Dataset - Exploratory and Descriptive Analysis

In this notebook, we perform an extensive exploratory and descriptive analysis of a credit card financial dataset, with the objective of uncovering behavioral and demographic patterns that influence credit usage, delinquency, and customer satisfaction.

This analytical phase is critical for understanding the underlying structure of the data, validating data quality, and generating insights that inform downstream decision-making and modeling strategies. Through a combination of descriptive statistics and interactive visualizations, we analyze customer profiles, credit card usage behaviors, financial metrics, and satisfaction levels.

The analysis covers key topics such as customer distribution by marital status, credit card activation trends, average interest earned across card types, delinquency by state, and customer breakdowns by `job`, `gender`, and `satisfaction score`. Each visualization is tailored to enhance interpretability and support business or operational decision-making.

We begin by importing the necessary Python libraries:

- `pandas`: for data manipulation, transformation, and tabular exploration.
- `numpy`: for numerical operations and efficient array handling.
- `os`: to manage file paths and export analysis outputs.
- `plotly.express`: for building clean, publication-ready interactive charts.

warnings: to suppress unnecessary runtime warnings for cleaner outputs.

This notebook lays the foundation for deeper statistical modeling and dashboard reporting by providing a clear and structured view of the data's characteristics and trends.

```
# Import libraries
import os
import pandas as pd
import numpy as np
import plotly.express as px
```

Define and Create Directory Paths

To ensure reproducibility and organized storage, we programmatically create directories if they don't already exist for:

- **raw data**
- **processed data**
- **results**
- **documentation**

These directories will store intermediate and final outputs for reproducibility.

```python
#Get working directory
current_dir = os.getcwd()
#go one directory up to root directory
project_root_dir = os.path.dirname(current_dir)
#Define path to data files
data_dir = os.path.join(project_root_dir, 'data')
raw_dir = os.path.join(data_dir, 'raw')
processed_dir = os.path.join(data_dir, 'processed')
#Define path to results folder
results_dir = os.path.join(project_root_dir, 'results')
#Define path to results folder
docs_dir = os.path.join(project_root_dir, 'docs')

#Create directories if they do not exist
os.makedirs(raw_dir, exist_ok=True)
os.makedirs(processed_dir, exist_ok=True)
os.makedirs(results_dir, exist_ok=True)
os.makedirs(docs_dir, exist_ok=True)
```

Loading the Cleaned Dataset

We load the cleaned version of the Credit Card Financial Dataset from the data/processed/ directory into a Pandas DataFrame. This dataset contains customer-level information including demographic attributes, financial activity, credit usage behavior, and satisfaction metrics.

The first ten records are displayed using the `head(5)` function to provide a preview of key columns such as `Client_Num`, `Card_Category`, `Annual_Fees`, `Credit_Limit`, `Total_Trans_Amt`, `Cust_Satisfaction_Score`.... This initial view helps confirm successful loading and gives a quick look at the structure and content of the cleaned dataset.

```python
merged_data_filename = os.path.join(processed_dir, "Credit_Card_Financial.csv")
merged_df = pd.read_csv(merged_data_filename)
merged_df.head(5)
```

	Client_Num	Card_Category	Annual_Fees	Activation_30_Days	Customer_Acq_Cost	Week_Sta...
0	708082083	blue	200	0	87	2023-01-01
1	708083283	blue	445	1	108	2023-01-01
2	708084558	blue	140	0	106	2023-01-01
3	708085458	blue	250	1	150	2023-01-01
4	708086958	blue	320	1	106	2023-01-01

```python
merged_df.columns
```

```
Index(['Client_Num', 'Card_Category', 'Annual_Fees', 'Activation_30_Days',
       'Customer_Acq_Cost', 'Week_Start_Date', 'Week_Num', 'Qtr',
       'current_year', 'Credit_Limit', 'Total_Revolving_Bal',
       'Total_Trans_Amt', 'Total_Trans_Vol', 'Avg_Utilization_Ratio',
       'Use Chip', 'Exp Type', 'Interest_Earned', 'Delinquent_Acc',
       'Customer_Age', 'Gender', 'Dependent_Count', 'Education_Level',
       'Marital_Status', 'state_cd', 'Car_Owner', 'House_Owner',
       'Personal_loan', 'Customer_Job', 'Income', 'Cust_Satisfaction_Score',
       'Month'],
      dtype='object')
```

```python
merged_df.shape
```

```
(10108, 31)
```

```python
merged_df.info
```

```
<bound method DataFrame.info of        Client_Num Card_Category  Annual_Fees  Activation_30_
0        708082083          blue          200                 0
1        708083283          blue          445                 1
2        708084558          blue          140                 0
3        708085458          blue          250                 1
4        708086958          blue          320                 1
...            ...           ...          ...               ...
10103    827695683          blue          340                 1
```

```
10104    827703258          blue          395                    1
10105    827712108          blue          125                    1
10106    827888433          blue          410                    0
10107    827890758          blue          100                    0

       Customer_Acq_Cost Week_Start_Date Week_Num Qtr  current_year  \
0                     87      2023-01-01   week-1  q1          2023
1                    108      2023-01-01   week-1  q1          2023
2                    106      2023-01-01   week-1  q1          2023
3                    150      2023-01-01   week-1  q1          2023
4                    106      2023-01-01   week-1  q1          2023
...                  ...             ...      ...  ..           ...
10103                106      2023-12-24  week-52  q4          2023
10104                104      2023-12-24  week-52  q4          2023
10105                107      2023-12-24  week-52  q4          2023
10106                 96      2023-12-24  week-52  q4          2023
10107                 43      2023-12-24  week-52  q4          2023

       Credit_Limit  ... Education_Level  Marital_Status   state_cd  \
0            3544.0  ...      uneducated          single    Florida
1            3421.0  ...         unknown         married  New Jersey
2            8258.0  ...         unknown         married  New Jersey
3            1438.3  ...      uneducated          single   New York
4            3128.0  ...        graduate          single      Texas
...             ...  ...             ...             ...        ...
10103       34516.0  ...        graduate          single     Nevada
10104       13426.0  ...         unknown         unknown      Texas
10105        2346.0  ...        graduate         married   New York
10106        6648.0  ...        graduate         married      Texas
10107        2062.0  ...     high school         married   New York

       Car_Owner House_Owner Personal_loan   Customer_Job  Income  \
0             no         yes            no     businessman  202326
1             no          no            no    selfemployeed    5225
2            yes          no            no    selfemployeed   14235
3             no          no            no     blue-collar   45683
4            yes         yes            no     businessman   59279
...          ...         ...           ...             ...     ...
10103        yes         yes           yes    selfemployeed   44903
10104         no         yes            no    selfemployeed   73655
10105        yes          no            no    selfemployeed   62287
10106         no          no            no    white-collar   21390
10107         no         yes            no    selfemployeed   42187
```

```
     Cust_Satisfaction_Score      Month
0                            3    January
1                            2    January
2                            2    January
3                            1    January
4                            1    January
...                        ...       ...
10103                        5   December
10104                        3   December
10105                        3   December
10106                        4   December
10107                        4   December

[10108 rows x 31 columns]>
```

Summary Statistics: Numerical Variables

```
merged_df.describe()
```

	Client__Num	Annual__Fees	Activation__30__Days	Customer__Acq__Cost	current__year	Credit_
count	1.010800e+04	10108.000000	10108.000000	10108.000000	10108.0	10108.0
mean	7.390104e+08	291.849525	0.574693	96.254056	2023.0	8635.64
std	3.673623e+07	118.339384	0.494414	25.768677	0.0	9093.13
min	7.080821e+08	95.000000	0.000000	40.000000	2023.0	1438.30
25%	7.130267e+08	195.000000	0.000000	79.000000	2023.0	2552.75
50%	7.179037e+08	295.000000	1.000000	95.000000	2023.0	4549.00
75%	7.727989e+08	395.000000	1.000000	112.000000	2023.0	11070.2
max	8.278908e+08	500.000000	1.000000	172.000000	2023.0	34516.0

This summary provides a snapshot of key distribution characteristics.

We see that annual fees range from $95 to $500, with a mean of $291.85 and a median of $295. The distribution appears approximately symmetrical, centered around common fee brackets, suggesting a standardized pricing structure across products. The upper range could reflect premium services or high-tier customers.

