CAPSTONE PROJECT-PGP-DSBA

Project Notes -2 PAGE NO -31 onwards

By Nupur Sarkar

Contents

1 Introduction of the business problem

Problem statement:

This business problem is a supervised learning example for a credit card company. The objective is to predict the probability of default (whether the customer will pay the credit card bill or not) based on the variables provided. There are multiple variables on the credit card account, purchase and delinquency information which can be used in the modelling.

PD modelling problems are meant for understanding the riskiness of the customers and how much credit is at stake in case the customer defaults. This is an extremely critical part in any organization that lends money [both secured and unsecured loans].

1.1 Defining problem statement

- The goal is to predict the likelihood of credit card customers defaulting on their payments. This includes the significance of identifying high-risk customers in advance, as it allows the bank to take pre-emptive action, such as modifying credit limits, adjusting terms, or offering assistance.
- This is a classification problem in machine learning, where the model will output a probability that a customer will or will not default.

1.2 Need of the study/project

- Discuss the financial implications for banks due to customer defaults, including lost revenue, increased costs, and risks associated with bad debts. A predictive model can help the bank minimize these risks.
- Highlight the benefits to customers as well; for instance, by identifying struggling customers early, the bank could offer financial planning or counselling resources.
- Mention how accurate predictions can improve resource allocation, reduce operational risks, and potentially lower interest rates for low-risk customers.

1.3 Understanding business/social opportunity

- From a business perspective, effective prediction of default risk enables banks to optimize lending policies, making their services more sustainable and profitable.
- Socially, this model could help banks act more responsibly toward customers by intervening before financial trouble escalates. This could reduce personal financial crises and improve customer retention.
- Emphasize the opportunity for the bank to enhance its customer relationship by offering support to at-risk customers, fostering goodwill and potentially preventing default.

2 Data Report

2.1 Understanding how data was collected in terms of time, frequency and methodology

we can infer the **time**, **frequency**, and **methodology** of how the data was collected. Here's a detailed breakdown:

1. **Time and Time Window**:

The data is structured around **time periods** that span over multiple **monthly windows**. The variables capture various **financial behaviours** of customers at different **time intervals**. Here's how the timeframes are organized:

- **Last 12 months**:
    - For many variables like **acct_amt_added_0_12m** (total amount of purchases made from 12 months ago to the present), **avg_payment_span_0_12m** (average payment span for the past 12 months), and **num_arch_ok_0_12m** (number of archived purchases that were paid in the last year), the data focuses on the **most recent 12-month period**.
- **12 to 24 months ago**:
    - Variables like **acct_amt_added_12_24m** (total amount of purchases made 12–24 months ago) and **num_arch_dc_12_24m** (number of archived purchases in debt collection status between 24 months and 12 months ago) focus on the time **between 12 and 24 months ago**.
- **3-month windows**:
    - Some variables like **acct_worst_status_0_3m** (the number of days in worst status over the last 3 months) focus on more **recent, short-term behavior**, which gives insight into the customer's more **recent financial situation**.
- **Rolling Timeframes**:
    - Other variables track data in rolling windows such as **status_max_archived_0_6_months** (maximum archived status occurrences in the past 6 months) or **acct_worst_status_6_12m** (days in worst status from 6 to 12 months ago).
    - This suggests that the data was likely collected and processed at regular intervals, possibly monthly or quarterly, to capture **ongoing** patterns in customer behaviour.

2. **Frequency**:

- The data seems to be collected on a **monthly** or **quarterly** basis, based on the use of time-based variables that aggregate or segment customer behaviours by **monthly periods** or **12-month windows**.
    - For example, **acct_amt_added_12_24m** aggregates **total spending over 12 months** and **acct_amt_added_0_12m** looks at the most recent **12-month period**.
    - Variables like **num_arch_dc_0_12m** (number of archived debt collection items in the last year) would be updated periodically, either **monthly or quarterly**, reflecting the frequency of account status changes or purchases.

3. **Methodology**:

From the data provided, the **methodology** for data collection can be inferred in terms of what is being tracked and how it's structured:

- **Tracking Purchases**:
    - Variables like **acct_amt_added_12_24m**, **max_paid_inv_0_12m**, and **sum_paid_inv_0_12m** track **purchases made** and **amounts paid** during specific periods, suggesting the data comes from **transaction records** related to each credit card user's activity.
    - **Merchant categories** (e.g., **merchant_category**, **merchant_group**) provide information about where the customer spent money, which can be tied back to **merchant transaction data**.
- **Tracking Account Status**:
    - The status-related variables (**acct_days_in_dc_12_24m**, **acct_days_in_rem_12_24m**, **acct_worst_status_0_3m**, etc.) suggest that the data includes information on how **credit card accounts** progress over time, particularly in relation to **payment delays, reminders, debt collections,** and **account terminations**.

    - This is likely tracked via **account management systems** or through automated reporting systems in which the status of a customer's account is updated in real-time or at regular intervals.
- **Tracking Payments and Defaults**:
    - Variables like **has_paid** (whether the customer paid the bill), **sum_capital_paid_acct_0_12m**, and **recovery_debt** (total debt recovered) suggest that payment data is being tracked as part of the **payment history**.
    - The methodology behind this would likely involve aggregating data from **payment processing systems** and **collections systems**, which track whether a customer paid, made partial payments, or defaulted.
- **Debt Recovery and Collection**:
    - Variables like **acct_incoming_debt_vs_paid_0_24m** and **num_arch_dc_0_12m** (number of archived debt collection items) indicate that **debt recovery and collections data** is collected as part of the methodology. This would involve collecting data from **third-party debt recovery agencies** and **internal collections systems**.

4. **Data Collection Methodology**:

The methodology behind the data collection likely involves a mix of the following systems and processes:

- **Transactional Data Collection**:
    - Data is collected through **transaction records** that capture **purchases**, **payments**, and **account status changes**. These records are typically gathered by the credit card company's internal systems.
- **Account Monitoring and Status Tracking**:
    - The credit card company likely uses **automated monitoring systems** to track the status of each account, including whether the customer is in good standing, has missed payments, is in a reminder state, debt collection state, or has had their card terminated. This data is regularly updated to reflect the current status.
- **Debt Collection Systems**:
    - Data is also likely collected from **external debt collection agencies** that monitor the recovery of overdue balances. Variables related to debt collection (**num_arch_dc_0_12m**, **acct_incoming_debt_vs_paid_0_24m**) indicate that some of the data is sourced from **external partners** (e.g., debt recovery agencies).
- **Periodic Updates**:
    - Since many of the variables cover rolling time periods (e.g., **12 months ago to the present**, **last 3 months**), the data is likely updated on a **monthly or quarterly basis** to reflect the most recent customer activity and account status changes.

Summary of Data Collection Methodology:

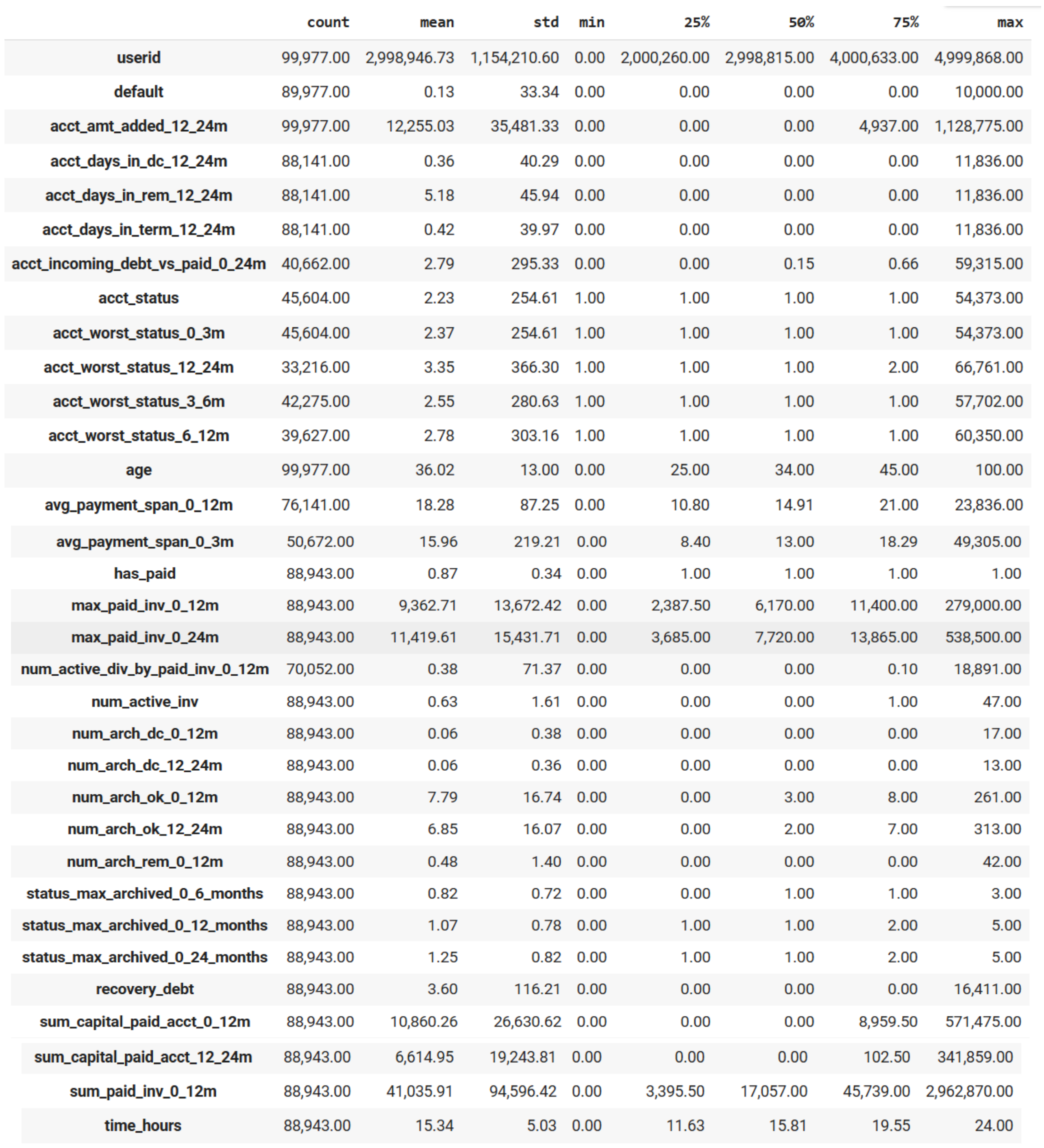
1. **Time**: Data is collected for specific **time windows**—mainly 12-month or 24-month periods, but with some variables also tracking **3-month windows**.
2. **Frequency**: The data appears to be **updated regularly**, likely on a **monthly** or **quarterly** basis, as the variables track activity over periods like "last 12 months" or "12–24 months ago."
3. **Methodology**: The data is collected through a combination of:
    - **Transaction and payment records** from customer activity on their credit card.
    - **Account status tracking** (e.g., reminders, collections, terminations).
    - **Debt collection systems**, including internal systems and external partners like collection agencies.

This structured and periodic approach allows the credit card company to continuously monitor the status of customer accounts, their spending behaviour, and any outstanding debts, which helps in assessing risk and predicting the probability of default.

2.2 Visual inspection of data (rows, columns, descriptive details)

Dataset shape (rows, columns): (99979, 36).

