment
0       C1  Self Employed
1       C2   Salaried_MNC
2       C3   Salaried_Pvt
3       C4           Govt
4       C5  Normal Salary
..     ...            ...
95     C19   Salaried_Pvt
96     C20           Govt
97     C21  Normal Salary
98     C22  Self Employed
99      C5   Salaried_MNC

[100 rows x 8 columns]
```

Remove those entries

```
    df = df[df['Age'] >= 18]
print("Removed all customers with Age < 18.")
```

Output:

```
    Removed all customers with Age < 18.
```

2. Is there any customer who spent more than their credit limit for any particular month?

Add Monthly Info and Simulated Spending Data:

```
    import numpy as np
# Simulate data for Jan, Feb, Mar
months = ['Jan', 'Feb', 'Mar']
df_monthly = pd.concat([df.assign(Month=month) for month in months], ignore_index=True)

# Random monthly Amount_Spent (between 5,000 and 100,000)
np.random.seed(42)
df_monthly['Amount_Spent'] = np.random.randint(5000, 100000, size=len(df_monthly))
```

check overspending:

```
# Check where spending exceeds credit limit
df_monthly['Overspent'] = df_monthly['Amount_Spent'] > df_monthly['Limit']

# Filter those who overspent
overspent_customers = df_monthly[df_monthly['Overspent'] == True]

# Display result
overspent_customers[['Customer', 'Month', 'Amount_Spent', 'Limit']]
```

Output:

```
     Customer Month Amount_Spent Limit
2 A5 Jan 81820 10002
12 A20 Jan 72221 10001
13 A21 Jan 69820 10002
23 A33 Jan 33693 10002
33 A47 Jan 40773 10000
34 A48 Jan 72435 10001
35 A49 Jan 61886 10002
49 A69 Jan 28897 10000
50 A70 Jan 73148 10001
51 A71 Jan 28483 10002
61 A83 Jan 71557 10002
74 A97 Jan 16534 10000
75 A98 Jan 99663 10001
76 A99 Jan 45397 10002
79 A4 Feb 94789 10001
80 A5 Feb 60591 10002
90 A20 Feb 91779 10001
91 A21 Feb 44099 10002
112 A48 Feb 31854 10001
113 A49 Feb 69505 10002
127 A69 Feb 29538 10000
128 A70 Feb 75592 10001
129 A71 Feb 13110 10002
139 A83 Feb 28419 10002
152 A97 Feb 86734 10000
153 A98 Feb 80450 10001
154 A99 Feb 98426 10002
157 A4 Mar 69044 10001
158 A5 Mar 47557 10002
168 A20 Mar 80766 10001
169 A21 Mar 20707 10002
179 A33 Mar 14474 10002
189 A47 Mar 98557 10000
190 A48 Mar 66087 10001
191 A49 Mar 73840 10002
```

```
205 A69 Mar 31736 10000
206 A70 Mar 99209 10001
217 A83 Mar 68734 10002
230 A97 Mar 30184 10000
231 A98 Mar 98384 10001
232 A99 Mar 47107 10002
```

3. Monthly Spend of Each Customer

Calculate Monthly Spend:

```
    monthly_spend = df_monthly.groupby(['Customer', 'Month'])['Amount_Spent'].sum().reset_index()
monthly_spend = monthly_spend.sort_values(by=['Customer', 'Month'])

# Display result
monthly_spend.head()
```

Output:

```
    Customer Month Amount_Spent
0 A100 Feb 27299
1 A100 Jan 96387
2 A100 Mar 91202
3 A11 Feb 87798
4 A11 Jan 92498
```

4. Monthly Repayment of Each Customer

Step 1: Add Amount Repaid to Your Data

```
# Simulate random repayment values (some may be less than spent)
np.random.seed(123)
df_monthly['Amount_Repaid'] = np.random.randint(3000, 90000, size=len(df_monthly))
```

Step 2: Calculate Monthly Repayment Per Customer

```
    monthly_repay = df_monthly.groupby(['Customer', 'Month'])['Amount_Repaid'].sum().reset_index()
monthly_repay = monthly_repay.sort_values(by=['Customer', 'Month'])