The activation within 30 days is a binary variable, and the mean of 0.57 indicates that about 57% of customers activated their accounts promptly. This majority suggests either strong onboarding or incentives driving early engagement.

Customer acquisition costs range from $40 to $172, with an average of $96.25. While the median is close to the mean at $95, the standard deviation of $25.77 suggests moderate variation in marketing or sales strategies. The higher end may reflect targeted campaigns for premium customer segments.

All records come from the year 2023, ensuring temporal consistency and simplifying trend comparisons.

The credit limit distribution is notably right-skewed. Limits range from $1,438 to $34,516, with a mean of $8,635 and a median of $4,549. This substantial gap implies that while most customers have modest limits, a small segment enjoys significantly higher lines of credit, potentially due to higher incomes or credit scores.

Total revolving balances and utilization ratios also exhibit right-skewness. The average revolving balance is $1,162.79, and the average utilization is 27.5%, though a portion of customers reach full utilization (max = 99.9%). This pattern is typical in credit datasets, where most users maintain moderate usage, but some hover near or at the limit, signaling financial stress or high spending behavior.

Total transaction amounts average $4,404.63, with a wide spread (up to $18,484), indicating variability in spending patterns. Transaction volumes range from 10 to 139, with a median of 67, aligning with moderate monthly use and consistent card engagement.

Interest earned also reveals financial diversity. The average is $775.96, but values go up to $4,785, implying some customers are carrying balances over time, while others pay off promptly and avoid interest.

The delinquency rate is low, with only about 6% of customers having a delinquent account. This suggests relatively healthy repayment behavior in the majority of the sample.

Customers report an average of 2.35 dependents, ranging up to 5, with the most common values between 1 and 3. This distribution supports a demographic base consisting of family households.

Income is perhaps the most skewed feature. It spans from $1,250 to $239,791, with a mean of $56,976 and median near $44,768. This implies income inequality in the sample, with a small number of high earners pulling up the average. The majority earn below $76K, with a significant concentration in the lower brackets.

Finally, customer satisfaction scores range from 1 (low) to 5 (high), with an average of 3.19. This moderate central tendency suggests generally neutral-to-positive feedback, but with room for improvement. The distribution's standard deviation of 1.26 shows variation in experience across customer segments.

Summary Statistics: Categorical Variables

```
merged_df.describe(include='object')
```

	Card_Category	Week_Start_Date	Week_Num	Qtr	Use Chip	Exp Type	Customer_Age
count	10108	10108	10108	10108	10108	10108	10108
unique	4	52	52	4	3	6	6
top	blue	2023-07-02	week-27	q2	swipe	bills	46-55
freq	9214	195	195	2535	7101	2970	4104

```
merged_df['Gender'].value_counts(normalize=True)
```

```
Gender
Female    0.581717
Male      0.418283
Name: proportion, dtype: float64
```

```
merged_df['Card_Category'].value_counts(normalize=True)
```

```
Card_Category
blue        0.911555
silver      0.063217
gold        0.018599
platinum    0.006628
Name: proportion, dtype: float64
```

```
merged_df['Marital_Status'].value_counts(normalize=True)
```

```
Marital_Status
married    0.507321
single     0.419074
unknown    0.073605
Name: proportion, dtype: float64
```

```
merged_df['Education_Level'].value_counts(normalize=True)
```

```
Education_Level
graduate        0.408983
high school     0.198753
unknown         0.149881
uneducated      0.146715
post-graduate   0.051049
doctorate       0.044618
Name: proportion, dtype: float64
```

```
merged_df['Cust_Satisfaction_Score'].value_counts(normalize=True)
```

```
Cust_Satisfaction_Score
3    0.303522
4    0.207657
5    0.195489
2    0.177285
1    0.116047
Name: proportion, dtype: float64
```

The dataset shows that 58.17% of the customers are female, while 41.83% are male. This indicates a higher representation of female credit card holders in the data. Such a distribution could suggest that women are either more likely to use the credit card services offered by this institution or are better represented in the customer base. Understanding this gender balance is important for designing personalized financial products, marketing strategies, and improving customer satisfaction

The majority of customers—91.16%—hold a Blue card, making it the most common card category by far. Silver cards account for 6.32%, while Gold and Platinum cards represent just 1.86% and 0.66% respectively. This distribution suggests that most customers are enrolled in entry-level or standard credit card programs. Premium cards like Gold and Platinum are significantly less common, likely due to stricter eligibility criteria or targeted offerings for high-income or high-credit-score individuals. This insight can help institutions reassess product penetration and evaluate the success of their premium card promotions.

The data shows that 50.73% of the customers are married, while 41.91% are single. A smaller portion, 7.36%, have their marital status listed as unknown. This suggests that over half of the customer base is in committed relationships, which could influence financial behaviors such as joint spending, credit sharing, or long-term financial planning. The relatively high percentage of single individuals also indicates a significant market segment for independent financial products. The presence of unknown entries may point to missing data or customers opting not to disclose personal details.

The largest portion of customers—40.90%—are graduates, followed by 19.88% with a high school education. Notably, 14.99% of the data falls under unknown, and 14.67% of customers are uneducated. Higher education levels, such as post-graduate and doctorate, account for 5.10% and 4.46% respectively. This indicates that the majority of the customer base has at least a college education, which could correlate with more stable income levels and credit behavior. However, the sizable unknown and uneducated segments suggest the need for inclusive financial services and possible improvement in data collection practices.

The most common satisfaction score is 3, making up 30.35% of the customers, followed by scores of 4 (20.77%) and 5 (19.55%). Lower satisfaction levels are less frequent, with 17.73% of customers rating 2, and only 11.60% giving the lowest score of 1. This distribution shows that most customers are moderately satisfied, but there is a nearly even split between higher (4–5) and lower (1–2) satisfaction scores. The presence of significant dissatisfaction (nearly 30%) highlights opportunities for improving customer experience, while the strong presence of mid-to-high scores shows potential for customer retention if services are optimized.

Insights

```
average_income = merged_df["Income"].mean()
print("Average Income:", round(average_income, 2))
```

```
Average Income: 56976.1
```

```
total_customers = merged_df["Client_Num"].nunique()
print("Total Customers:", total_customers)
```

```
Total Customers: 10108
```

```
avg_credit_limit = merged_df["Credit_Limit"].mean()
print("Average Credit Limit:", round(avg_credit_limit, 2))
```

```
Average Credit Limit: 8635.64
```

```
avg_score = merged_df["Cust_Satisfaction_Score"].mean()
print("Average Satisfaction Score:", round(avg_score, 2))
```

```
Average Satisfaction Score: 3.19
```

1. Average Income: 56,976.10 This indicates that on average, customers in the dataset earn around $56,976 annually. This relatively moderate income level suggests a mostly middle-income customer base. Financial institutions can use this figure to tailor credit products and services that align with average earning capacity.

2. Total Customers: 10,108 The dataset contains information on 10,108 customers, offering a substantial sample size for analysis. This ensures a diverse representation of demographics, occupations, and behaviors, making any derived patterns or trends more reliable and insightful for business decisions.

3. Average Credit Limit: 8,635.64 On average, each customer has access to about $8,635 in credit. This reflects the institution's credit allocation strategy and risk tolerance. Comparing this with income, the average credit limit is approximately 15% of the average annual income, suggesting a conservative credit extension policy.

4. Average Satisfaction Score: 3.19 (out of 5) The satisfaction score indicates a moderate level of customer satisfaction, slightly above neutral. A score of 3.19 implies that while many customers are relatively content, there's still room for improvement in service delivery, credit products, or customer support.