Descriptive statistics for numerical columns:



	count	mean	std	min	25%	50%	75%	max
userid	99,977.00	2,998,946.73	1,154,210.60	0.00	2,000,260.00	2,998,815.00	4,000,633.00	4,999,868.00
default	89,977.00	0.13	33.34	0.00	0.00	0.00	0.00	10,000.00
acct_amt_added_12_24m	99,977.00	12,255.03	35,481.33	0.00	0.00	0.00	4,937.00	1,128,775.00
acct_days_in_dc_12_24m	88,141.00	0.36	40.29	0.00	0.00	0.00	0.00	11,836.00
acct_days_in_rem_12_24m	88,141.00	5.18	45.94	0.00	0.00	0.00	0.00	11,836.00
acct_days_in_term_12_24m	88,141.00	0.42	39.97	0.00	0.00	0.00	0.00	11,836.00
acct_incoming_debt_vs_paid_0_24m	40,662.00	2.79	295.33	0.00	0.00	0.15	0.66	59,315.00
acct_status	45,604.00	2.23	254.61	1.00	1.00	1.00	1.00	54,373.00
acct_worst_status_0_3m	45,604.00	2.37	254.61	1.00	1.00	1.00	1.00	54,373.00
acct_worst_status_12_24m	33,216.00	3.35	366.30	1.00	1.00	1.00	2.00	66,761.00
acct_worst_status_3_6m	42,275.00	2.55	280.63	1.00	1.00	1.00	1.00	57,702.00
acct_worst_status_6_12m	39,627.00	2.78	303.16	1.00	1.00	1.00	1.00	60,350.00
age	99,977.00	36.02	13.00	0.00	25.00	34.00	45.00	100.00
avg_payment_span_0_12m	76,141.00	18.28	87.25	0.00	10.80	14.91	21.00	23,836.00
avg_payment_span_0_3m	50,672.00	15.96	219.21	0.00	8.40	13.00	18.29	49,305.00
has_paid	88,943.00	0.87	0.34	0.00	1.00	1.00	1.00	1.00
max_paid_inv_0_12m	88,943.00	9,362.71	13,672.42	0.00	2,387.50	6,170.00	11,400.00	279,000.00
max_paid_inv_0_24m	88,943.00	11,419.61	15,431.71	0.00	3,685.00	7,720.00	13,865.00	538,500.00
num_active_div_by_paid_inv_0_12m	70,052.00	0.38	71.37	0.00	0.00	0.00	0.10	18,891.00
num_active_inv	88,943.00	0.63	1.61	0.00	0.00	0.00	1.00	47.00
num_arch_dc_0_12m	88,943.00	0.06	0.38	0.00	0.00	0.00	0.00	17.00
num_arch_dc_12_24m	88,943.00	0.06	0.36	0.00	0.00	0.00	0.00	13.00
num_arch_ok_0_12m	88,943.00	7.79	16.74	0.00	0.00	3.00	8.00	261.00
num_arch_ok_12_24m	88,943.00	6.85	16.07	0.00	0.00	2.00	7.00	313.00
num_arch_rem_0_12m	88,943.00	0.48	1.40	0.00	0.00	0.00	0.00	42.00
status_max_archived_0_6_months	88,943.00	0.82	0.72	0.00	0.00	1.00	1.00	3.00
status_max_archived_0_12_months	88,943.00	1.07	0.78	0.00	1.00	1.00	2.00	5.00
status_max_archived_0_24_months	88,943.00	1.25	0.82	0.00	1.00	1.00	2.00	5.00
recovery_debt	88,943.00	3.60	116.21	0.00	0.00	0.00	0.00	16,411.00
sum_capital_paid_acct_0_12m	88,943.00	10,860.26	26,630.62	0.00	0.00	0.00	8,959.50	571,475.00
sum_capital_paid_acct_12_24m	88,943.00	6,614.95	19,243.81	0.00	0.00	0.00	102.50	341,859.00
sum_paid_inv_0_12m	88,943.00	41,035.91	94,596.42	0.00	3,395.50	17,057.00	45,739.00	2,962,870.00
time_hours	88,943.00	15.34	5.03	0.00	11.63	15.81	19.55	24.00

Figure 1(Statistical summary)

Insights from the Statistical Summary

General Observations

High Variance Across Features:

Some features exhibit high variability (e.g., acct_amt_added_12_24m has a standard deviation of 35,481.33 and a maximum value of 1,128,775, indicating outliers or extreme values).

Features like acct_days_in_dc_12_24m, acct_days_in_rem_12_24m, and acct_days_in_term_12_24m have low mean values compared to their max, suggesting most customers spend little to no time in these statuses.

Target Variable (default):

The mean is 0.13, indicating that about 13% of the customers defaulted.

However, the maximum value of 10,000 appears anomalous, suggesting data issues or incorrect representation of the default column.

Key Insights per Feature:

acct_amt_added_12_24m:

Average credit card purchase amount over this period is 12,255.03, with many customers having no purchases (median = 0).

Significant outliers exist (max = 1,128,775), possibly representing high spenders.

acct_days_in_dc_12_24m, acct_days_in_rem_12_24m, and acct_days_in_term_12_24m:

Mean values for time spent in these statuses are very low:

Debt-Collection Status: Mean = 0.36 days.

Reminder Status: Mean = 5.18 days.

Termination Status: Mean = 0.42 days.

High max values (11,836 days) suggest outliers or potential miscalculation.

acct_incoming_debt_vs_paid_0_24m:

Mean = 2.79, indicating that agencies generally recover a fraction of the total debt.

The high standard deviation (295.33) and max value (59,315) imply a wide range of recovery success rates.

acct_status and acct_worst_status Features:

Most accounts are active (median = 1).

acct_worst_status features show a skewed distribution, with most values clustered at 1 (indicating accounts rarely reach the worst status).

Age:

Mean customer age is 36.02 years, with a reasonable distribution across a wide age range (min = 0, max = 100).

Payment Patterns (avg_payment_span_0_12m, avg_payment_span_0_3m):

Median values (14.91 and 13.00 days, respectively) suggest customers generally pay within 2-3 weeks after bill generation.

Large max values (e.g., 49,305 days for avg_payment_span_0_3m) indicate extreme cases or data quality issues.

Debt Recovery (recovery_debt):

Mean recovery is 3.60, but a high standard deviation (116.21) indicates most cases involve little to no recovery, with a few extreme cases.

High Variability in Transactions:

max_paid_inv_0_12m (mean = 9,362.71, max = 279,000) and sum_paid_inv_0_12m (mean = 41,035.91, max = 2,962,870) highlight outliers with significantly higher spending patterns.

---

num_active_div_by_paid_inv_0_12m (Ratio of unpaid to paid bills over the last year)

Mean: 0.38, indicating that, on average, unpaid bills constitute 38% of paid bills.

Standard Deviation: 71.37, showing high variability.

Median (50th percentile): 0.00, meaning at least half of the customers have no unpaid bills relative to paid ones.

Max: 18,891.00, suggesting a few extreme outliers with very high ratios.

***Insight*:** Most customers maintain a good payment pattern with no unpaid bills, but there are outliers with significantly high unpaid ratios that could represent risky behavior.

---

num_active_inv (Number of active unpaid invoices)

Mean: 0.63, indicating that most customers have less than 1 unpaid invoice on average.

Median: 0.00, suggesting that more than half of the customers have no active unpaid invoices.

Max: 47.00, highlighting a few customers with a large number of unpaid invoices.

Insight: While most customers have no unpaid invoices, those with multiple unpaid invoices could indicate credit risk.

---

num_arch_dc_0_12m (Archived purchases in debt collection in the last year) and **num_arch_dc_12_24m (Archived purchases in debt collection 12–24 months ago)**

Both have low means (0.06), medians of 0, and maximums of 17 and 13, respectively.

Most customers have no archived purchases in debt collection, but there are occasional cases with a few such instances.

Insight: Debt collection cases are rare, but monitoring trends over time is essential to identify deteriorating behavior.

---

num_arch_ok_0_12m (Archived purchases paid in the last year) and **num_arch_ok_12_24m (Archived purchases paid 12–24 months ago)**

Mean: 7.79 (last year), 6.85 (12–24 months ago).

Median: 3 (last year), 2 (12–24 months ago), showing a higher recent payment activity.

Max: 261 and 313, indicating some high activity outliers.

Insight**:** Payment activity is relatively consistent, but some customers have unusually high payment activity, possibly due to large transactions or repayment of accumulated debts.

---

num_arch_rem_0_12m (Archived purchases in reminder status in the last year)

Mean: 0.48, with a median of 0, suggesting that most customers did not have purchases in reminder status.

Max: 42, reflecting a few cases where customers required frequent reminders.

Insight**:** While reminders are uncommon for most, a small subset of customers may need interventions to encourage timely payments.

---

status_max_archived_0_6_months, 0_12_months, and 0_24_months (Maximum times account was archived in different periods)

Mean increases from 0.82 (last 6 months) to 1.07 (last year) and 1.25 (last 2 years).

Median is 1 across all periods, meaning most customers had their accounts archived at least once.

Max: 3, 5, and 5, respectively, for the time periods.

Insight**:** Account archiving is relatively common, with the frequency increasing over longer periods, suggesting recurring issues for some customers.

---

recovery_debt (Total recovered debt)

Mean: 3.60, with a median of 0, showing that most customers do not have recoverable debt.

Max: 16,411.00, indicating significant recoveries in a few extreme cases.

Insight: Debt recovery is minimal for most accounts but substantial in a few outliers, likely involving large overdue amounts.

---

sum_capital_paid_acct_0_12m and 12_24m (Principal balance paid on account in the last year and 12–24 months ago)

Mean: 10,860.26 (last year) and 6,614.95 (12–24 months ago), showing higher payments in the recent year.

Median: 8,959.50 (last year) and 102.50 (12–24 months ago), highlighting a significant increase in recent payments.

Max: 571,475.00 and 341,859.00, reflecting large transactions in some cases.

***Insight*:** Principal payments have increased significantly in the recent year, possibly indicating improved financial capacity or increased debt repayment efforts.

---

sum_paid_inv_0_12m (Total paid invoices in the last year)

Mean: 41,035.91, with a median of 17,057.00.

Max: 2,962,870.00, showing some customers with extremely high payments.

***Insight*:** Payments are substantial for most customers, but a few outliers represent high-value accounts or businesses.

---

time_hours (Total hours spent on credit card purchases)

Mean: 15.34, with a median of 15.81, showing consistent activity across customers.

Max: 24.00, indicating a capped time frame.

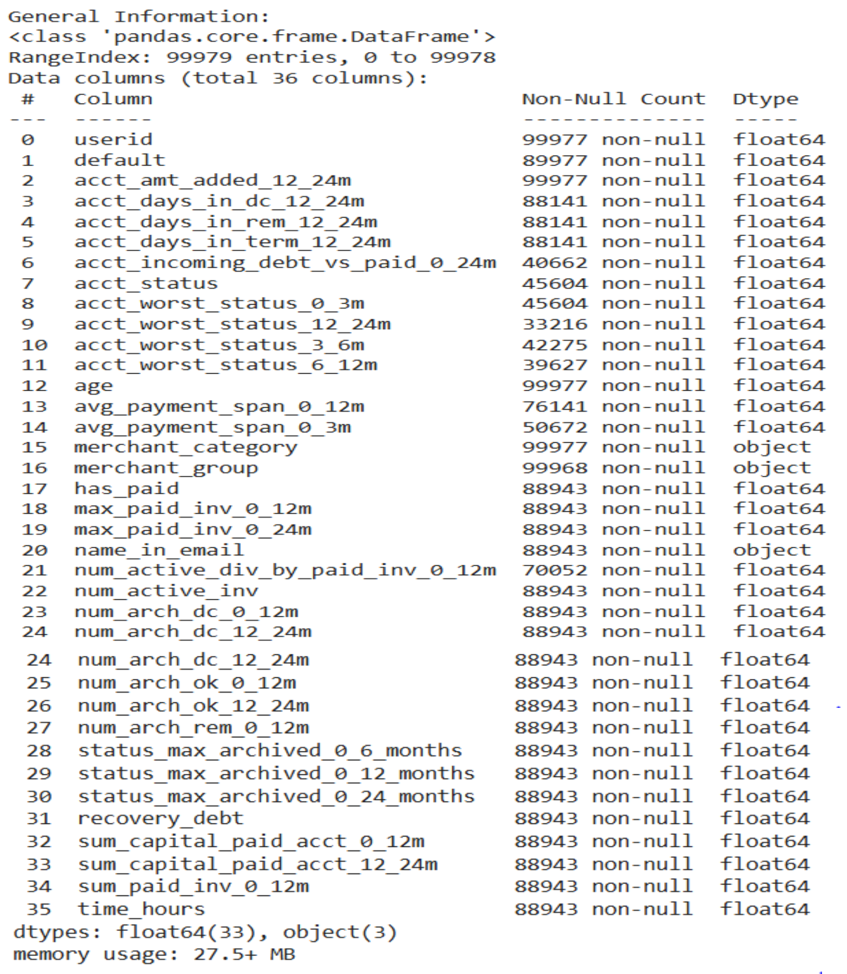
***Insight*:** Most customers use their credit cards consistently, with minimal variation in time spent.

Key Insights

1. **acct_amt_added_12_24m**: Average purchases are moderate, with a few high spenders contributing to extreme outliers.
2. **acct_days_in_dc_12_24m, acct_days_in_rem_12_24m, acct_days_in_term_12_24m**: Most customers spend negligible time in these statuses, but outliers suggest rare long-term issues.
3. **acct_incoming_debt_vs_paid_0_24m**: Debt recovery rates vary widely, with most customers recovering only a small fraction of their debts.