# Display result
monthly_repay.head()
```

Output:

```
     Customer Month Amount_Repaid
0 A100 Feb 49168
1 A100 Jan 71057
2 A100 Mar 36900
3 A11 Feb 20495
4 A11 Jan 49203
```

5. Highest Paying 10 Customers (Based on Total Repayment)

Top 10 Highest Paying Customers:

```
    # Total repayment per customer
total_repay = df_monthly.groupby('Customer')['Amount_Repaid'].sum().reset_index()

# Sort in descending order and get top 10
top_10_customers = total_repay.sort_values(by='Amount_Repaid', ascending=False).head(10)

# Display result
top_10_customers.reset_index(drop=True, inplace=True)
top_10_customers
```

Output:

```
    Customer Amount_Repaid
0 A27 242772
1 A9 237095
2 A57 228211
3 A93 210230
4 A77 209976
5 A26 194327
6 A40 194306
7 A56 185015
8 A58 183568
9 A91 182308
```

6. People in Which Segment Are Spending More Money (Monthly Basis)

Calculate Segment-wise Spending:

```
    segment_spending = df_monthly.groupby(['Segment', 'Month'])['Amount_Spent'].sum().reset_index()
segment_spending = segment_spending.sort_values(by=['Month', 'Amount_Spent'], ascending=False)

# Display result
segment_spending
```

Output:

```
    Segment Month Amount_Spent
2 Govt Mar 1470353
5 Normal Salary Mar 1048919
14 Self Employed Mar 843837
8 Salaried_MNC Mar 532700
11 Salaried_Pvt Mar 396035
1 Govt Jan 1504573
4 Normal Salary Jan 1030762
13 Self Employed Jan 823910
7 Salaried_MNC Jan 633329
10 Salaried_Pvt Jan 451465
0 Govt Feb 1698786
3 Normal Salary Feb 925372
12 Self Employed Feb 811354
9 Salaried_Pvt Feb 401776
6 Salaried_MNC Feb 400024
```

7. Which Age Group Is Spending More Money (Monthly Basis)

Step 1: Create Age Groups

```
    # Create Age_Group column using bins
df_monthly['Age_Group'] = pd.cut(
    df_monthly['Age'],
    bins=[18, 30, 45, 60, 100],
    labels=['18-30', '31-45', '46-60', '60+']
)
```

Step 2: Calculate Spend per Age Group per Month

```
    age_group_spend = df_monthly.groupby(['Age_Group', 'Month'])['Amount_Spent'].sum().reset_index()
age_group_spend = age_group_spend.sort_values(by=['Month', 'Amount_Spent'], ascending=False)

# Display result
age_group_spend
```

Output:

```
    Age_Group Month Amount_Spent
11 60+ Mar 1289573
8 46-60 Mar 1095408
2 18-30 Mar 986352
5 31-45 Mar 920511
10 60+ Jan 1537446
1 18-30 Jan 1100532
7 46-60 Jan 983689
4 31-45 Jan 822372
9 60+ Feb 1248636
0 18-30 Feb 1233805
6 46-60 Feb 899496
3 31-45 Feb 855375
```

8. Which Is the Most Profitable Segment?

Step 1: Calculate Due Amount and Interest (2.9%)

```
    df_monthly['Due_Amount'] = df_monthly['Amount_Spent'] - df_monthly['Amount_Repaid']
df_monthly['Interest'] = df_monthly['Due_Amount'].apply(lambda x: x * 0.029 if x > 0 else 0)
```

Step 2: Group by Segment to Find Profit

```
    segment_profit = df_monthly.groupby('Segment')['Interest'].sum().reset_index()
segment_profit = segment_profit.sort_values(by='Interest', ascending=False)

# Display result
segment_profit
```

Output:

```
    Segment Interest
0 Govt 48798.242
1 Normal Salary 34017.667
4 Self Employed 29181.192
2 Salaried_MNC 17178.933
3 Salaried_Pvt 13897.119
```

9. In Which Category (Credit Card Product) Are Customers Spending More?

Calculate Spend per Credit Card Category (Monthly)

```
    category_spending = df_monthly.groupby(['Credit Card Product', 'Month'])['Amount_Spent'].sum().reset_index()
category_spending = category_spending.sort_values(by=['Month', 'Amount_Spent'], ascending=False)