Gender **Distribution**

```
gender_dist = merged_df['Gender'].value_counts().reset_index()
gender_dist.columns = ['Gender', 'count']

fig = px.pie(gender_dist, names='Gender', values='count', title="Gender Distribution",
             color_discrete_sequence=["#003399 ", "#3366cc ", "#dbe4f0 "])

fig.update_layout(
    template = "presentation",
    legend_title = dict(text="Gender"),
    paper_bgcolor="rgba(0, 0, 0, 0)",
    plot_bgcolor="rgba(0, 0, 0, 0)"
)

fig.update_traces(texttemplate="%{percent:.2%}", textposition="inside")

fig.show()

fig.write_image(os.path.join(results_dir, 'Gender_Distribution_Pie_Chart.jpg'))
fig.write_image(os.path.join(results_dir, 'Gender_Distribution_Pie_Chart.png'))
fig.write_html(os.path.join(results_dir, 'Gender_Distribution_Pie_Chart.html'))
```

Gender Distribution

This pie chart visualizes the proportion of male and female customers in the dataset. Females make up 58.17% of the sample, while males account for 41.83%. The slight overrepresentation of women could indicate gender-based trends in credit card usage, spending habits, or customer satisfaction. Financial institutions might use this insight to tailor marketing strategies or credit offerings to different demographic groups.

Income by `Education Level`

```
merged_df_income_edlevel = merged_df.groupby('Education_Level')['Income'].mean().reset_index
merged_df_income_edlevel
```

	Education_Level	average_income
5	unknown	58040.557096
2	high school	57771.664510
4	uneducated	57364.834794
3	post-graduate	56640.678295
1	graduate	56254.135462
0	doctorate	55579.789357

```
total_income = merged_df_income_edlevel['average_income'].sum()
merged_df_income_edlevel['percentage'] = (merged_df_income_edlevel['average_income'] / total_
merged_df_income_edlevel = merged_df_income_edlevel.sort_values(by='Education_Level', ascend:
merged_df_income_edlevel
```

	Education_Level	average_income	percentage
5	unknown	58040.557096	16.988226
4	uneducated	57364.834794	16.790445
3	post-graduate	56640.678295	16.578488
2	high school	57771.664510	16.909523
1	graduate	56254.135462	16.465348
0	doctorate	55579.789357	16.267970

```python
fig = px.bar(
    merged_df_income_edlevel,
    x='Education_Level',
    y='percentage',
    title='Average Income Distribution by Education Level (%)',
    barmode='group',
    height=600,
    width=1000,
    color_discrete_sequence=['#002366'],
    text='percentage'
)

fig.update_layout(
    template="presentation",
    xaxis_title="Education Level",
    yaxis_title="Percentage of Total Average Income",
    legend_title_text=None,
    paper_bgcolor = "rgba(0, 0, 0, 0)",
    plot_bgcolor = "rgba(0, 0, 0, 0)",
    xaxis=dict(showgrid=False),
    yaxis=dict(showgrid=False)
)

fig.update_traces(
    texttemplate="%{text:.2f}%",
    textposition="outside",
    marker_line_width=0
)

fig.write_image(os.path.join(results_dir, 'Avg_Income_by_EdLevel.jpg'))
fig.write_image(os.path.join(results_dir, 'Avg_Income_by_EdLevel.png'))
fig.write_html(os.path.join(results_dir, 'Avg_Income_by_EdLevel.html'))
```

```
fig.show()
```

Average Income Distribution by Education Level (%)
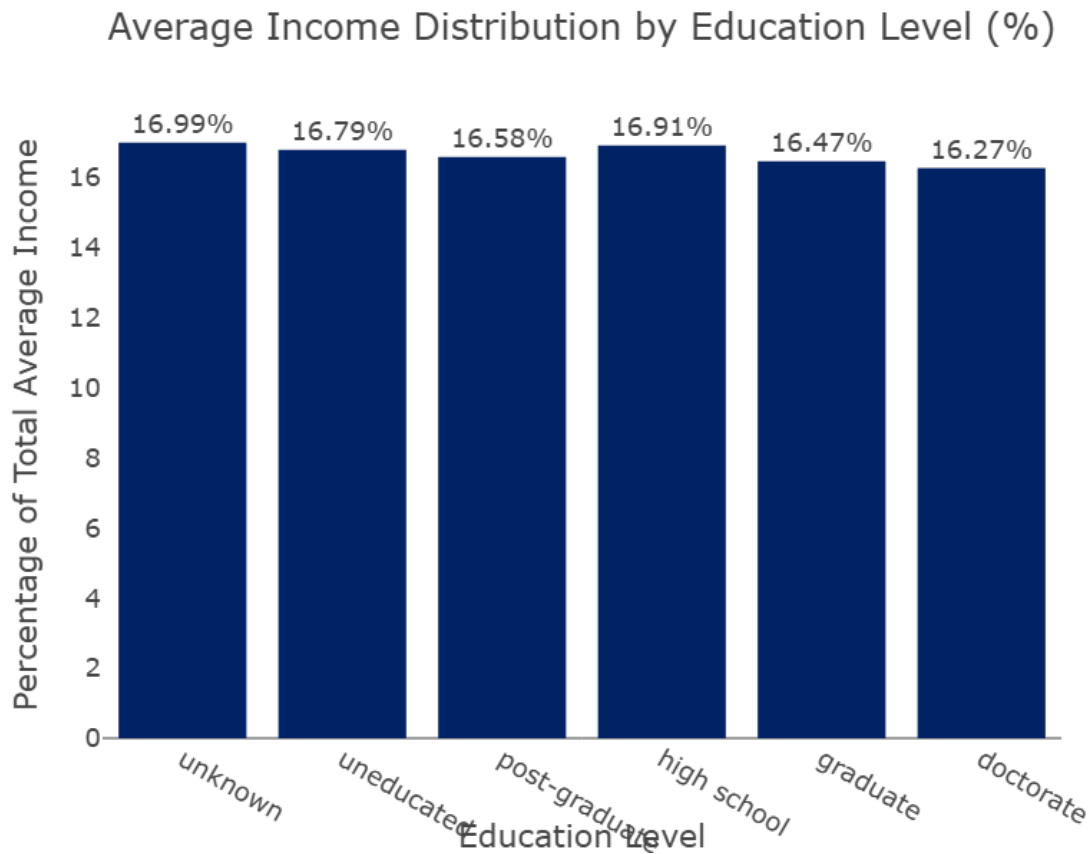
This bar chart compares average income across different education levels. Surprisingly, individuals with an "Unknown" education level report the highest average income (16.99%), followed by " Uneducated" graduates (16.79%). Meanwhile, those with "Doctorate" degrees have the lowest average income (16.27%). This suggests that formal education does not necessarily correlate with higher income in this dataset, possibly due to other factors like occupation type or regional economic conditions.

Customers by Marital Status

```
marital_status_df = merged_df['Marital_Status'].value_counts().reset_index()
marital_status_df.columns = ['Marital_Status', 'total_customers']
marital_status_df = marital_status_df.sort_values(by='total_customers', ascending=False)
marital_status_df
```

	Marital_Status	total_customers
0	married	5128
1	single	4236
2	unknown	744

```python
total = marital_status_df['total_customers'].sum()
marital_status_df['percentage'] = (marital_status_df['total_customers'] / total) * 100
marital_status_df
```