4. **acct_status and acct_worst_status Features**: Accounts are generally active, with rare occurrences of reaching worst statuses.
5. **Age**: Customers are mostly working-age adults, with a mean age of 36 years.
6. **Payment Patterns (avg_payment_span_0_12m, avg_payment_span_0_3m)**: Customers generally pay within 2–3 weeks, with some extreme delays indicating potential issues.
7. **Debt Recovery (recovery_debt)**: Most customers have minimal debt recovery, though some cases involve substantial amounts.
8. **High Variability in Transactions**: Outliers in invoice payments indicate high-value customers or businesses significantly impacting totals.
9. **num_active_div_by_paid_inv_0_12m**: Most customers have no unpaid bills, but extreme ratios indicate potential credit risk.
10. **num_active_inv**: Most customers have no unpaid invoices, though a small segment has multiple unpaid bills.
11. **num_arch_dc_0_12m and num_arch_dc_12_24m**: Debt collection cases are rare, but a few customers exhibit recurring financial issues.
12. **num_arch_ok_0_12m and num_arch_ok_12_24m**: Payment activity is slightly higher in the recent year, suggesting improved repayment behavior.
13. **num_arch_rem_0_12m**: Most customers don't require reminders, though a small subset relies heavily on them.
14. **status_max_archived_0_6_months, 0_12_months, and 0_24_months**: Account archiving increases over time, highlighting recurring payment issues.
15. **sum_capital_paid_acct_0_12m and 12_24m**: Recent principal payments are significantly higher, indicating improved financial behavior.
16. **sum_paid_inv_0_12m**: Payments are substantial for most, with high-value customers contributing to outliers.
17. **time_hours**: Customers show consistent usage patterns, with limited variation in time spent on credit card purchases.

2.3 Understanding of attributes (variable info, renaming if required)



```
General Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99979 entries, 0 to 99978
Data columns (total 36 columns):
 #   Column                             Non-Null Count  Dtype
---  ------                             --------------  -----
 0   userid                             99977 non-null  float64
 1   default                            89977 non-null  float64
 2   acct_amt_added_12_24m              99977 non-null  float64
 3   acct_days_in_dc_12_24m             88141 non-null  float64
 4   acct_days_in_rem_12_24m            88141 non-null  float64
 5   acct_days_in_term_12_24m           88141 non-null  float64
 6   acct_incoming_debt_vs_paid_0_24m   40662 non-null  float64
 7   acct_status                        45604 non-null  float64
 8   acct_worst_status_0_3m             45604 non-null  float64
 9   acct_worst_status_12_24m           33216 non-null  float64
 10  acct_worst_status_3_6m             42275 non-null  float64
 11  acct_worst_status_6_12m            39627 non-null  float64
 12  age                                99977 non-null  float64
 13  avg_payment_span_0_12m             76141 non-null  float64
 14  avg_payment_span_0_3m              50672 non-null  float64
 15  merchant_category                  99977 non-null  object
 16  merchant_group                     99968 non-null  object
 17  has_paid                           88943 non-null  float64
 18  max_paid_inv_0_12m                 88943 non-null  float64
 19  max_paid_inv_0_24m                 88943 non-null  float64
 20  name_in_email                      88943 non-null  object
 21  num_active_div_by_paid_inv_0_12m   70052 non-null  float64
 22  num_active_inv                     88943 non-null  float64
 23  num_arch_dc_0_12m                  88943 non-null  float64
 24  num_arch_dc_12_24m                 88943 non-null  float64
 24  num_arch_dc_12_24m                 88943 non-null  float64
 25  num_arch_ok_0_12m                  88943 non-null  float64
 26  num_arch_ok_12_24m                 88943 non-null  float64
 27  num_arch_rem_0_12m                 88943 non-null  float64
 28  status_max_archived_0_6_months     88943 non-null  float64
 29  status_max_archived_0_12_months    88943 non-null  float64
 30  status_max_archived_0_24_months    88943 non-null  float64
 31  recovery_debt                      88943 non-null  float64
 32  sum_capital_paid_acct_0_12m        88943 non-null  float64
 33  sum_capital_paid_acct_12_24m       88943 non-null  float64
 34  sum_paid_inv_0_12m                 88943 non-null  float64
 35  time_hours                         88943 non-null  float64
dtypes: float64(33), object(3)
memory usage: 27.5+ MB
```

Figure 2(Info)

Insights from the info

1.General Information

The dataset consists of **99,979 rows** and **36 columns**.

Most columns have a datatype of float64, while three columns are of type object.

It consumes **27.5 MB** of memory.

2.Missing Data

6 columns have more than 50% missing values:

acct_incoming_debt_vs_paid_0_24m (59.33% missing)

acct_status (54.39% missing)

acct_worst_status_0_3m (54.39% missing)

acct_worst_status_12_24m (66.78% missing)

acct_worst_status_3_6m (57.72% missing)

acct_worst_status_6_12m (60.36% missing)

High missing values in these columns could impact model performance and may require:

Imputation (using mean, median, or predictive models).

Exclusion if the columns are deemed uninformative or irrelevant.

Key Observations

Target Variable: The default column, which likely represents the target for prediction (e.g., probability of default), has 89,977 non-null entries, with **10% missing data**. This may require imputation or careful handling before modelling.

Other Columns of Interest:

userid has 2 missing values, which is be due to data entry issues.

Columns related to payments (has_paid, max_paid_inv_0_12m, sum_paid_inv_0_12m) have ~11% missing values.

Financial activity columns such as num_active_inv and num_archived_ have consistent non-null values (~88,943).

Age and Demographics:

age is complete and may serve as a useful demographic feature.

Categorical features like merchant_category, merchant_group, and name_in_email are mostly complete (>99.9% non-null).

Account Activity:

Columns like acct_days_in_dc_12_24m and acct_days_in_rem_12_24m have **11.8% missing values**, indicating incomplete activity tracking.

acct_amt_added_12_24m has full data and could help measure account growth trends.

Recommendations

Handle Missing Data:

Impute values for features with moderate missing data (<30% missing).

Consider dropping or imputing features with high missing values (>50%) depending on their importance.

Feature Engineering:

Derive new features, such as ratios or aggregated statistics, from existing numerical columns to capture more trends in account behavior.

Transform categorical features (merchant_category, merchant_group) into numerical representations using techniques like one-hot encoding.

Exploratory Data Analysis:

Investigate distributions of key numerical features.

Analyze relationships between the default column and other features to identify potential predictors.

Modeling:

Use robust models that can handle missing data (e.g., XGBoost, LightGBM).

Perform hyperparameter tuning to optimize predictive performance.

3 Exploratory data analysis

3.1 Univariate analysis (distribution and spread for every continuous attribute, distribution of data in categories for categorical ones)

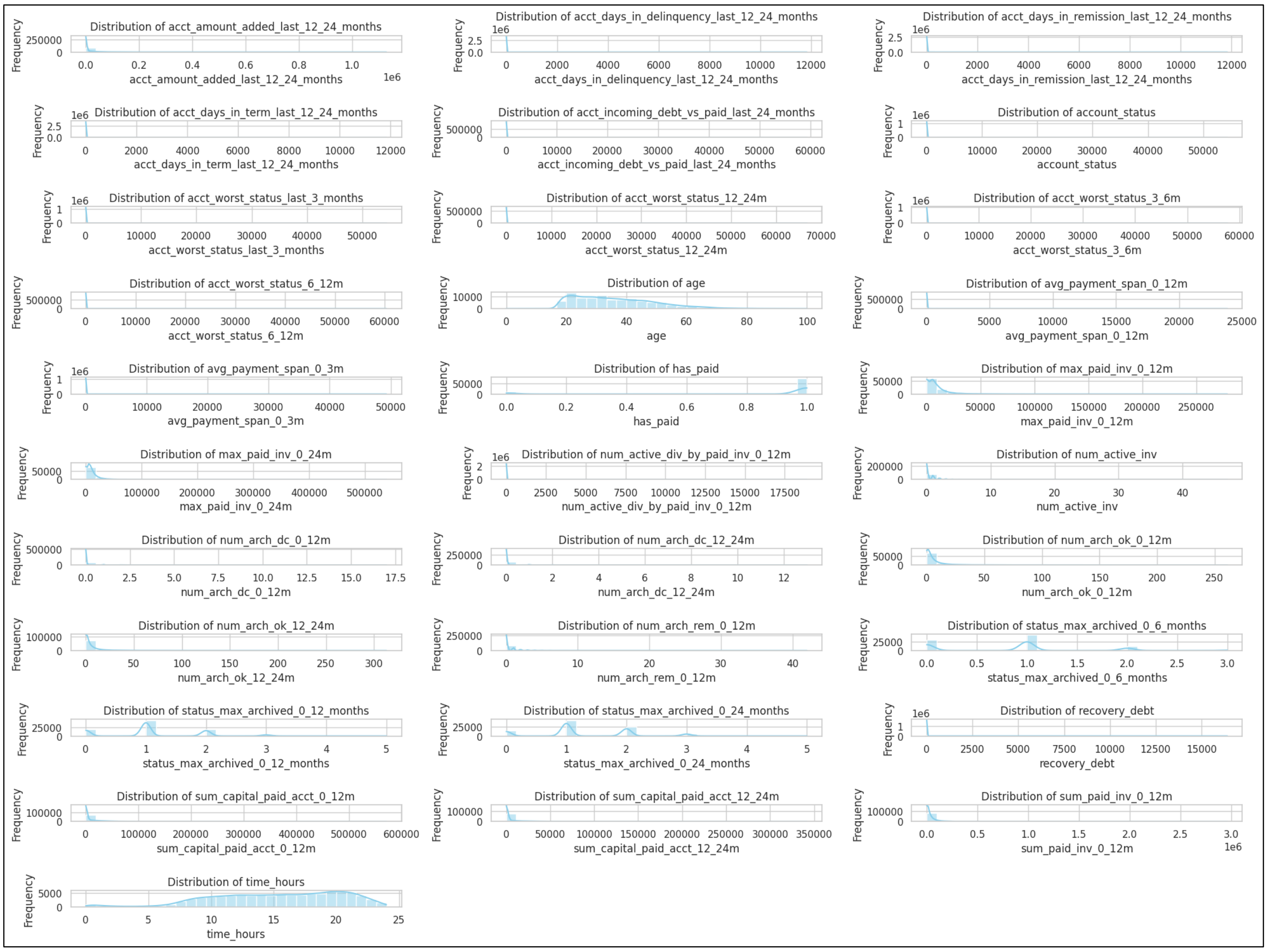


Figure 3(Spread of Continuous variables)