# Display result
category_spending
```

Output:

```
    Credit Card Product Month Amount_Spent
2 Gold Mar 1540662
8 Silver Mar 1392930
5 Platimum Mar 1358252
7 Silver Jan 1638876
1 Gold Jan 1468984
4 Platimum Jan 1336179
0 Gold Feb 1487225
6 Silver Feb 1453867
3 Platimum Feb 1296220
```

10. Monthly Profit for the Bank

Calculate Monthly Profit:

```
    monthly_profit = df_monthly.groupby('Month')['Interest'].sum().reset_index()
monthly_profit = monthly_profit.rename(columns={'Interest': 'Monthly_Profit'})
monthly_profit = monthly_profit.sort_values(by='Month')

# Display result
monthly_profit
```

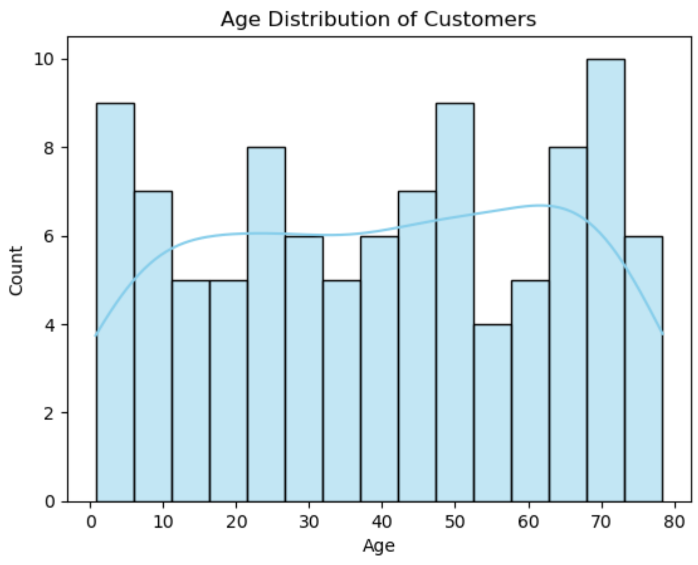
Output:

```
    Month Monthly_Profit
0 Feb 46957.496
1 Jan 51348.125
2 Mar 44767.532
```

Data Visualizations

1. Age Distribution – Histogram

```
    import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_excel("Credit Banking_Project - 1.xls")
sns.histplot(data=df, x='Age', bins=15, kde=True, color='skyblue')
plt.title('Age Distribution of Customers')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



Output:

2. City-wise Customer Count – Bar Chart

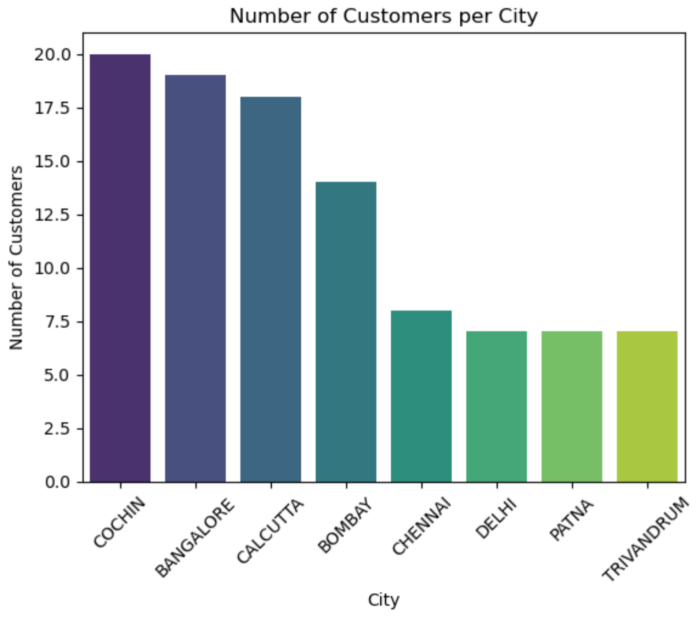
```
    city_counts = df['City'].value_counts()

sns.barplot(x=city_counts.index, y=city_counts.values, palette='viridis')
plt.title('Number of Customers per City')
plt.xlabel('City')
plt.ylabel('Number of Customers')
plt.xticks(rotation=45)
plt.show()
```