	Marital_Status	total_customers	percentage
0	married	5128	50.732093
1	single	4236	41.907400
2	unknown	744	7.360507

```python
fig = px.bar(
    marital_status_df,
    y='Marital_Status',
    x='percentage',
    orientation='h',
    title='Customer Distribution by Marital Status (%)',
    color_discrete_sequence=['#002366'],
    text = 'percentage',
    height=600,
    width=900
)

fig.update_layout(
    template = "presentation",
    xaxis_title="Percentage of Customers",
    yaxis_title="Marital Status",
    legend_title_text=None,
    paper_bgcolor="rgba(0, 0, 0, 0)",
    plot_bgcolor="rgba(0, 0, 0, 0)",
    xaxis=dict(showgrid=False),
    yaxis=dict(showgrid=False)
)

fig.update_traces(
    texttemplate="%{text:.2f}%",
```
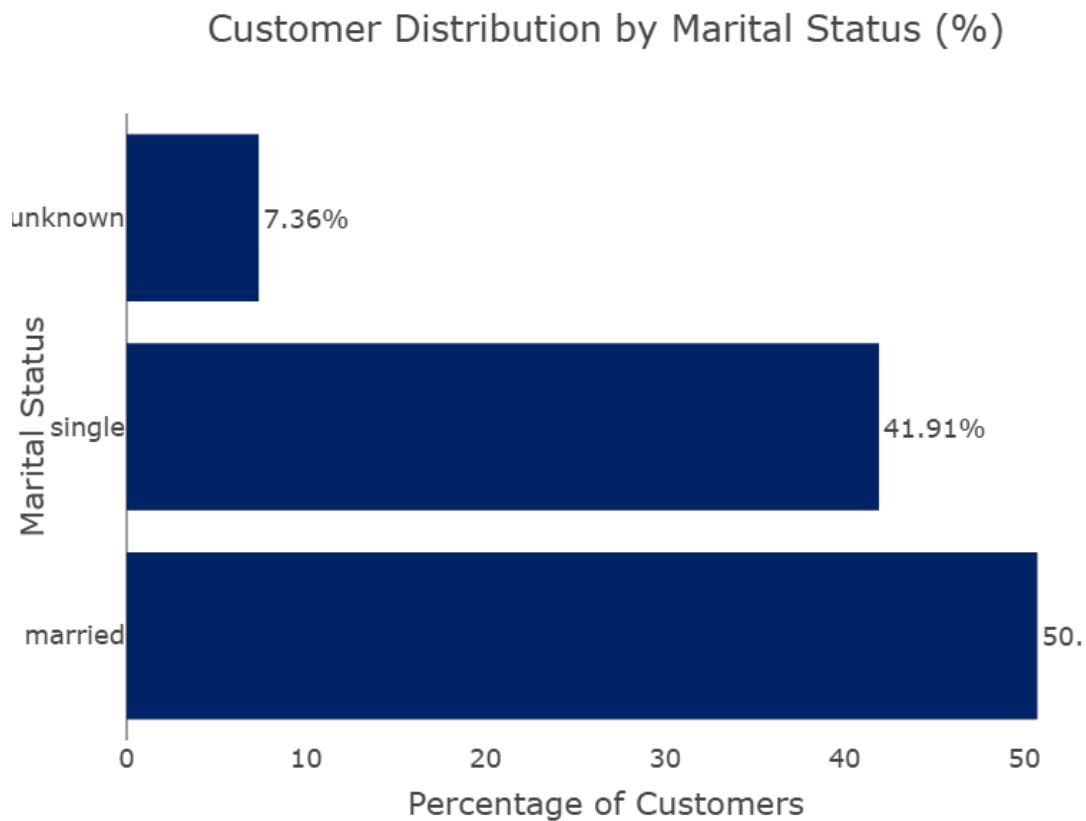
```
        textposition="outside",
        marker_line_width=0
)

fig.write_image(os.path.join(results_dir, 'Customers_by_Marital_Status.jpg'))
fig.write_image(os.path.join(results_dir, 'Customers_by_Marital_Status.png'))
fig.write_html(os.path.join(results_dir, 'Customers_by_Marital_Status.html'))

fig.show()
```


The dataset shows that 50.73% of customers are married, 41.91% are single, and 7.36% have an unknown marital status. Since married individuals dominate, banks could explore whether marital status influences spending behavior, credit utilization, or repayment patterns. For example, married couples might have higher combined credit limits or different financial priorities.

Average Credit Limit **by** Card Type

```
avg_credit_by_card = merged_df.groupby('Card_Category')['Credit_Limit'].mean().reset_index(na
avg_credit_by_card = avg_credit_by_card.sort_values(by='average_credit_limit', ascending=Fals
avg_credit_by_card
```

	Card_Category	average_credit_limit
3	silver	23391.641628
1	gold	21857.835106
2	platinum	16455.134328
0	blue	7285.657858

```
total_credit = avg_credit_by_card['average_credit_limit'].sum()
avg_credit_by_card['percentage'] = (avg_credit_by_card['average_credit_limit'] / total_credit
avg_credit_by_card
```

	Card_Category	average_credit_limit	percentage
3	silver	23391.641628	33.905712
1	gold	21857.835106	31.682490
2	platinum	16455.134328	23.851385
0	blue	7285.657858	10.560414

```
Purpose: Shows the distribution of average credit limits across different card types.

Key Insights:

Platinum cards have the highest average credit limit (33.9%), followed by gold (31.68%) and s

Blue cards have the lowest limit (10.56%).

Implication: Premium cards (platinum/gold) offer higher credit limits, likely targeting high-

fig = px.bar(
    avg_credit_by_card,
    x='Card_Category',
    y='percentage',
    title='Average Credit Limit by Card Type (%)',
    color_discrete_sequence=['#002366'],
```

```
        text='percentage',
        height=600,
        width=1000
)

fig.update_layout(
        template="presentation",
        xaxis_title="Card Type",
        yaxis_title="Percentage of Total Avg Credit Limit",
        legend_title_text=None,
        paper_bgcolor="rgba(0, 0, 0, 0)",
        plot_bgcolor="rgba(0, 0, 0, 0)",
        xaxis=dict(showgrid=False),
        yaxis=dict(showgrid=False)
)

fig.update_traces(
        texttemplate="%{text:.2f}%",
        textposition="outside",
        marker_line_width=0
)

fig.write_image(os.path.join(results_dir, 'Avg_Credit_Limit_by_Card_Type.jpg'))
fig.write_image(os.path.join(results_dir, 'Avg_Credit_Limit_by_Card_Type.png'))
fig.write_html(os.path.join(results_dir, 'Avg_Credit_Limit_by_Card_Type.html'))

fig.show()
```
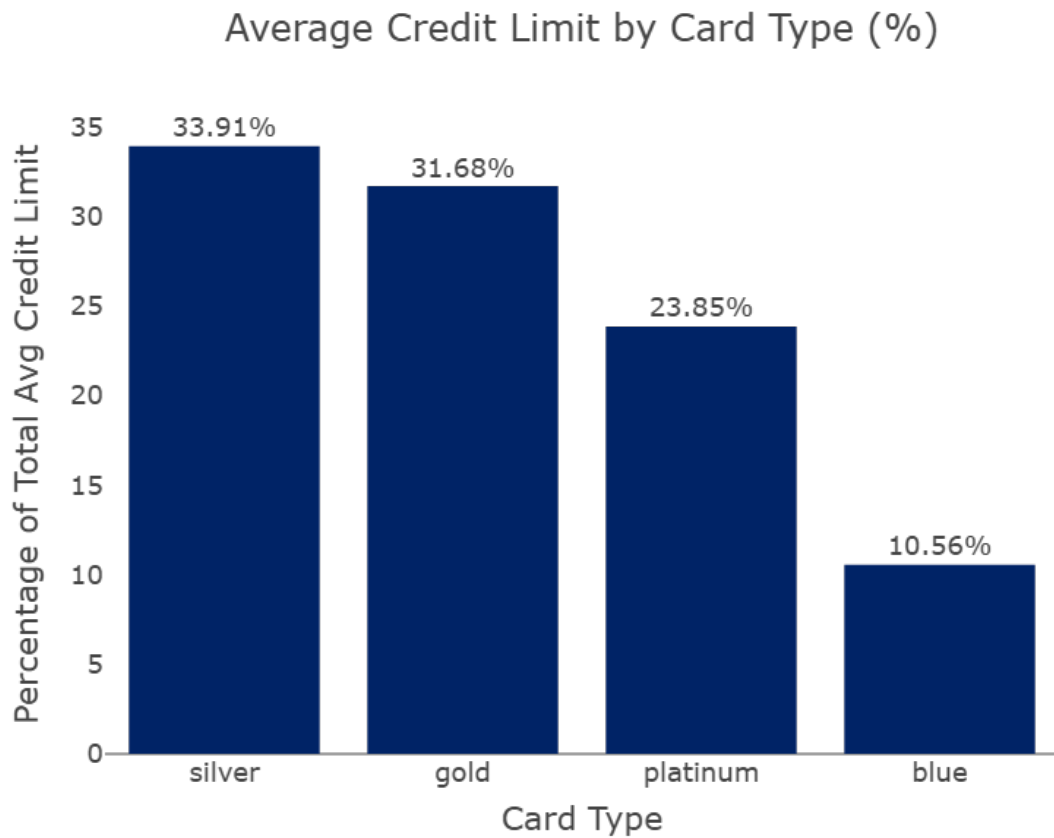
Average Credit Limit by Card Type (%)

This bar chart illustrates how the average credit limit differs across various card types (e.g., Silver, Gold, Platinum). It's useful for evaluating which card types offer more credit and for what customer profiles. KPlatinum cards have the highest average credit limit (33.9%), followed by gold (31.68%) and silver (23.85%). Blue cards have the lowest limit (10.56%). Premium cards (platinum/gold) offer higher credit limits, likely targeting high-income customers.