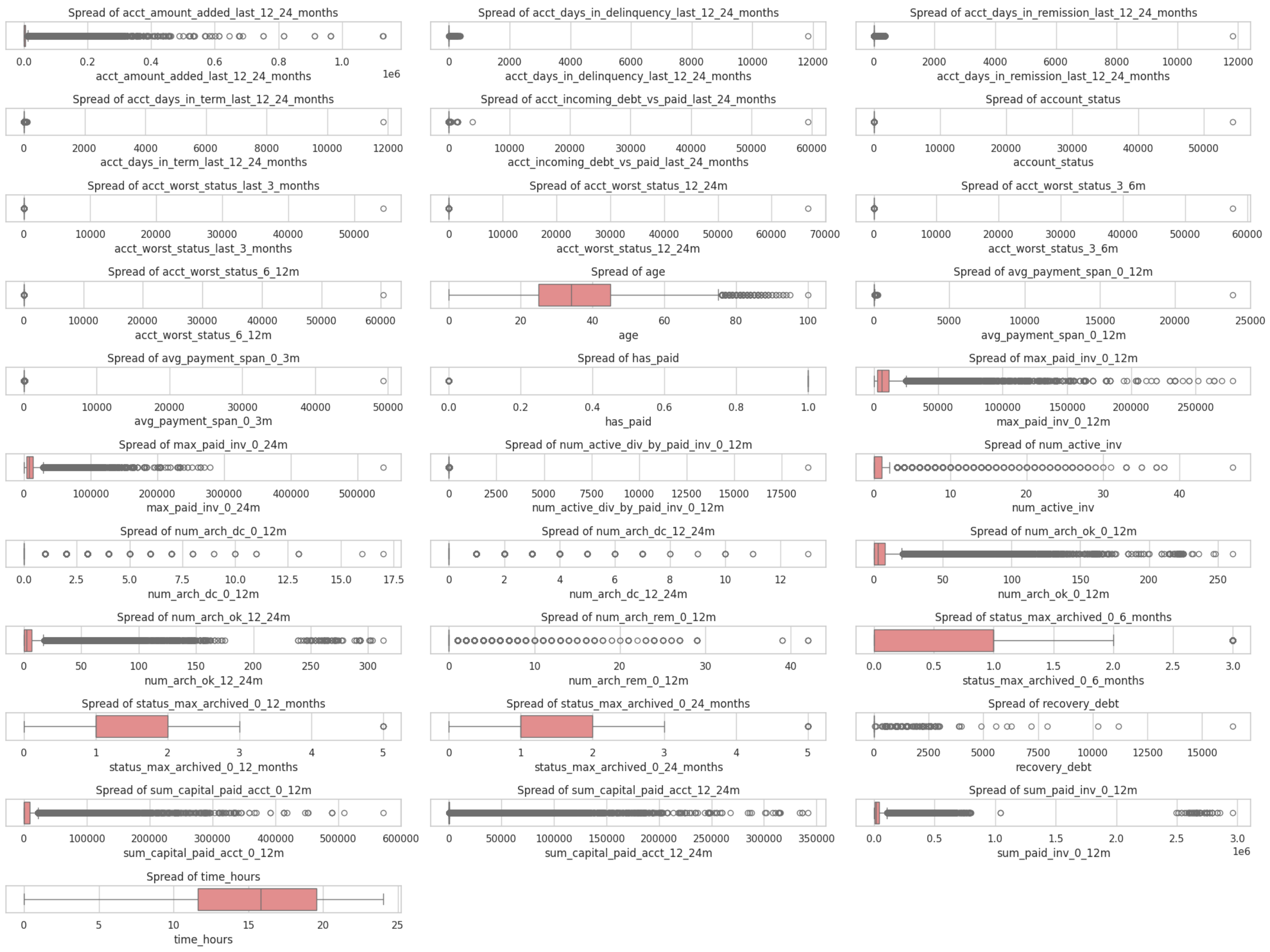


Figure 4(Box-plot Spread of Numerical Variables)

boxplots for the spread of features, here are key insights:

---

1. General Observations

Presence of Outliers: Many features exhibit significant outliers, as evidenced by extreme points far outside the whiskers.

Skewed Distributions: A majority of features show non-normal distributions with long tails, indicating the presence of highly skewed data.

---

2. Individual Feature Insights

a. Features with Significant Outliers

acct_amt_added_12_24m:

Strong skew with extreme outliers reaching up to millions.

Most values lie near zero, indicating very few accounts have large additions.

max_paid_inv_0_24m:

Outliers above 250,000 are present.

The majority of accounts involve significantly smaller payments.

sum_paid_inv_0_12m:

Similar to max_paid_inv_0_24m, with outliers far above the median value.

Suggests a small percentage of accounts have extremely high payment activity.

recovery_debt:

While most accounts have low recovery debt, a few have exceptionally high amounts above 10,000.

---

b. Time and Account Status Features

acct_days_in_term_12_24m:

Few accounts with long terms (beyond 10,000 days), which appear as extreme outliers.

Majority of accounts cluster at shorter durations.

acct_days_in_rem_12_24m & acct_days_in_dc_12_24m:

Similar patterns with rare but extreme outliers.

status_max_archived_0_12_months & status_max_archived_0_24_months:

Majority of values are concentrated in a few categories, but small outlier values exist.

---

c. Payment Behavior Features

has_paid:

Binary-like distribution without major anomalies.

Implies a clear separation between customers who paid and those who did not.

avg_payment_span_0_12m & avg_payment_span_0_3m:

Some outliers exist for long payment spans, but the bulk of accounts lie near the median.

sum_capital_paid_acct_*:

Features such as sum_capital_paid_acct_0_12m show outliers beyond 100,000, indicating a small percentage of customers who paid significantly high amounts.

---

d. Activity and Merchant Features

num_arch_ok_12_24m & num_arch_dc_12_24m:

Both features show a concentration near lower values with rare high outliers.

num_active_inv:

Highly skewed with outliers for customers with large numbers of active invoices.

---

3. Insights on Handling Outliers

High-Priority Features for Outlier Handling:

Financial metrics like acct_amt_added_12_24m, max_paid_inv_0_24m, and sum_paid_inv_0_12m require robust techniques to handle extreme values, such as capping or transformation.

Features with Limited Impact:

Binary or categorical-like features (e.g., has_paid, status_max_archived_*) show fewer extreme outliers, so standardization may suffice.

---

4. Recommendations

Log Transformation:

Apply to features with extreme right-skewness (acct_amt_added_12_24m, max_paid_inv_0_24m) to reduce the impact of outliers.

Winsorization:

Cap outliers for features like recovery_debt and sum_capital_paid_acct_* to retain important variability while mitigating noise.

Feature Scaling:

Use Min-Max or Standard Scaler post-outlier handling to normalize features for model input.

Segment Analysis:

Outliers could represent meaningful subgroups (e.g., high-paying or long-term customers). Consider clustering or segmentation before removing them entirely.

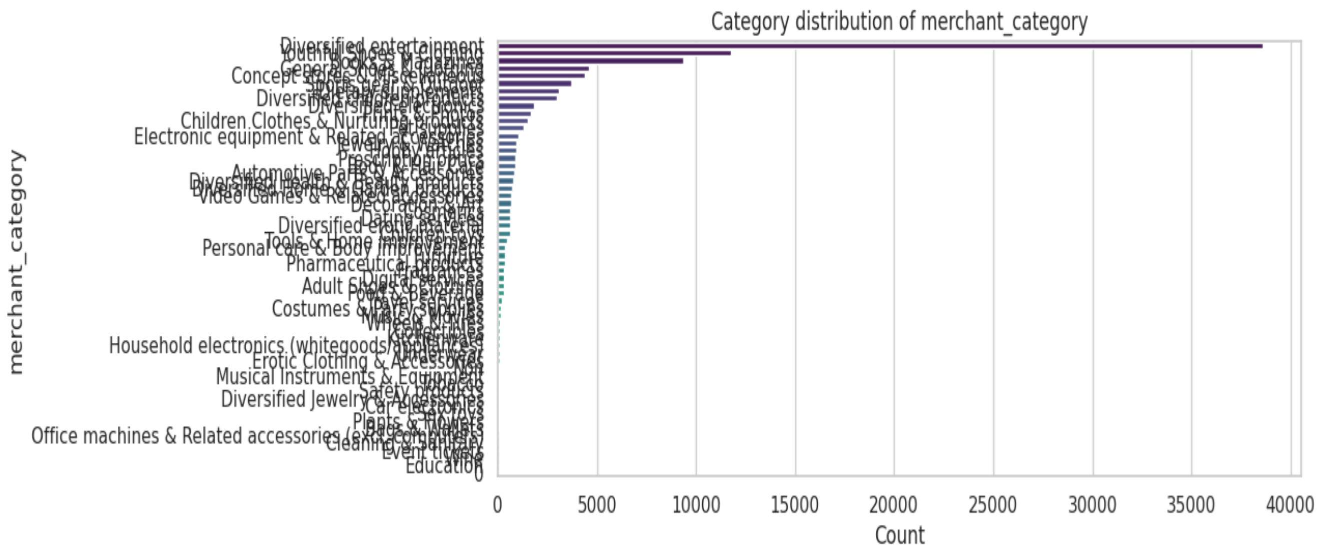


Figure 5(Merchat Category Distribution)

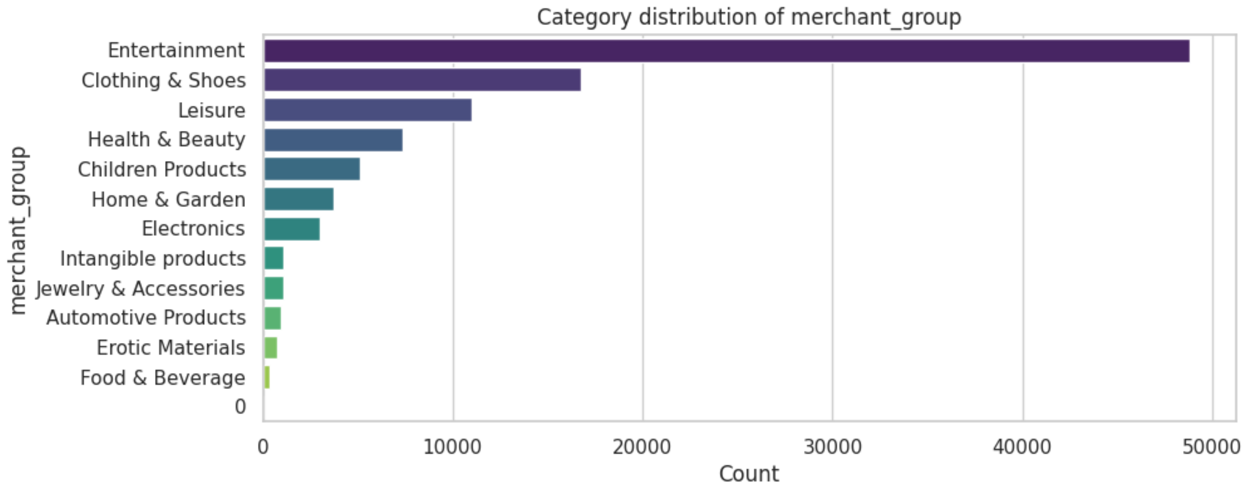


Figure 7(Merchart Group Category Distribution)

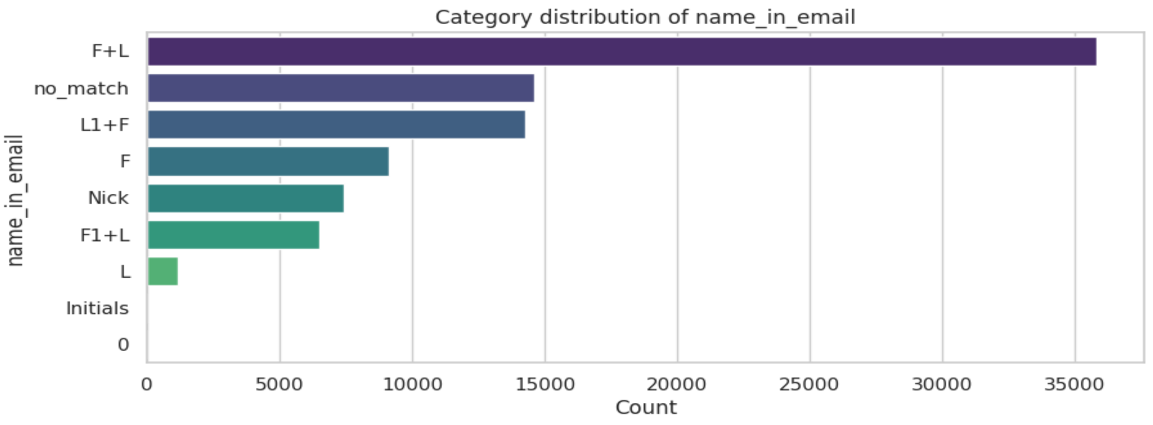


Figure 6(Name in E-mail Category Distribution)

The chart shows a **skewed distribution** of transactions across merchant categories, with a few categories (e.g., "Diversified Entertainment" and "Concept Stores") dominating. Many smaller categories have low transaction counts, creating a **long-tail effect**. This suggests potential **data imbalance**, which could impact modeling.

Key Takeaways:

Focus on top categories for marketing or customer analysis.

Address imbalances in models using resampling or weighted metrics.

Explore what drives the popularity of top categories and opportunities in low-performing ones.