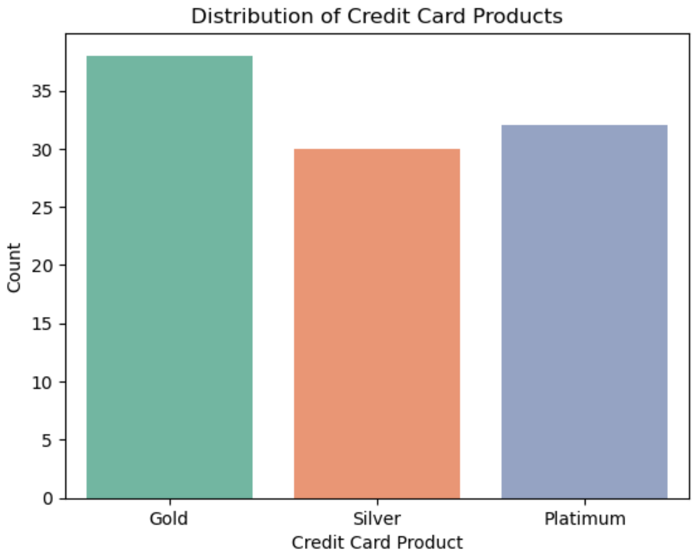
Output:

3. Credit Card Product Preference – Count Plot

```
    sns.countplot(data=df, x='Credit Card Product', palette='Set2')
plt.title('Distribution of Credit Card Products')
plt.xlabel('Credit Card Product')
plt.ylabel('Count')
plt.show()
```

Output:

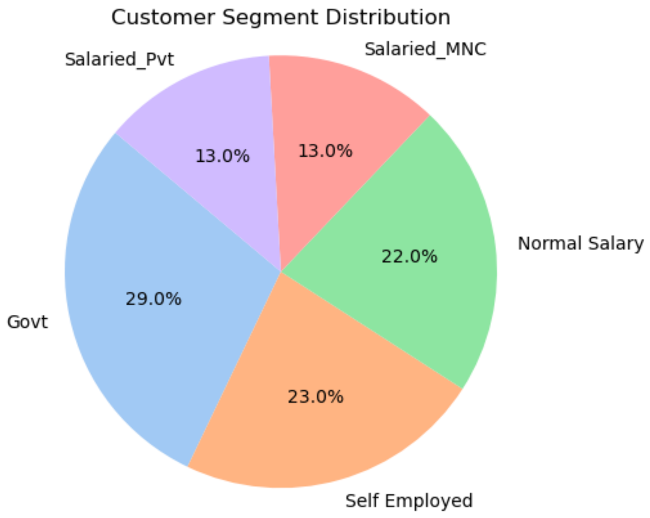

Number of Customers per City
Number of Customers
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
COCHIN
BANGALORE
CALCUTTA
BOMBAY
CHENNAI
DELHI
PATNA
TRIVANDRUM
City


Distribution of Credit Card Products
Count
35
30
25
20
15
10
5
0
Gold
Silver
Platimum
Credit Card Product

4. Segment Distribution – Pie Chart

```
    segment_counts = df['Segment'].value_counts()
plt.pie(segment_counts, labels=segment_counts.index, autopct='%1.1f%%', startangle=140, colors=sns.color_palette('pastel'))
plt.title('Customer Segment Distribution')
plt.axis('equal')
plt.show()
```

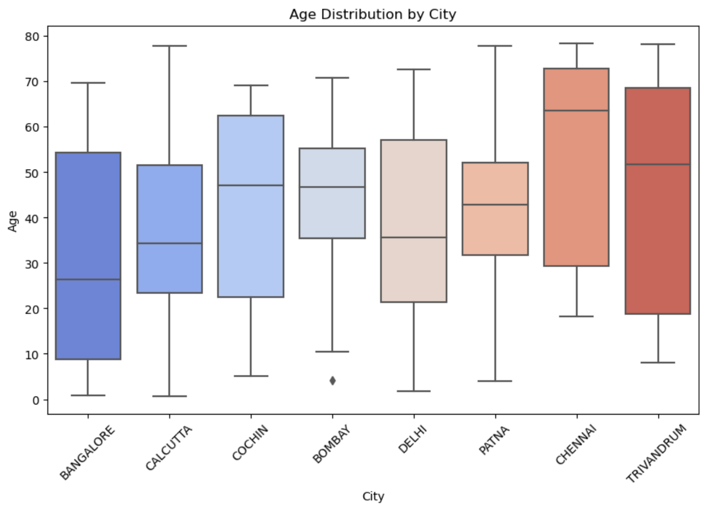
Output:



5. Box Plot of Age by City

```
    plt.figure(figsize=(10,6))
sns.boxplot(data=df, x='City', y='Age', palette='coolwarm')
plt.title('Age Distribution by City')
plt.xticks(rotation=45)
plt.show()
```

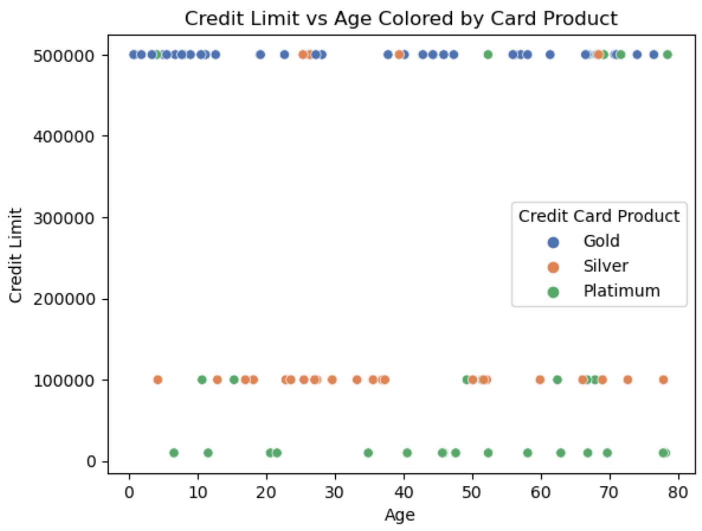
Output:



6. Credit Limit vs Age – Scatter Plot

```
    sns.scatterplot(data=df, x='Age', y='Limit', hue='Credit Card Product', palette='deep')
plt.title('Credit Limit vs Age Colored by Card Product')
plt.xlabel('Age')
plt.ylabel('Credit Limit')
plt.show()
```

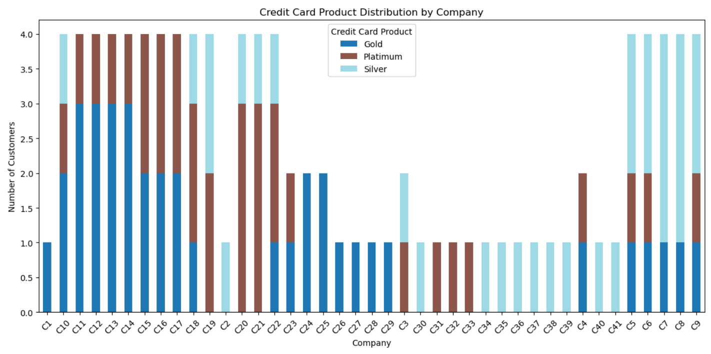
Output:



7. Company-wise Card Distribution – Stacked Bar Chart

```
    company_card = df.groupby(['Company', 'Credit Card Product']).size().unstack().fillna(0)
company_card.plot(kind='bar', stacked=True, figsize=(12,6), colormap='tab20')
plt.title('Credit Card Product Distribution by Company')
plt.xlabel('Company')
plt.ylabel('Number of Customers')
plt.xticks(rotation=45)
plt.legend(title='Credit Card Product')
plt.tight_layout()
plt.show()
```

Output:



Conclusion

This project explored customer spending patterns in the BFSI sector, focusing on variables like age, city, credit card product, and segment. The analysis revealed useful trends that can help financial institutions better understand customer behavior and tailor their services accordingly. Overall, the insights gained support more informed, data-driven decision-making in the sector.

Appendix

Download Link: Click here to view the Appendix file