Average Interest Earned per Card Type

```
avg_interest_by_card = merged_df.groupby('Card_Category')['Interest_Earned'].mean().reset_ind
avg_interest_by_card = avg_interest_by_card.sort_values(by='average_interest', ascending=Fals
avg_interest_by_card
```

	Card_Category	average_interest
2	platinum	2412.373881

	Card_Category	average_interest
1	gold	1988.213617
3	silver	1270.862723
0	blue	705.001925

```python
total_interest = avg_interest_by_card['average_interest'].sum()
avg_interest_by_card['percentage'] = (avg_interest_by_card['average_interest'] / total_intere
avg_interest_by_card
```

	Card_Category	average_interest	percentage
2	platinum	2412.373881	37.832541
1	gold	1988.213617	31.180562
3	silver	1270.862723	19.930562
0	blue	705.001925	11.056335

```python
fig = px.bar(
    avg_interest_by_card,
    x='Card_Category',
    y='percentage',
    title='Average Interest Earned per Card Type (%)',
    color_discrete_sequence=['#002366'],
    text='percentage',
    height=600,
    width=1000
)

fig.update_layout(
    template="presentation",
    xaxis_title="Card Type",
    yaxis_title="Percentage of Total Avg Interest Earned",
    legend_title_text=None,
    paper_bgcolor="rgba(0, 0, 0, 0)",
    plot_bgcolor="rgba(0, 0, 0, 0)",
    xaxis=dict(showgrid=False),
    yaxis=dict(showgrid=False)
)

fig.update_traces(
    texttemplate="%{text:.2f}%",
```
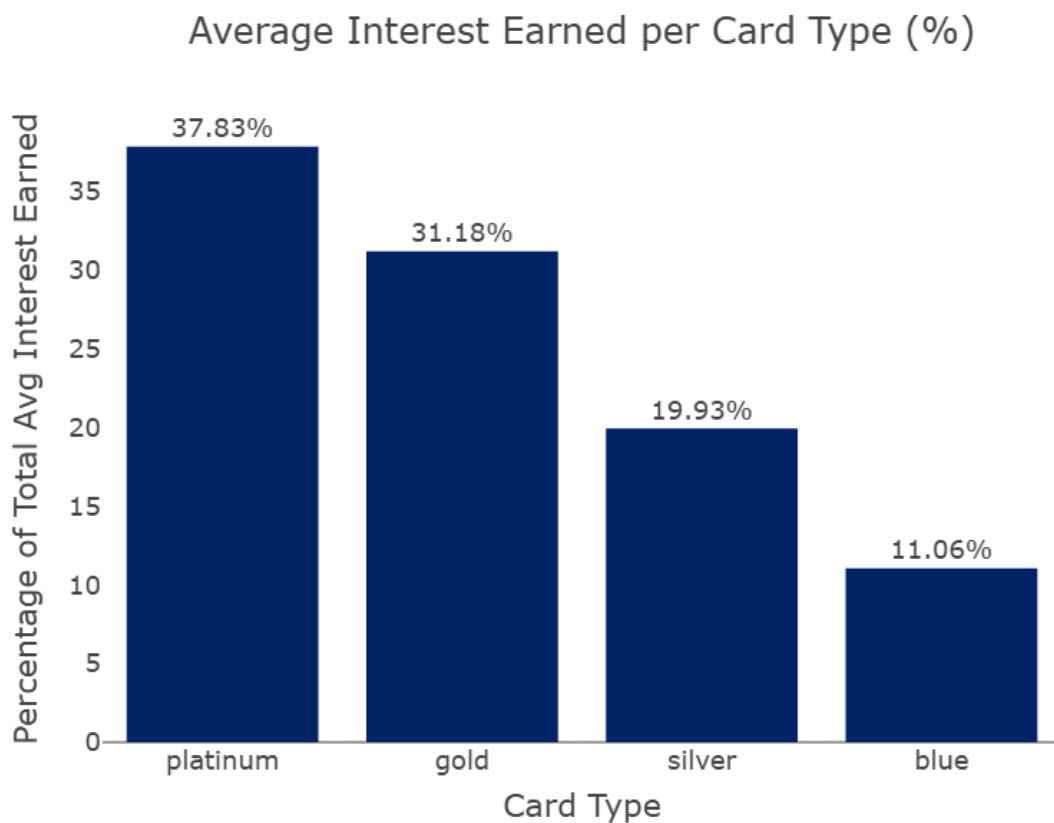
```
    textposition="outside",
    marker_line_width=0
)

fig.write_image(os.path.join(results_dir, 'Avg_Interest_by_Card_Type.jpg'))
fig.write_image(os.path.join(results_dir, 'Avg_Interest_by_Card_Type.png'))
fig.write_html(os.path.join(results_dir, 'Avg_Interest_by_Card_Type.html'))

fig.show()
```

Average Interest Earned per Card Type (%)

This chart compares the average interest earned from each card type. It helps in assessing which card types are more profitable for the issuer based on customer behavior. Platinum cards generate the most interest (31.18%), followed by gold (19.93%). Silver and blue cards contribute less (exact percentages unclear due to missing labels).

Implication: Higher credit limits (platinum/gold) may lead to more borrowing and interest income for the issuer.

Usage Mode vs Total Spend

```
usage_vs_spend = merged_df.groupby('Use Chip')['Total_Trans_Amt'].sum().reset_index(name='tot
usage_vs_spend = usage_vs_spend.sort_values(by='total_spend', ascending=True)
usage_vs_spend
```

	Use Chip	total_spend
1	online	2776610
0	chip	13850621
2	swipe	27894782

```
total_spend_sum = usage_vs_spend['total_spend'].sum()
usage_vs_spend['percentage'] = (usage_vs_spend['total_spend'] / total_spend_sum) * 100
usage_vs_spend
```

	Use Chip	total_spend	percentage
1	online	2776610	6.236488
0	chip	13850621	31.109602
2	swipe	27894782	62.653910

```
fig = px.bar(
    usage_vs_spend,
    x='percentage',
    y='Use Chip',
    orientation='h',
    title='Usage Mode vs Total Spend (%)',
    color_discrete_sequence=['#002366'],
    text='percentage',
    height=600,
    width=1000
)

fig.update_layout(
    template="presentation",
    xaxis_title="Percentage of Total Spend",
    yaxis_title="Usage Mode",
    legend_title_text=None,
    paper_bgcolor="rgba(0, 0, 0, 0)",
```

```
    plot_bgcolor="rgba(0, 0, 0, 0)",
    xaxis=dict(showgrid=False),
    yaxis=dict(showgrid=False)
)

fig.update_traces(
    texttemplate="%{text:.2f}%",
    textposition="outside",
    marker_line_width=0
)

fig.write_image(os.path.join(results_dir, 'Usage_Mode_vs_Total_Spend.jpg'))
fig.write_image(os.path.join(results_dir, 'Usage_Mode_vs_Total_Spend.png'))
fig.write_html(os.path.join(results_dir, 'Usage_Mode_vs_Total_Spend.html'))

fig.show()
```