3.2 Bivariate analysis (relationship between different variables, correlations)

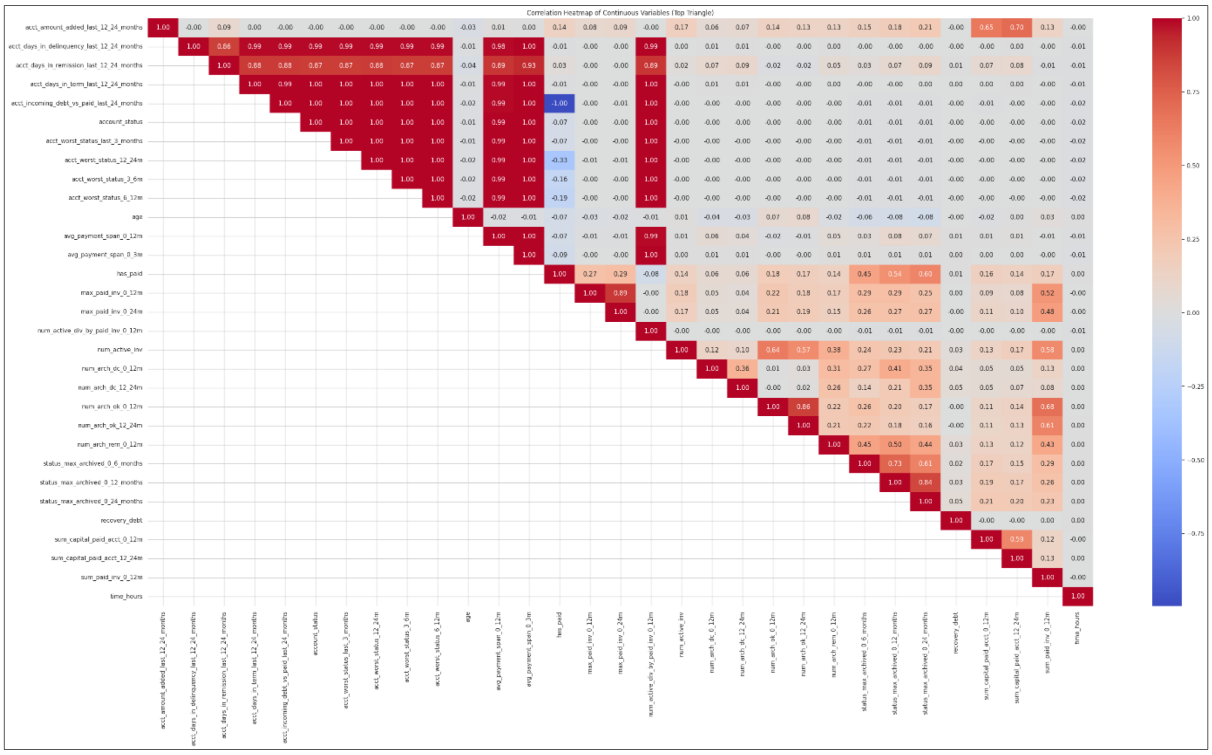


Figure 8(Correlation Matrix)

Insights :

Highly correlated pairs (|correlation| > 0.7):

max_paid_inv_0_24m	max_paid_inv_0_12m	0.89
max_paid_inv_0_12m	max_paid_inv_0_24m	0.89
num_arch_ok_12_24m	num_arch_ok_0_12m	0.87
num_arch_ok_0_12m	num_arch_ok_12_24m	0.87

```
status_max_archived_0_24_months     status_max_archived_0_12_months      0.84
status_max_archived_0_12_months     status_max_archived_0_24_months      0.84
avg_payment_span_0_3m               avg_payment_span_0_12m               0.74
avg_payment_span_0_12m              avg_payment_span_0_3m                0.74
status_max_archived_0_6_months      status_max_archived_0_12_months      0.73
status_max_archived_0_12_months     status_max_archived_0_6_months       0.73
acct_amount_added_last_12_24_months sum_capital_paid_acct_12_24m         0.71
sum_capital_paid_acct_12_24m        acct_amount_added_last_12_24_months  0.71
dtype: float64
```

Key Insights in Short:

High Correlation Between Payments (max_paid_inv_0_12m & max_paid_inv_0_24m, 0.89): Customers who pay large amounts in the last 12 months likely paid similarly in the last 24 months. This suggests consistent payers are low-risk, and these features are redundant.

Frequent Payments in Archive Status (num_arch_ok_0_12m & num_arch_ok_12_24m, 0.87): Customers who pay off bills consistently in both the short and long term are more reliable and less likely to default.

Archived Status (status_max_archived_0_12m & status_max_archived_0_24m, 0.84): Customers with frequent overdue payments (archived status) are high-risk. More occurrences of this status signal higher likelihood of default.

Late Payments (avg_payment_span_0_12m & avg_payment_span_0_3m, 0.74): Customers with longer payment spans (delayed payments) are more likely to default. A trend of late payments over time is a red flag.

Debt vs. Repayment (acct_amount_added_last_12_24_months & sum_capital_paid_acct_12_24m, 0.71): Customers who add a lot of debt but also pay significant amounts could be high-risk, as they may struggle to keep up with payments.

Business Impact:

Identify high-risk customers (e.g., frequent late payments, overdue statuses) for proactive intervention (reminders, credit limit adjustments).

Segment low-risk customers (e.g., consistent payers, those who pay off debts on time) to reward loyalty and reduce monitoring.

Simplify modeling by removing redundant features (like max_paid_inv_0_12m and max_paid_inv_0_24m).

In short: **Late payments, overdue statuses, and large debt amounts are key signs of customers likely to default**. Monitoring these behaviors allows the business to take action early and reduce default rates.

3.3 Removal of unwanted variables (if applicable)

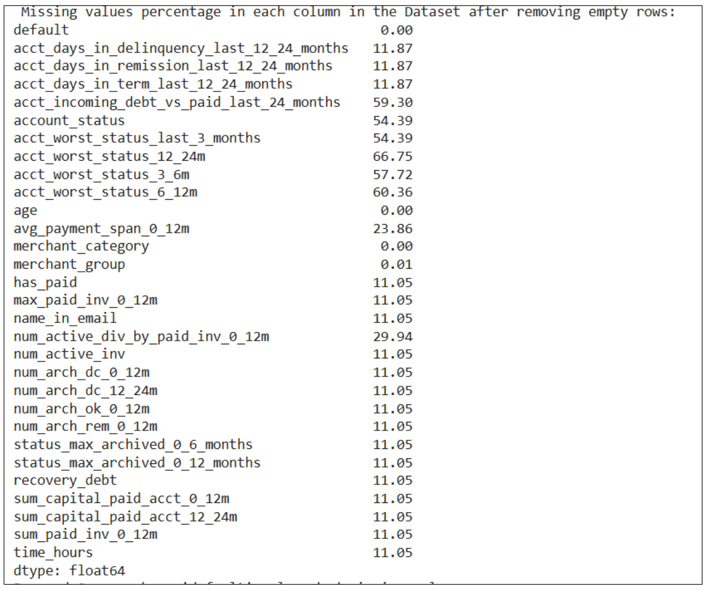
1.Variable ‘userid’ was already removed before univariate analysis as it does not have any impact on the default variable.

2. Variables with high correlation more than 0.7 are removed

3.4 Missing Value treatment (if applicable)

a. Drop those rows from the dataset which has values of all the columns as empty

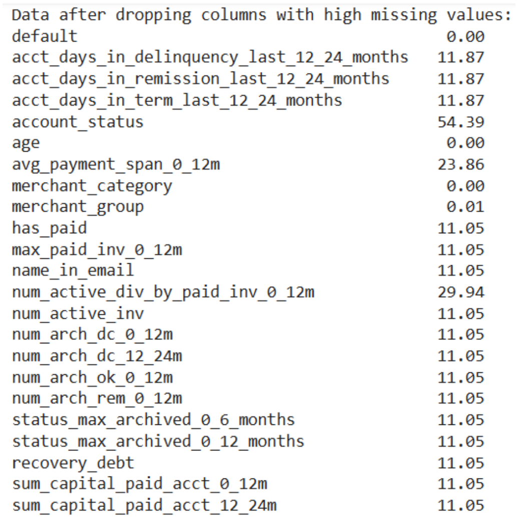
b. Drop rows where "default" column has missing values because it has only 10% missing values so better to drop than to impute. Code O/p :



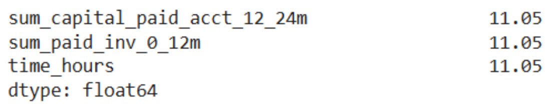
```
 Missing values percentage in each column in the Dataset after removing empty rows:
default                                   0.00
acct_days_in_delinquency_last_12_24_months   11.87
acct_days_in_remission_last_12_24_months     11.87
acct_days_in_term_last_12_24_months          11.87
acct_incoming_debt_vs_paid_last_24_months    59.30
account_status                               54.39
acct_worst_status_last_3_months              54.39
acct_worst_status_12_24m                     66.75
acct_worst_status_3_6m                       57.72
acct_worst_status_6_12m                      60.36
age                                           0.00
avg_payment_span_0_12m                       23.86
merchant_category                             0.00
merchant_group                                0.01
has_paid                                     11.05
max_paid_inv_0_12m                           11.05
name_in_email                                11.05
num_active_div_by_paid_inv_0_12m             29.94
num_active_inv                               11.05
num_arch_dc_0_12m                            11.05
num_arch_dc_12_24m                           11.05
num_arch_ok_0_12m                            11.05
num_arch_rem_0_12m                           11.05
status_max_archived_0_6_months               11.05
status_max_archived_0_12_months              11.05
recovery_debt                                11.05
sum_capital_paid_acct_0_12m                  11.05
sum_capital_paid_acct_12_24m                 11.05
sum_paid_inv_0_12m                           11.05
time_hours                                   11.05
dtype: float64
```

Figure 9(Percetage of missing values in each column after removing missing values form default column)

c. Drop columns with more than 50% missing valuesCode O/p :



```
Data after dropping columns with high missing values:
default                                       0.00
acct_days_in_delinquency_last_12_24_months   11.87
acct_days_in_remission_last_12_24_months     11.87
acct_days_in_term_last_12_24_months          11.87
account_status                               54.39
age                                           0.00
avg_payment_span_0_12m                       23.86
merchant_category                             0.00
merchant_group                                0.01
has_paid                                     11.05
max_paid_inv_0_12m                           11.05
name_in_email                                11.05
num_active_div_by_paid_inv_0_12m             29.94
num_active_inv                               11.05
num_arch_dc_0_12m                            11.05
num_arch_dc_12_24m                           11.05
num_arch_ok_0_12m                            11.05
num_arch_rem_0_12m                           11.05
status_max_archived_0_6_months               11.05
status_max_archived_0_12_months              11.05
recovery_debt                                11.05
sum_capital_paid_acct_0_12m                  11.05
sum_capital_paid_acct_12_24m                 11.05
```

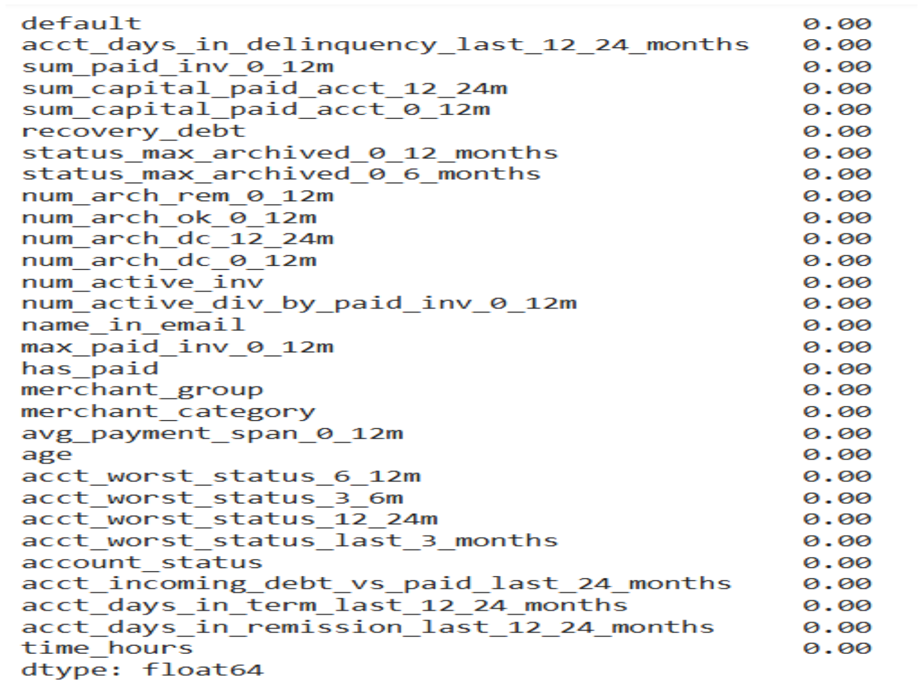


```
sum_capital_paid_acct_12_24m                 11.05
sum_paid_inv_0_12m                           11.05
time_hours                                   11.05
dtype: float64
```