Usage Mode vs Total Spend (%)
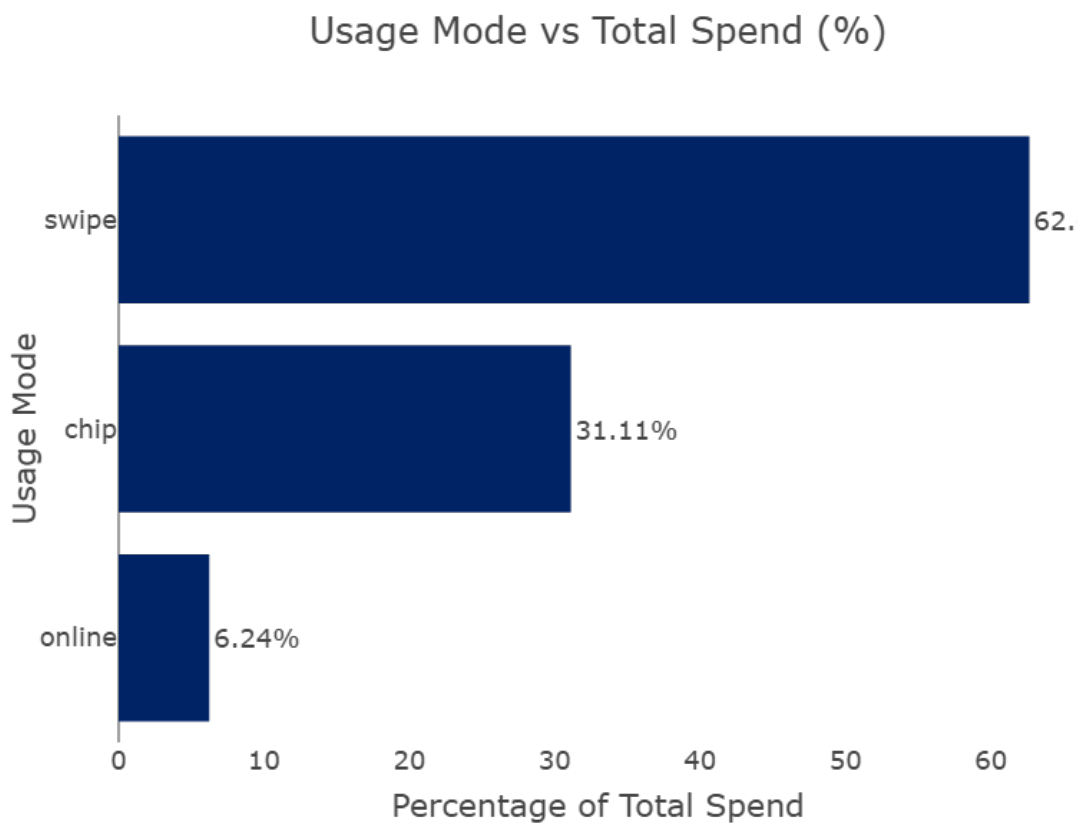
his graph shows how spending is distributed across usage modes (e.g., swipe, online, tap). It highlights customer preferences and transaction habits across platforms. Swipe dominates (62%), followed by chip (31.11%) and online (6.24%).

Implication: Customers prefer in-person transactions (swipe/chip) over online payments.

Total Transaction Amount Over Time

```
month_order = ['January', 'February', 'March', 'April', 'May', 'June',
               'July', 'August', 'September', 'October', 'November', 'December']
merged_df['Month'] = pd.Categorical(merged_df['Month'], categories=month_order, ordered=True)
```

```
monthly_trans_amt = merged_df.groupby('Month')['Total_Trans_Amt'].sum().reset_index(name='to
monthly_trans_amt = monthly_trans_amt.sort_values('Month')
monthly_trans_amt
```

C:\Users\user\AppData\Local\Temp\ipykernel_18356\1145090672.py:1: FutureWarning:

The default of observed=False is deprecated and will be changed to True in a future version

	Month	total_transaction_amount
0	January	4322186
1	February	3539575
2	March	3388827
3	April	4174728
4	May	3426913
5	June	3533660
6	July	4546958
7	August	3449868
8	September	3452874
9	October	4050909
10	November	3405420
11	December	3230095

```
fig = px.line(
    monthly_trans_amt,
    x='Month',
    y='total_transaction_amount',
```

```
    title='Total Transaction Amount Over Time (Monthly)',
    markers=True
)

# Apply smoothing using a spline
fig.update_traces(line_shape='spline', line=dict(color='#002366', width=3))

fig.update_layout(
    template="presentation",
    xaxis_title="Month",
    yaxis_title="Total Transaction Amount",
    paper_bgcolor="rgba(0,0,0,0)",
    plot_bgcolor="rgba(0,0,0,0)",
    xaxis=dict(showgrid=False),
    yaxis=dict(showgrid=False)
)

fig.write_image(os.path.join(results_dir, 'Total_Transaction_Amount_Over_Time.jpg'))
fig.write_image(os.path.join(results_dir, 'Total_Transaction_Amount_Over_Time.png'))
fig.write_html(os.path.join(results_dir, 'Total_Transaction_Amount_Over_Time.html'))

fig.show()
```
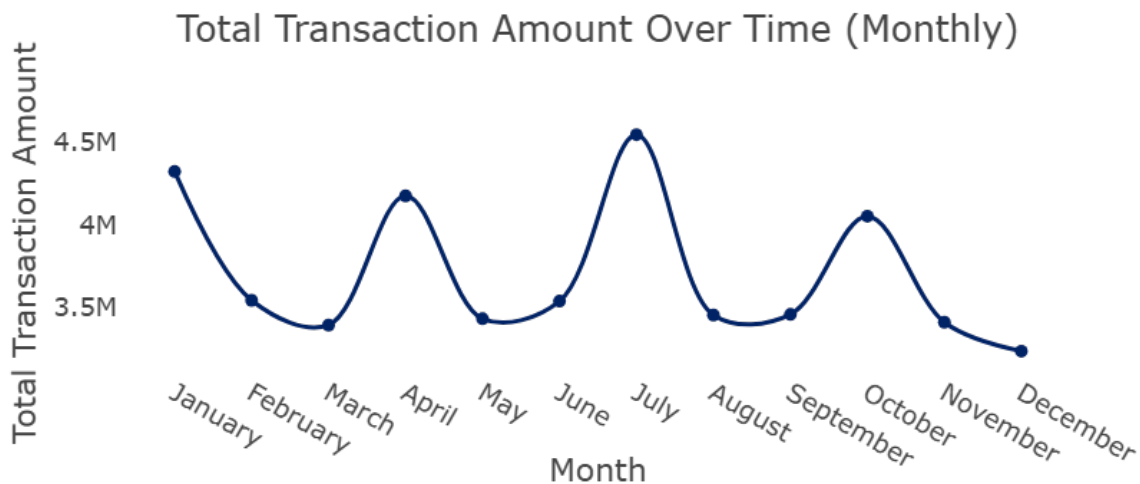

This line chart tracks the monthly trend of total transaction amounts. It reveals seasonal patterns, spikes, or drops in spending that might relate to holidays or economic changes. Peaks in December (~2.2M) and March (~3.38M), with dips in April and July.

Implication: Seasonal spikes (e.g., holidays, tax season) drive higher spending.

Spending by Expense Type

```
exp_type_spending = merged_df.groupby('Exp Type')['Client_Num'].nunique().reset_index(name='t
exp_type_spending = exp_type_spending.sort_values(by='total_customers', ascending=False)
exp_type_spending
```

	Exp Type	total_customers
0	bills	2970
1	entertainment	1988
3	fuel	1759
4	grocery	1502
2	food	1187
5	travel	702

```
total_customers = exp_type_spending['total_customers'].sum()
exp_type_spending['percentage'] = (exp_type_spending['total_customers'] / total_customers) *
exp_type_spending
```

	Exp Type	total_customers	percentage
0	bills	2970	29.382667
1	entertainment	1988	19.667590
3	fuel	1759	17.402058
4	grocery	1502	14.859517
2	food	1187	11.743174
5	travel	702	6.944994

```
fig = px.bar(
    exp_type_spending,
    x='Exp Type',
    y='percentage',
    title='Spending by Expense Type (%)',
    text='percentage',
    color_discrete_sequence=['#002366'],
    height=600,
    width=1000
)

fig.update_layout(
```

```python
    template="presentation",
    xaxis_title="Expense Type",
    yaxis_title="Percentage of Customers",
    paper_bgcolor="rgba(0,0,0,0)",
    plot_bgcolor="rgba(0,0,0,0)",
    xaxis=dict(showgrid=False),
    yaxis=dict(showgrid=False)
)

fig.update_traces(
    texttemplate="%{text:.2f}%",
    textposition="outside",
    marker_line_width=0
)

fig.write_image(os.path.join(results_dir, 'Spending_by_Expense_Type.jpg'))
fig.write_image(os.path.join(results_dir, 'Spending_by_Expense_Type.png'))
fig.write_html(os.path.join(results_dir, 'Spending_by_Expense_Type.html'))

fig.show()
```

Spending by Expense Type (%)
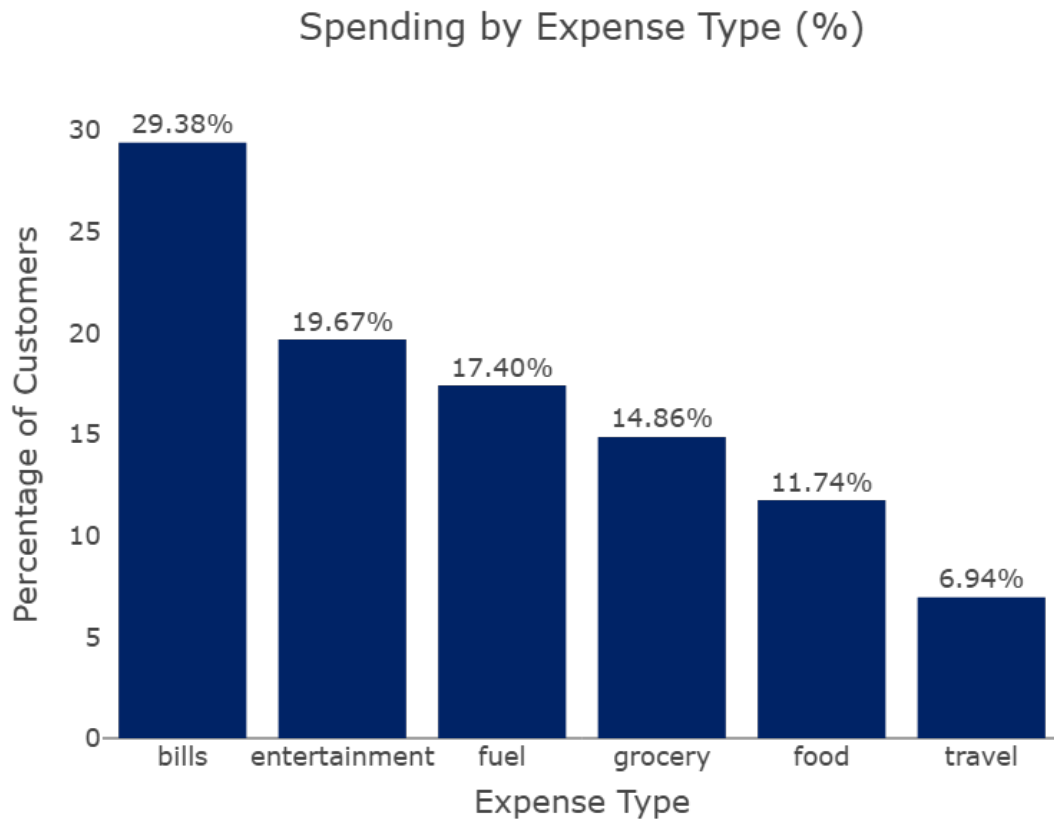
This chart breaks down total spending into categories like groceries, travel, or bills. It provides insights into where customers spend most of their money and can guide product recommendations. Bills and Entertainment likely dominate (exact percentages unclear due to missing labels).

Implication: Essential expenses (Bills/Entertainment) are primary spending drivers.

Delinquent Accounts by State

```
delinq_by_state = merged_df.groupby('state_cd')['Delinquent_Acc'].sum().reset_index(name='to
delinq_by_state = delinq_by_state.sort_values(by='total_delinquent', ascending=False)
delinq_by_state
```

	state_cd	total_delinquent
19	New York	154

	state_cd	total_delinquent
3	California	145
24	Texas	144
6	Florida	90
17	New Jersey	45
10	Iowa	6
12	Michigan	5
16	Nevada	4
26	Virginia	4
22	Pennsylvania	4
11	Massachusetts	3
0	Alaska	1
25	Utah	1
23	South Carolina	1
21	Oregon	1
14	Missouri	1
13	Minnesota	1
9	Illinois	1
7	Georgia	1
5	Connecticut	1
27	Washington	1
15	Nebraska	0
1	Arizona	0
18	New Mexico	0
20	Ohio	0
8	Hawaii	0
4	Colorado	0
2	Arkansas	0

```
total_delinq = delinq_by_state['total_delinquent'].sum()
delinq_by_state['percentage'] = (delinq_by_state['total_delinquent'] / total_delinq) * 100
delinq_by_state
```

	state_cd	total_delinquent	percentage
19	New York	154	25.081433
3	California	145	23.615635
24	Texas	144	23.452769
6	Florida	90	14.657980
17	New Jersey	45	7.328990
10	Iowa	6	0.977199

	state_cd	total_delinquent	percentage
12	Michigan	5	0.814332
16	Nevada	4	0.651466
26	Virginia	4	0.651466
22	Pennsylvania	4	0.651466
11	Massachusetts	3	0.488599
0	Alaska	1	0.162866
25	Utah	1	0.162866
23	South Carolina	1	0.162866
21	Oregon	1	0.162866
14	Missouri	1	0.162866
13	Minnesota	1	0.162866
9	Illinois	1	0.162866
7	Georgia	1	0.162866
5	Connecticut	1	0.162866
27	Washington	1	0.162866
15	Nebraska	0	0.000000
1	Arizona	0	0.000000
18	New Mexico	0	0.000000
20	Ohio	0	0.000000
8	Hawaii	0	0.000000
4	Colorado	0	0.000000
2	Arkansas	0	0.000000

```python
num = 10
delinq_by_states =  delinq_by_state.head(num)

fig = px.bar(
    delinq_by_states,
    x='total_delinquent',
    y='state_cd',
    orientation='h',
    title = f'Top {num} Delinquent Accounts by State',
    height=500,
    width=1100,
    color_discrete_sequence=['#002366'],
    text='total_delinquent'
)

fig.update_layout(
    template="presentation",
```

```
    xaxis_title='Number of Delinquent Accounts',
    yaxis_title='State',
    margin=dict(l=250, r=50, t=50, b=50),
    paper_bgcolor="rgba(0,0,0,0)",
    plot_bgcolor="rgba(0,0,0,0)",
    xaxis=dict(showgrid=False),
    yaxis=dict(showgrid=False)
)

fig.update_traces(textposition='inside')
fig.write_image(os.path.join(results_dir, 'Delinquent_Accounts_by_State.jpg'))
fig.write_image(os.path.join(results_dir, 'Delinquent_Accounts_by_State.png'))
fig.write_html(os.path.join(results_dir, 'Delinquent_Accounts_by_State.html'))

fig.show()
```


Top 10 Delinquent Accounts by State
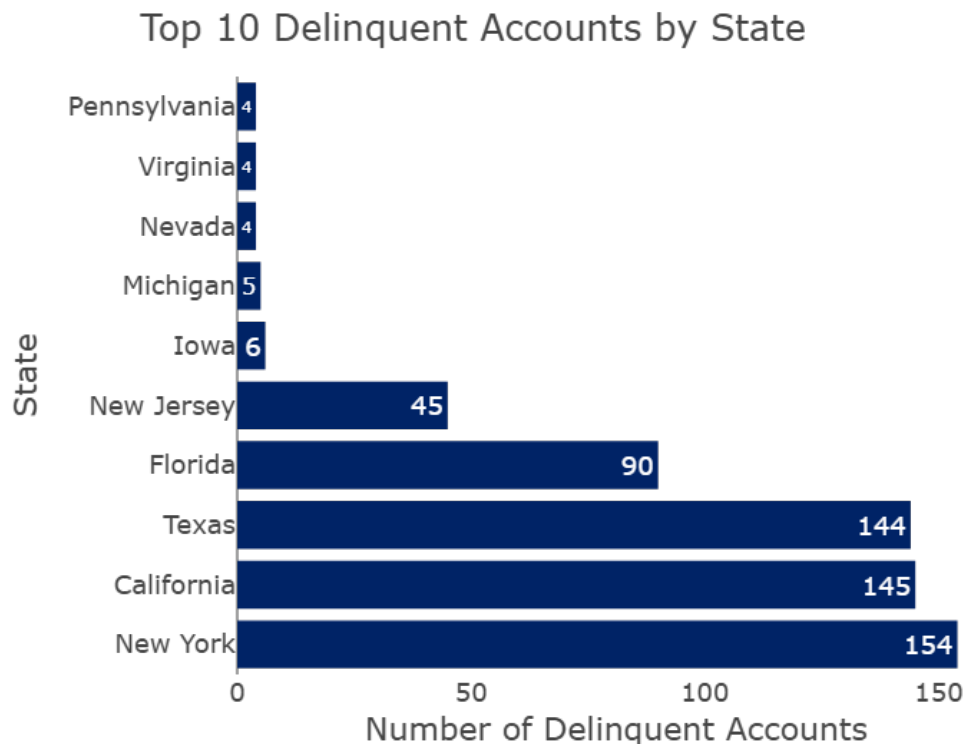
This bar chart highlights the states with the most delinquent accounts. It is useful for regional risk assessment and credit policy adjustments. Ney York (154), California (145), and Texas (144) lead in delinquencies.