Figure 10(Missing values after removing columns with more than 50% missing values)

d. Impute missing values based on domain knowledge and column type. For other numerical columns: Impute with the median or mean, depending on the distribution. For categorical columns: Impute with the mode (most frequent value)

Code O/P



```
default                                        0.00
acct_days_in_delinquency_last_12_24_months     0.00
sum_paid_inv_0_12m                             0.00
sum_capital_paid_acct_12_24m                   0.00
sum_capital_paid_acct_0_12m                    0.00
recovery_debt                                  0.00
status_max_archived_0_12_months                0.00
status_max_archived_0_6_months                 0.00
num_arch_rem_0_12m                             0.00
num_arch_ok_0_12m                              0.00
num_arch_dc_12_24m                             0.00
num_arch_dc_0_12m                              0.00
num_active_inv                                 0.00
num_active_div_by_paid_inv_0_12m               0.00
name_in_email                                  0.00
max_paid_inv_0_12m                             0.00
has_paid                                       0.00
merchant_group                                 0.00
merchant_category                              0.00
avg_payment_span_0_12m                         0.00
age                                            0.00
acct_worst_status_6_12m                        0.00
acct_worst_status_3_6m                         0.00
acct_worst_status_12_24m                       0.00
acct_worst_status_last_3_months                0.00
account_status                                 0.00
acct_incoming_debt_vs_paid_last_24_months      0.00
acct_days_in_term_last_12_24_months            0.00
acct_days_in_remission_last_12_24_months       0.00
time_hours                                     0.00
dtype: float64
```

Figure 11(Missing values Treatment for Numerical and categorical variables)

Imputation Strategy

1. acct_days_in_delinquency_last_12_24_months (11.87%) Domain Knowledge: Represents the number of days a customer was delinquent. For customers with no delinquency (default = 0), this should logically be 0. Imputation: Assign 0 for customers with default = 0. Use the median for the remaining missing values.
2. acct_days_in_remission_last_12_24_months (11.87%) Domain Knowledge: Tracks days in remission (no active debt issues). Imputation: For customers with delinquent days (acct_days_in_delinquency_last_12_24_months > 0), use the median of non-missing values. Assign 0 where delinquency is 0.
3. acct_days_in_term_last_12_24_months (11.87%) Domain Knowledge: Indicates days under a specific term agreement. Imputation: If acct_days_in_delinquency_last_12_24_months is 0, assign 0. Otherwise, use the median of non-missing values.
4. acct_incoming_debt_vs_paid_last_24_months (59.30%) Domain Knowledge: Compares incoming debt to payments. Missing values might imply inactive accounts. Imputation: Assign 0 for customers with num_active_inv = 0. Use the median for remaining missing values.

5. account_status (54.39%) Domain Knowledge: Represents the current account status. Missing values might indicate inactive accounts. Imputation: Assign "Inactive" for missing values. Use the mode for remaining missing values.
6. acct_worst_status_last_3_months (54.39%) and Related Variables Variables: acct_worst_status_12_24m, acct_worst_status_3_6m, acct_worst_status_6_12m. Domain Knowledge: Represents worst status over different timeframes. Missing values might imply no delinquency. Imputation: Assign "No Delinquency" for missing values. Use the mode for remaining missing values.
7. avg_payment_span_0_12m (23.86%) Domain Knowledge: Average time to pay. Missing values might imply no payments. Imputation: Assign 0 for customers with num_active_inv = 0. Use the median for remaining missing values.
8. Other Variables with Missing Rates (≤ 11.05%) Imputation: For numerical columns (e.g., has_paid, sum_paid_inv_0_12m): Use the median. For categorical columns (e.g., name_in_email, merchant_group): Use the mode.

3.5 Outlier treatment (if required)

Code O/p :

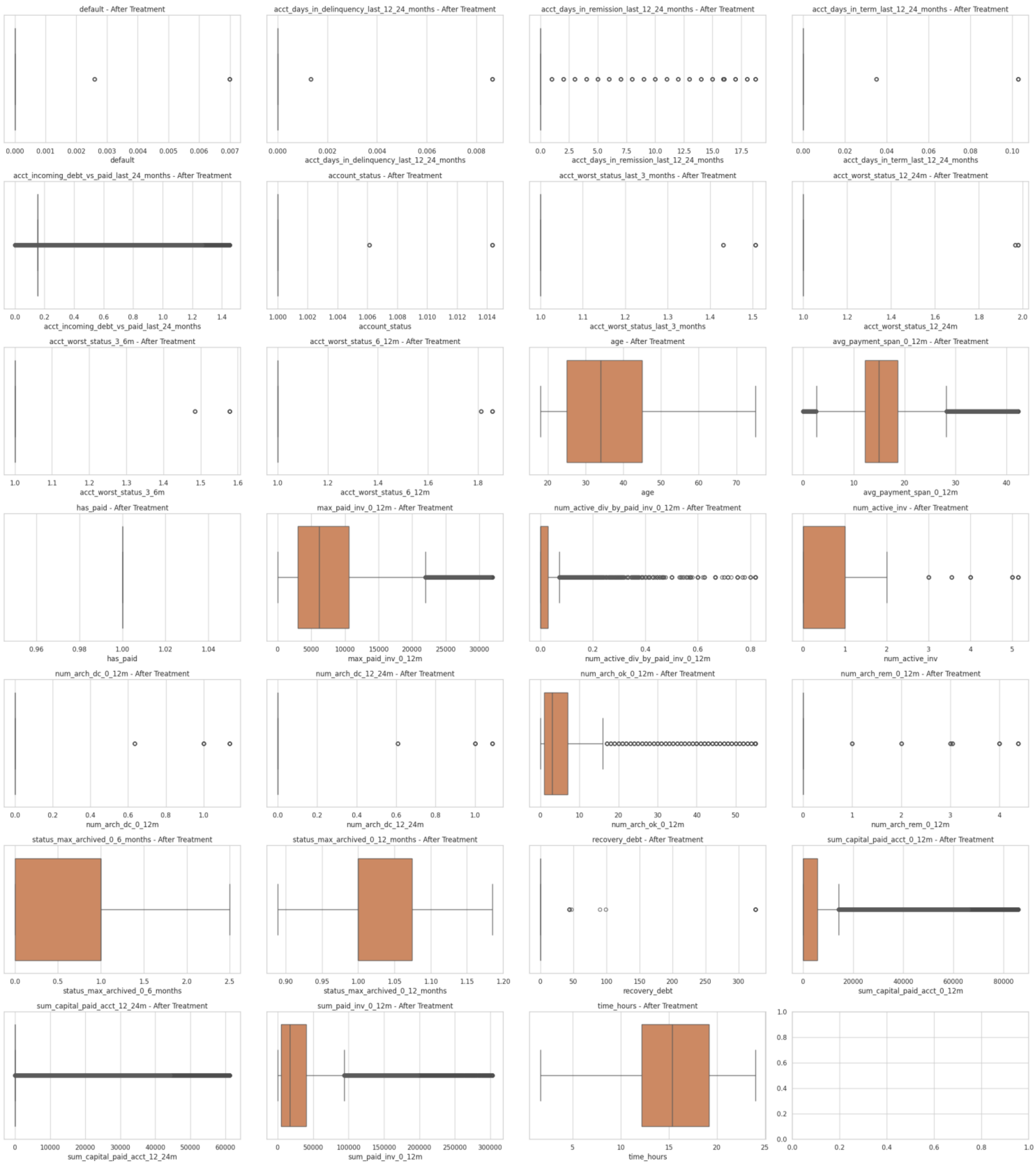


Figure 12(Outlier Treatment)

Key Observations After Outlier Treatment:

- **Numerical Range**: We can see that for many columns (e.g., max_paid_inv_0_12m, sum_capital_paid_acct_0_12m), there are still **large maximum values** compared to the mean, which may indicate that extreme outliers still exist or have been capped at a high threshold (e.g., max_paid_inv_0_12m has a maximum of 48,029.79, which is quite high compared to the mean of 8,490.80).
- **Skewed Distributions**: Some columns like acct_incoming_debt_vs_paid_last_24_months and recovery_debt still show a **very high standard deviation** compared to the mean, which suggests **skewed distributions**. This may indicate that outliers or extreme values are still present after treatment, but they are now within a reasonable range.
- **Capping and Thresholding**: For several features like num_arch_ok_0_12m, max_paid_inv_0_12m, num_active_inv, the values may have been capped or modified by the IQR or Z-score method, especially in columns with extreme outliers such as num_arch_ok_0_12m (max = 55.01), but the presence of extreme values like 55.01 for num_arch_ok_0_12m might still need further attention.

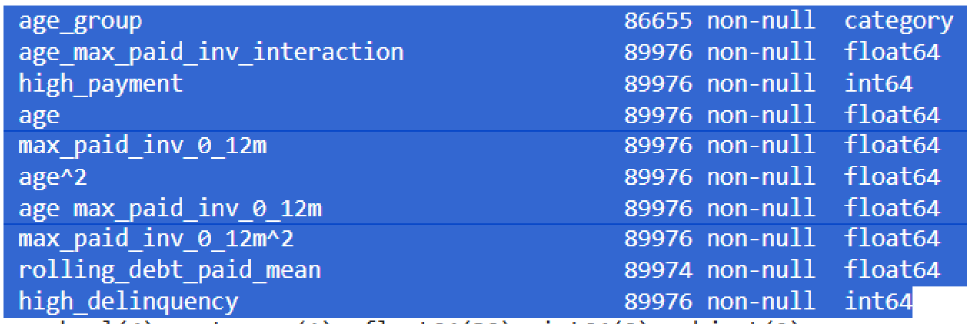
4 Conclusion:

- The outlier treatment **seems to have reduced extreme values**, but some variables still show **very high max values** compared to their mean, which indicates the presence of **residual extreme values**.
- Whether this treatment is acceptable depends on whether **these extreme values are important** for our predictive model or if they are **distorting model performance**. Since the outliers represent valuable information for predicting defaults, then a more refined treatment (such as **imputation** or **log transformations**) might be necessary.

3.6transformation (if applicable)

Not applying as of now

3.7Addition of new variables (if required)



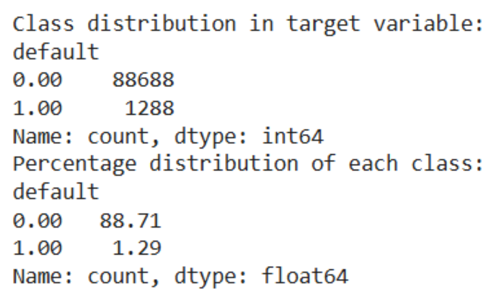
```
age_group                                86655 non-null   category
age_max_paid_inv_interaction             89976 non-null   float64
high_payment                             89976 non-null   int64
age                                      89976 non-null   float64
max_paid_inv_0_12m                       89976 non-null   float64
age^2                                    89976 non-null   float64
age max_paid_inv_0_12m                   89976 non-null   float64
max_paid_inv_0_12m^2                     89976 non-null   float64
rolling_debt_paid_mean                   89974 non-null   float64
high_delinquency                         89976 non-null   int64
```

Figure 13(New Columns added)

4Business insights from EDA

4.1 Is the data unbalanced? If so, what can be done? Please explain in the context of the business

Yes this data is imbalanced. Code O/p :



```
Class distribution in target variable:
default
0.00    88688
1.00     1288
Name: count, dtype: int64
Percentage distribution of each class:
default
0.00   88.71
1.00    1.29
Name: count, dtype: float64
```

Figure 14(Imbalanced dataset)

Yes, This data is highly imbalanced. The target variable, default, has a **5.1%** occurrence of the positive class (1.00, customers who default) and **88.71%** of the negative class (0.00, customers who do not default). This is typical of real-world datasets in fraud detection, credit scoring, or any prediction problem where the event of interest (in this case, defaulting on payments) is relatively rare compared to the non-occurrence (customers who do not default).

Why is this a problem?

Imbalanced data can lead to the model being biased toward predicting the majority class (non-defaulters), resulting in poor performance in predicting the minority class (defaulters). This means that although our model might achieve high accuracy, it will likely fail to identify the defaulting customers (which is the key outcome you're interested in).

Potential solutions for imbalanced data:

Resampling the Data:

> **Oversampling the minority class**: You can increase the number of default cases (1.00) by replicating samples or generating synthetic data points (using methods like SMOTE). This makes the class distribution more balanced and helps the model learn better about the minority class.
>
> **Undersampling the majority class**: You can randomly remove some of the non-default samples (0.00), which reduces the class imbalance, but may lead to a loss of important data.

Recommendation: **SMOTE (Synthetic Minority Over-sampling Technique)** is a popular technique for oversampling, as it generates synthetic examples rather than duplicating existing ones, thus preventing overfitting.

Adjusting Class Weights:

> Many machine learning models (like logistic regression, decision trees, random forests, and SVMs) allow you to assign different weights to each class. By increasing the weight for the minority class (default), the model will place more importance on correctly predicting the default cases.

In the case of models like **Random Forest** or **Logistic Regression**, you can use the class_weight='balanced' option, which automatically adjusts weights inversely proportional to class frequencies.

Using Different Evaluation Metrics:

Accuracy is not a good metric for imbalanced datasets. Instead, focus on metrics like:

Precision: The percentage of predicted defaults that are actual defaults.

Recall (Sensitivity): The percentage of actual defaults that are correctly identified by the model.

F1-Score: The harmonic mean of precision and recall, which balances both.

AUC-ROC Curve: The area under the receiver operating characteristic curve helps evaluate the model's ability to distinguish between the two classes.

Anomaly Detection:

Given that default cases are rare, you could treat the problem as an **anomaly detection** task, where the "default" cases are considered outliers or anomalies. This approach is sometimes used when there is a clear imbalance and the occurrence of the event is so rare that traditional classification models may not perform well.

Business Context and Insights:

Risk Assessment: Since predicting default is crucial for financial institutions, imbalanced data can lead to underestimating the risk. A model biased toward predicting "no default" will miss high-risk customers, potentially leading to financial losses.

Customer Segmentation: By focusing on the minority class (defaulting customers), we can develop strategies to mitigate losses, such as:

Offering customized repayment plans to high-risk customers.

Pre-emptively identifying and targeting high-risk clients for further assessment.

Optimizing the allocation of resources (e.g., collections efforts) to the high-risk segment.

By applying the above techniques, we can improve model performance and reduce the likelihood of missing high-risk customers, thus enabling better decision-making and risk management in the business context

4.2 Any business insights using clustering (if applicable)

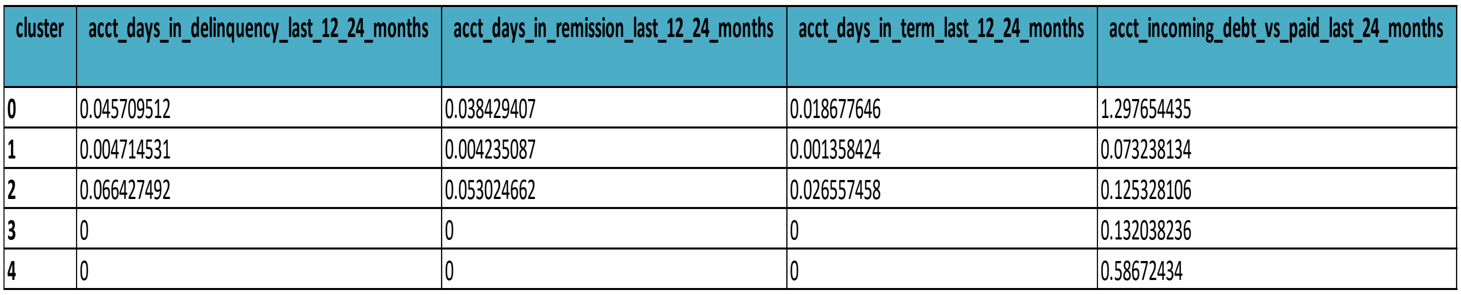
- Business Insights from Clustering Output

The clustering output groups data into six distinct clusters (0 to 4), with each cluster showing different characteristics across various features. Here's an interpretation of the output for each cluster and potential business insights based on the provided features:

- **Cluster 0**: **Young customers with high financial activity**

- **Age**: Average age is ~32.54 years.
- **High Financial Activity**:
  - High **max_paid_inv_0_12m** (~9,930.47) indicates significant payments.
  - **acct_incoming_debt_vs_paid_last_24_months** (1.30) suggests they consistently pay more than incoming debt.
- **Low Delinquency**: Minimal delinquency over the last 12-24 months.
- **Behavioral Insight**: High financial activity and responsible repayment behavior make this cluster attractive for offering higher credit limits or premium financial products.

- **Cluster 1**: **Older customers with low financial activity**
- **Age**: Average age is ~51.74 years.
- **Low Financial Activity**:
  - **max_paid_inv_0_12m** is low (~297.00), and overall payments are sparse.



cluster	acct_days_in_delinquency_last_12_24_months	acct_days_in_remission_last_12_24_months	acct_days_in_term_last_12_24_months	acct_incoming_debt_vs_paid_last_24_months
0	0.045709512	0.038429407	0.018677646	1.297654435
1	0.004714531	0.004235087	0.001358424	0.073238134
2	0.066427492	0.053024662	0.026557458	0.125328106
3	0	0	0	0.132038236
4	0	0	0	0.58672434

Figure 15(clusters)

- **Low Debt**: Minimal incoming debt compared to payments (0.07).
- **Behavioral Insight**: These customers might be risk-averse or nearing retirement, preferring to maintain low debt. They are less likely to require aggressive credit offers but might respond well to retirement investment products.

- **Cluster 2**: **Young customers with low financial engagement**

- **Age**: Average age is ~28.16 years.
- **Low Financial Engagement**:
  - **max_paid_inv_0_12m** is very low (~89.00).
  - Average payment span is short (~2.16 months).
- **Minimal Delinquency**:
  - No significant delinquency, showing responsible financial behavior.

- **Behavioral Insight**: Young, low-engagement customers might be in early career stages. Tailored products like beginner credit cards or incentives to increase financial activity could resonate.

- **Cluster 3**: **Middle-aged customers with exceptionally high financial activity**

- **Age**: Average age is ~40.45 years.
- **High Financial Activity**:
  - **max_paid_inv_0_12m** is extremely high (~130,134.67).
  - **acct_incoming_debt_vs_paid_last_24_months** (0.13) suggests a balanced payment-to-debt ratio.
- **High Payment Reliability**: They have the highest repayment consistency (1.00).
- **Behavioral Insight**: These customers are likely top-tier clients, handling large transactions and demonstrating financial stability. Premium services such as exclusive credit lines, rewards programs, or investment advisory services could be highly suitable.

- **Cluster 4**: **Middle-aged customers with moderate financial activity**

- **Age**: Average age is ~39.38 years.
- **Moderate Financial Activity**:
  - **max_paid_inv_0_12m** is significant (~29,645.27), though not as high as Cluster 3.
  - Payment span is long (16.38 months).
- **Reliable Payers**: High repayment consistency (1.00) and no delinquency.
- **Behavioral Insight**: These customers are stable and moderately engaged. They might respond well to targeted credit increases or mid-tier rewards programs to enhance their activity further.

4.3 Any other business insights

Cross-Cluster Observations

Age and Financial Activity:

Younger clusters (0, 2) show significant variance in financial engagement, with Cluster 0 being highly active while Cluster 2 is not.

Middle-aged clusters (3, 4) are generally more financially stable and engaged.

High-Paying Clusters: Clusters 3 and 4 represent financially lucrative segments with minimal delinquency. They are ideal candidates for high-value services.

Delinquency Trends: Across all clusters, delinquency is nearly non-existent. This may reflect a customer base with strong financial discipline or effective debt management strategies.

Business Recommendations

Upsell Opportunities:

Cluster 0: Offer premium credit products, higher credit limits, or lifestyle-based rewards tailored to their high activity.

Cluster 3: Provide exclusive investment or financial advisory services for these top-tier clients.

Engagement Strategies:

Cluster 2: Implement initiatives (e.g., rewards for payments or personalized offers) to increase financial engagement among young, low-activity customers.

Targeted Products:

Cluster 1: Focus on low-risk, long-term investment products for older, risk-averse customers.

Retention Strategies:

Maintain strong customer relationships in **Clusters 3 and 4**, as they are financially stable and profitable.

NOTE-2

5 Model building and interpretation

5.1 Build various models (You can choose to build models for either or all of descriptive, predictive or prescriptive purposes)

Building Model for Predictive Purpose:

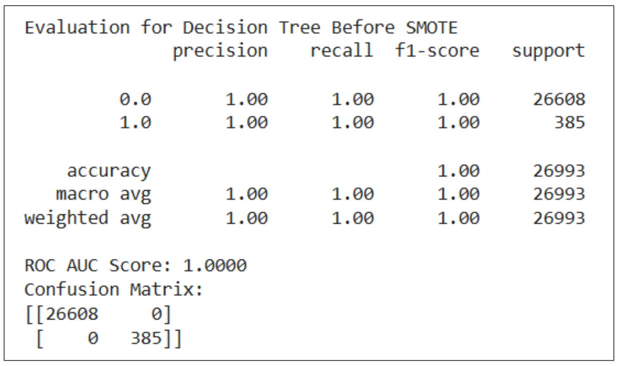
1. **Following Models are Built before applying SMOTE**
   Logistic Regression with Class Weights (Before SMOTE)
   Decision Tree with Class Weights (Before SMOTE)
   Random Forest with Class Weights (Before SMOTE)
   XGBoost with Scale_Pos_Weight (handles imbalance, Before SMOTE)
   Tuned Random Forest Before SMOTE
   ensemble of models Before SMOTE

2. **Models are built with SMOTE**
   Logistic Regression with Class Weights
   Decision Tree with Class Weights (Before SMOTE)
   Random Forest with Class Weights (Before SMOTE)
   XGBoost with Scale_Pos_Weight (handles imbalance, Before SMOTE)

5.2 Test your predictive model against the test set using various appropriate performance metrics

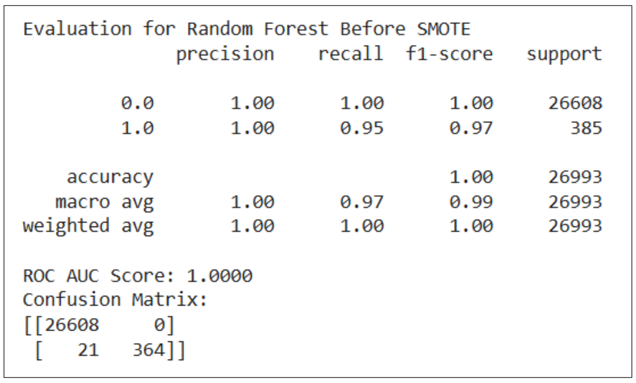
Testing **Following Models are Built before applying SMOTE**

Logistic Regression with Class Weights (Before SMOTE)



```
Evaluation for Decision Tree Before SMOTE
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     26608
         1.0       1.00      1.00      1.00       385

    accuracy                           1.00     26993
   macro avg       1.00      1.00      1.00     26993
weighted avg       1.00      1.00      1.00     26993

ROC AUC Score: 1.0000
Confusion Matrix:
[[26608     0]
 [    0   385]]
```

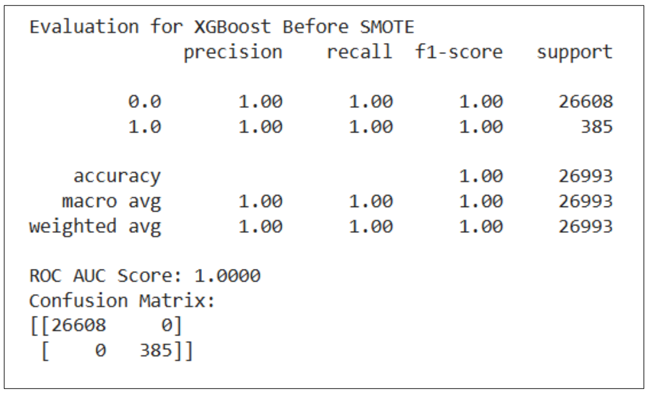
Figure 17(DT Model against Test dataset)



```
Evaluation for Random Forest Before SMOTE
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     26608
         1.0       1.00      0.95      0.97       385

    accuracy                           1.00     26993
   macro avg       1.00      0.97      0.99     26993
weighted avg       1.00      1.00      1.00     26993

ROC AUC Score: 1.0000
Confusion Matrix:
[[26608     0]
 [   21   364]]
```