Implication: Higher-risk regions may need targeted collection strategies.

Customer Count by Satisfaction Level

```
funnel_df = merged_df.groupby('Cust_Satisfaction_Score')['Client_Num'].count().reset_index(na
funnel_df = funnel_df.sort_values(by='customer_count', ascending=False)
funnel_df
```

	Cust_Satisfaction_Score	customer_count
2	3	3068
3	4	2099
4	5	1976
1	2	1792
0	1	1173

```
total_customers = funnel_df['customer_count'].sum()
funnel_df['percentage'] = (funnel_df['customer_count'] / total_customers) * 100
funnel_df
```

	Cust_Satisfaction_Score	customer_count	percentage
2	3	3068	30.352196
3	4	2099	20.765730
4	5	1976	19.548872
1	2	1792	17.728532
0	1	1173	11.604670

```
import plotly.express as px

# Group and sort
bar_df = merged_df.groupby('Cust_Satisfaction_Score')['Client_Num'].count().reset_index(name=
bar_df = bar_df.sort_values(by='customer_count', ascending = False)

# Plot
fig = px.bar(
    bar_df,
    x='Cust_Satisfaction_Score',
    y='customer_count',
    text='customer_count',
    title='Customer Count by Satisfaction Level',
    labels={'Cust_Satisfaction_Score': 'Satisfaction Score', 'customer_count': 'Number of Cus
```

```
    color_discrete_sequence=["#002366"]
)

fig.update_traces(textposition='outside')
fig.update_layout(
    template="presentation",
    paper_bgcolor="rgba(0,0,0,0)",
    plot_bgcolor="rgba(0,0,0,0)",
    height=500,
    width=800,
    xaxis=dict(showgrid=False),
    yaxis=dict(showgrid=False)
)

# Save
fig.write_image(os.path.join(results_dir, 'Customer_Count_by_Satisfaction_Bar.jpg'))
fig.write_image(os.path.join(results_dir, 'Customer_Count_by_Satisfaction_Bar.png'))
fig.write_html(os.path.join(results_dir, 'Customer_Count_by_Satisfaction_Bar.html'))

fig.show()
```

Customer Count by Satisfaction Level

This chart shows how many customers fall into different satisfaction levels (1-5). It's impor-

tant for customer experience evaluation and service improvement. Most customers cluster around mid-range satisfaction (scores 3–4), with ~2099–3088 customers. Fewer extremes (very satisfied/dissatisfied).

Implication: Service improvements could target mid-range scorers to boost loyalty.

Customer Occupation Breakdown

```
job_df = merged_df.groupby('Customer_Job')['Client_Num'].count().reset_index(name='customer_
job_df = job_df.sort_values(by='customer_count', ascending=False)
job_df
```

	Customer_Job	customer_count
4	selfemployeed	2575
1	businessman	1901
0	blue-collar	1579
5	white-collar	1542
2	govt	1525
3	retirees	986

```
total_jobs = job_df['customer_count'].sum()
job_df['percentage'] = (job_df['customer_count'] / total_jobs) * 100
job_df
```

	Customer_Job	customer_count	percentage
4	selfemployeed	2575	25.474871
1	businessman	1901	18.806886
0	blue-collar	1579	15.621290
5	white-collar	1542	15.255243
2	govt	1525	15.087060
3	retirees	986	9.754650

```
fig = px.bar(
    job_df,
    x='Customer_Job',
    y='percentage',
    title='Customer Occupation Breakdown (%)',
```

```python
    height=600,
    width=1000,
    color_discrete_sequence=['#002366'],
    text='percentage'
)

fig.update_layout(
    template="presentation",
    xaxis_title="Customer Job",
    yaxis_title="Percentage of Customers",
    paper_bgcolor="rgba(0,0,0,0)",
    plot_bgcolor="rgba(0,0,0,0)",
    xaxis=dict(showgrid=False),
    yaxis=dict(showgrid=False)
)

fig.update_traces(
    texttemplate="%{text:.2f}%",
    textposition="outside",
    marker_line_width=0
)

fig.write_image(os.path.join(results_dir, 'Customer_Job_Breakdown.jpg'))
fig.write_image(os.path.join(results_dir, 'Customer_Job_Breakdown.png'))
fig.write_html(os.path.join(results_dir, 'Customer_Job_Breakdown.html'))

fig.show()
```
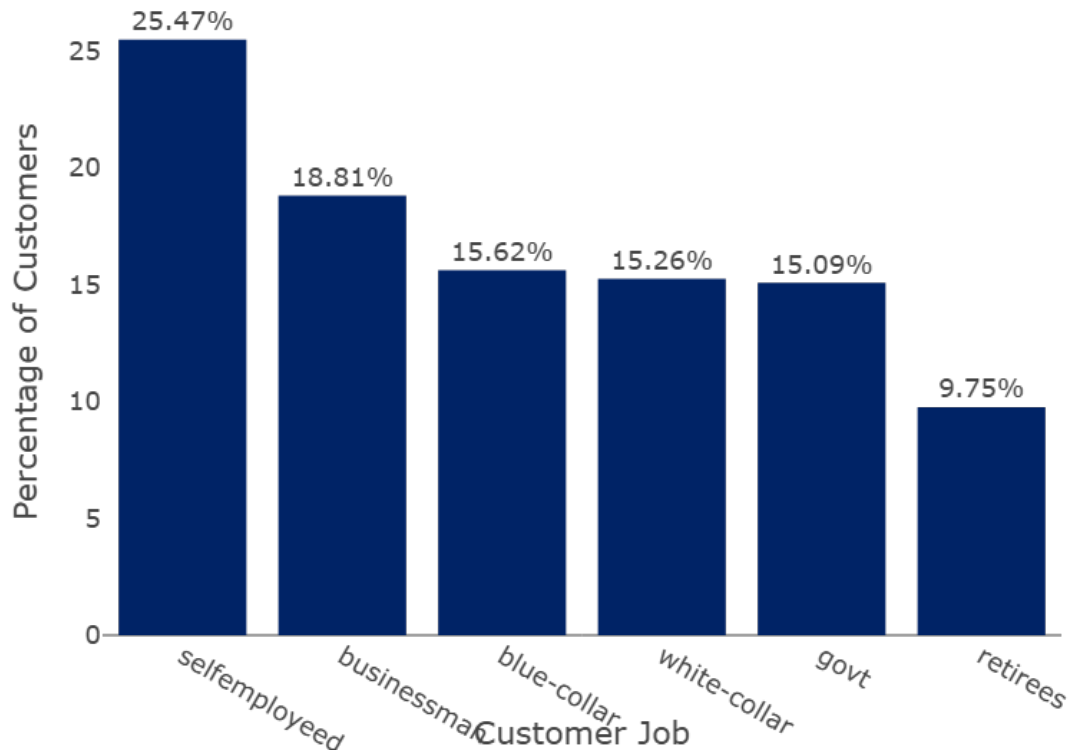
Customer Occupation Breakdown (%)

This chart presents the percentage distribution of customers across different job types. It helps in profiling the customer base and targeting services based on occupation-related income stability Top occupations: "SelfEmployed" (25.47%), BusinessMan (18.81%), blue-collar (15.62%).