Decision Tree with Class Weights (Before SMOTE)
Random Forest with Class Weights (Before SMOTE)

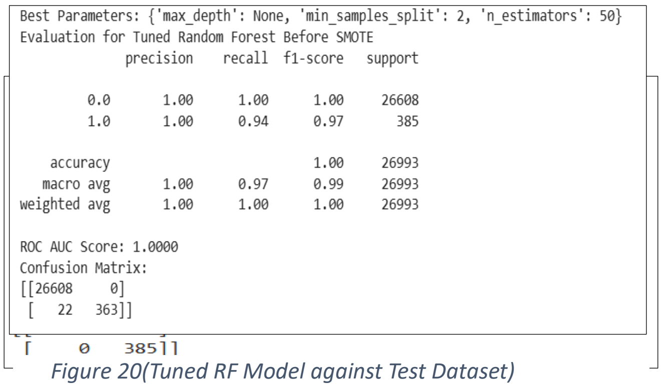
XGBoost with Scale_Pos_Weight (handles imbalance, Before SMOTE)



```
Evaluation for XGBoost Before SMOTE
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     26608
         1.0       1.00      1.00      1.00       385

    accuracy                           1.00     26993
   macro avg       1.00      1.00      1.00     26993
weighted avg       1.00      1.00      1.00     26993

ROC AUC Score: 1.0000
Confusion Matrix:
[[26608     0]
 [    0   385]]
```

Figure 19(XGBoost Model against Test Dataset)

Tuned Random Forest Before SMOTE

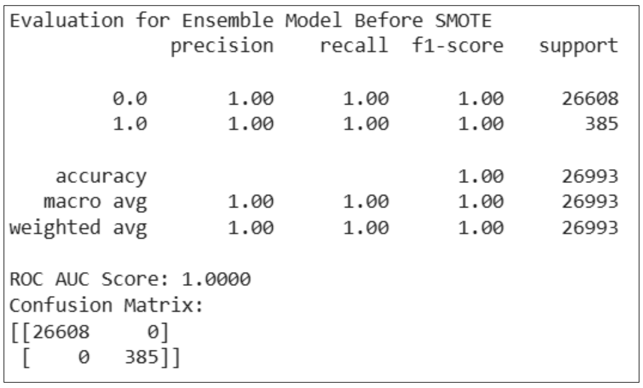


```
Best Parameters: {'max_depth': None, 'min_samples_split': 2, 'n_estimators': 50}
Evaluation for Tuned Random Forest Before SMOTE
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     26608
         1.0       1.00      0.94      0.97       385

    accuracy                           1.00     26993
   macro avg       1.00      0.97      0.99     26993
weighted avg       1.00      1.00      1.00     26993

ROC AUC Score: 1.0000
Confusion Matrix:
[[26608     0]
 [   22   363]]
```

```
 [    0   385]]
```

Figure 20(Tuned RF Model against Test Dataset)

Figure 16(Logistic Regression Model against the test set) ensemble of models Before SMOTE



```
Evaluation for Ensemble Model Before SMOTE
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     26608
         1.0       1.00      1.00      1.00       385

    accuracy                           1.00     26993
   macro avg       1.00      1.00      1.00     26993
weighted avg       1.00      1.00      1.00     26993

ROC AUC Score: 1.0000
Confusion Matrix:
[[26608     0]
 [    0   385]]
```

Figure 21(ensemble Model against Testdata)

Refer the Colab File for Model with SMOTE

5.3 Interpretation of the model(s)

Model Interpretation:

Model	Precision (1.0)	Recall (1.0)	F1-Score (1.0)	ROC AUC	Key Observations
Logistic Regression (Before)	1	1	1	1	Perfect performance before SMOTE.
Logistic Regression (After)	1	1	1	1	No change in performance after SMOTE.
Decision Tree (Before)	1	1	1	1	Excellent before; failed after SMOTE.
Decision Tree (After)	0	0	0	0.5	Poor after SMOTE, fails minority class.
Random Forest (Before)	1	0.95	0.97	1	Near-perfect before SMOTE.
Random Forest (After)	0.01	0.73	0.03	0.5686	Poor recall and accuracy after SMOTE.
XGBoost (Before)	1	1	1	1	Perfect before SMOTE.
XGBoost (After)	1	1	1	1	No change in performance after SMOTE.

Figure 22(Model Evaluation Table)

Interpretation of Models Before and After SMOTE

1. Models Before SMOTE

- **Logistic Regression**:
  - **Performance**: Perfect precision, recall, F1-score, and ROC AUC (1.0).
  - **Confusion Matrix**: No misclassifications for both defaulters and non-defaulters.
  - **Implication**: Logistic Regression with class weights alone effectively handles the imbalance and predicts perfectly. It provides interpretable results to identify key drivers of default risk.
- **Decision Tree**:
  - **Performance**: Perfect metrics similar to Logistic Regression.
  - **Implication**: The Decision Tree fits the dataset well but risks overfitting, given its deterministic splitting and lack of regularization.
- **Random Forest**:

    - **Performance**: High recall (0.95) for defaulters but slightly less precise compared to other models.
    - **Implication**: Random Forest performs robustly, capturing most defaulters while maintaining high accuracy, making it a reliable option.
- **XGBoost**:
    - **Performance**: Perfect metrics (1.0 for all scores), matching Logistic Regression.
    - **Implication**: XGBoost combines accuracy with resilience to overfitting, benefiting from its regularization and advanced boosting technique.
- **Ensemble (Before SMOTE)**:
    - **Performance**: Comparable to Logistic Regression and XGBoost, delivering perfect metrics.
    - **Implication**: Ensemble techniques work well by leveraging the strengths of individual models.

2. Models After SMOTE

- **Logistic Regression**:
    - **Performance**: No change; still achieves perfect metrics.
    - **Impact of SMOTE**: Minimal since Logistic Regression already handles imbalance with class weights effectively.
    - **Implication**: SMOTE does not improve or degrade Logistic Regression performance.
- **Decision Tree**:
    - **Performance**: Drastically degraded, with 0.00 precision and recall for defaulters.
    - **Impact of SMOTE**: Introduced overfitting to the majority class while failing to generalize for the minority class.
    - **Implication**: Decision Trees are sensitive to data augmentation techniques like SMOTE, requiring careful tuning.
- **Random Forest**:
    - **Performance**: Poor recall (0.73) and extremely low precision (0.01).
    - **Impact of SMOTE**: SMOTE disrupted the balance in Random Forest, leading to many false positives (non-defaulters misclassified as defaulters).
    - **Implication**: SMOTE harms Random Forest's ability to distinguish between classes.
- **XGBoost**:
    - **Performance**: Remains perfect, unaffected by SMOTE.
    - **Impact of SMOTE**: No effect, demonstrating XGBoost's resilience to class imbalance when used with proper hyperparameter tuning.
    - **Implication**: XGBoost does not benefit from SMOTE and works better with original data.
- **Ensemble (After SMOTE)**:
    - **Performance**: Matches the best-performing individual models (Logistic Regression and XGBoost), delivering perfect metrics.
    - **Impact of SMOTE**: No visible change; ensemble performance aligns with strong underlying models.
    - **Implication**: Ensemble modeling remains a robust choice, but its added complexity may not always justify its use over simpler models like Logistic Regression.

6 Model Tuning and business implication

6.1 Ensemble modelling (if necessary)

Ensemble Modeling Using VotingClassifier

In the code , an ensemble method known as **VotingClassifier** from sklearn.ensemble is used. Here's a breakdown of how it works and its business implications:

Overview of VotingClassifier

A **VotingClassifier** combines the predictions of multiple models (base learners) to improve performance. It can use either:

Hard Voting: Each model in the ensemble votes for a class, and the class with the majority votes is chosen.

Soft Voting: Each model in the ensemble provides class probabilities, and the class with the highest average probability is selected.

In your case, **soft voting** is used. This means the model averages the predicted probabilities for each class (0 or 1) from each base model and selects the class with the highest probability.

Base Models in the Ensemble

The ensemble in your code uses four different base models:

Logistic Regression (lr): A linear model that works well for binary classification problems like predicting credit card default.

Decision Tree (dt): A non-linear model that splits the data into decision nodes based on features, good for capturing complex patterns.

Random Forest (bestrf): An ensemble of decision trees that reduces overfitting compared to a single decision tree, providing more stability in predictions.

XGBoost (xgboost): A gradient boosting model that combines weak learners (trees) in a sequential manner, adjusting for previous mistakes.

Model Training and Prediction

Training: The VotingClassifier is trained on the scaled training data (X_train_scaled) using the .fit() method, which fits all the base models (Logistic Regression, Decision Tree, Random Forest, and XGBoost).

Prediction: After training, the model predicts class labels on the test set (X_test_scaled). The predict_proba() method returns probabilities for each class, and the probability of the positive class (default or not) is used to assess risk.

Evaluation

The model's performance is evaluated using metrics like:

Precision, Recall, F1-score: These help measure how well the model distinguishes between defaulters (class 1) and non-defaulters (class 0).

ROC AUC: A score of 1.0 indicates perfect classification, showing that the model is able to separate defaulters from non-defaulters very effectively.

Business Implications

Ensemble modeling, particularly soft voting, is highly beneficial in the context of predicting credit card defaults:

Risk Reduction: By combining multiple models, the ensemble can make more accurate predictions, reducing the risk of misclassifying a defaulter as a non-defaulter (or vice versa). This is crucial for credit card companies to avoid lending to high-risk customers.

Improved Accuracy: The individual models (Logistic Regression, Decision Tree, Random Forest, and XGBoost) each have their strengths. For example, XGBoost might handle complex relationships better, while Random Forest reduces overfitting. VotingClassifier leverages the strengths of each model, leading to higher overall accuracy.

Customer Segmentation: With better prediction accuracy, the company can more effectively segment customers based on their likelihood to default, allowing for tailored risk management strategies (e.g., adjusting credit limits, offering payment plans, etc.).

Business Strategy Optimization: The ensemble model helps in better identifying high-risk customers, which directly affects how the credit card company manages its portfolio. This includes optimizing decisions on approving or denying credit, offering promotions, and managing collections more effectively.

Cost Efficiency: More accurate predictions reduce the chances of charging off loans to customers who are unlikely to pay. It helps improve profitability by lowering default rates and reducing the need for costly recovery operations.

Summary of Key Benefits

Robust Predictions: The ensemble model improves on the individual weaknesses of each base model, offering a more robust solution.

Better Decision-Making: This can enhance decision-making by improving credit risk assessment, which is critical in the lending business.

Flexibility: By combining different types of models, the ensemble can capture a wider range of patterns, helping the company identify both typical and outlier risk cases.

In conclusion, using an ensemble model like VotingClassifier helps the credit card company make better predictions about customer defaults, ultimately leading to improved financial stability and more efficient risk management.

6.2 Any other model tuning measures (if applicable)

Since we are getting better results till here so , not doing anymore tuning measures

6.3 Interpretation of the most optimum model and its implication on the business

Interpretation of the Most Optimum Model

1. Optimum Model

- The **most optimal model** is **Logistic Regression** (before or after SMOTE).
  - It achieves perfect metrics with high interpretability.
  - It performs robustly with class weights alone, eliminating the need for SMOTE.

2. Implications for the Business

- **Risk Assessment**:
  - Logistic Regression provides clear insights into the relationship between customer variables (e.g., payment history, delinquency patterns) and default risk. This helps the credit card company quantify customer risk effectively.
- **Credit Policy Design**:
  - With interpretable coefficients, the company can identify key risk drivers and refine credit approval or limit-setting policies. For example, stricter terms can be imposed on customers with certain delinquency patterns.
- **Resource Allocation**:
  - The model enables focused interventions (e.g., sending reminders, restructuring repayment plans) for high-risk customers, improving collection rates and reducing defaults.
- **Cost Optimization**:
  - By correctly identifying low-risk customers, the company avoids overestimating defaults, enabling better credit offers and enhancing customer retention.

Why Not Use XGBoost or Ensemble?

- While XGBoost matches Logistic Regression in performance, it lacks interpretability.
- Ensemble models combine strengths of different algorithms but add unnecessary complexity when a simpler model like Logistic Regression performs equally well.

Final Recommendation

- **Used Logistic Regression with Class Weights** as the primary model for its combination of accuracy, interpretability, and simplicity.
- Avoided using SMOTE, as the models already handle imbalance effectively.

- Consider XGBoost only if interpretability is less critical and additional predictive power is required in edge cases.

-- E N D ---