Dataset Link :
https://docs.google.com/spreadsheets/d/1TtUzxyGc5XUyD2dOVvsKXLvRTlZwtYAAJoPnY69sMdM/edit?gid=0#gid=0

Project Title

Customer Risk & Banking Behaviour Analysis using Statistical Methods

Problem Statement

A bank wants to understand customer behavior, credit risk, and demographic patterns using real-world, messy data.
You are assigned to clean the dataset, perform statistical analysis, visualize insights, and prepare the data for modeling.

Dataset Details (1000 rows, 15 columns)

Columns included:

1. Customer_ID

- **Type:** Numerical (Discrete)
- **Description:** Unique identifier assigned to each customer.
- **Notes:** Used for record tracking; no analytical meaning.

---

2. Name

- **Type:** Categorical (Nominal)
- **Description:** Full name of the customer.
- **Notes:** May include inconsistent formats (upper/lowercase, extra spaces).

---

3. Age

- **Type:** Numerical (Continuous)
- **Description:** Age of the customer in years.
- **Notes:** May contain missing values or unrealistic ages (e.g., <18 or >90).

---

4. Gender

- **Type:** Categorical (Nominal)
- **Description:** Gender of the customer (Male/Female).
- **Notes:** Includes inconsistent entries (e.g., “M”, “male”, “FEMALE”, “f”).

---

5. Account_Type

- **Type:** Categorical (Nominal)
- **Description:** Type of bank account held (Savings, Current, Salary, Joint).
- **Notes:** Contains typos and mixed-format values.

---

6. Account_Balance

- **Type:** Numerical (Continuous)
- **Description:** Current available balance in the customer's account (INR).
- **Notes:** Contains outliers and missing values.

---

7. Loan_Amount

- **Type:** Numerical (Continuous)
- **Description:** Total sanctioned loan amount for the customer.
- **Notes:** May include extreme values and incorrect entries.

---

8. Loan_Status

- **Type:** Categorical (Nominal)
- **Description:** Status of the customer's loan (Approved / Rejected / Pending).
- **Notes:** Includes inconsistent labels ("APPROVED", "approved", "A", etc.).

---

9. Credit_Score

- **Type:** Numerical (Continuous)
- **Description:** Credit rating of the customer (300–900 scale).
- **Notes:** Outliers possible; useful for credit risk analysis.

---

10. Monthly_Income

- **Type:** Numerical (Continuous)
- **Description:** Customer's monthly income in INR.
- **Notes:** Contains missing values and noise.

---

11. Employment_Status

- **Type:** Categorical (Nominal)
- **Description:** Type of employment (Salaried, Self-Employed, Unemployed, Student).
- **Notes:** May contain typos and inconsistent capitalization.

---

12. Transaction_Count

- **Type:** Numerical (Discrete)
- **Description:** Number of monthly bank transactions made by the customer.
- **Notes:** Missing values possible; used in behavior-based segmentation.

---

13. Phone

- **Type:** Categorical (Nominal)
- **Description:** Customer’s phone number.
- **Notes:** Contains formatting inconsistencies, missing digits, symbols.

---

14. City

- **Type:** Categorical (Nominal)
- **Description:** Customer’s city of residence.
- **Notes:** Contains NULLs and inconsistent spellings (e.g., “Mumbai”, “mumbai”, “Bombay”).

---

15. Late_Payments

- **Type:** Numerical (Discrete)
- **Description:** Number of late EMI/credit-card payments recorded.
- **Notes:** May include extremely high values due to input errors.

Includes **missing values, outliers, inconsistent formats, categorical noise, typos, mixed data types**.

Data Types Classification

Column	**Type**	**Measurement Type**
Customer_ID	Numerical	Discrete
Name	Categorical	Nominal

Age	Numerical	Continuous
Gender	Categorical	Nominal
Account_Type	Categorical	Nominal
Account_Balance	Numerical	Continuous
Loan_Amount	Numerical	Continuous
Loan_Status	Categorical	Nominal
Credit_Score	Numerical	Continuous
Monthly_Income	Numerical	Continuous
Employment_Status	Categorical	Nominal
Transaction_Count	Numerical	Discrete
Phone	Categorical	Nominal
City	Categorical	Nominal
Late_Payments	Numerical	Discrete

Exploratory Data Analysis (EDA)

✔ Descriptive Statistics

Use df.describe() (already generated in the file).
Includes:

- Mean
- Median
- Standard Deviation
- Min/Max
- Percentiles

Percentiles, Quartiles & IQR

Percentiles

df['Account_Balance'].quantile([0.25,0.5,0.75,0.9,0.95])

Quartiles

- Q1 = 25th percentile
- Q2 = Median (50%)
- Q3 = 75th percentile

IQR

IQR = Q3 - Q1

Missing Values Handling

Identify missing values

df.isnull().sum()

Handling Strategy

Column	**Method**
Age	Median Imputation
Credit Score	Mean
Account Balance	Median
Monthly Income	Regression / Mean
Phone	Remove/Flag

Outlier Detection & Treatment

Z-Score Method

from scipy import stats

z = np.abs(stats.zscore(df['Account_Balance'].dropna()))

IQR Method

upper = Q3 + 1.5 * IQR

lower = Q1 - 1.5 * IQR

Handling:

- Cap outliers (winsorization)
- Replace with percentile values

1. Winsorization (Cap Outliers)

Meaning:
You **do NOT remove** the outliers.
You **cap (limit)** the extreme values to a certain percentile.

Example:

Let's say Account_Balance has extreme outliers.

- 1st percentile = ₹5,000
- 99th percentile = ₹1,50,000

Now apply winsorization:

- Any value **below 1st percentile** → replace with **₹5,000**
- Any value **above 99th percentile** → replace with **₹1,50,000**

Python Example:

```
lower = df['Account_Balance'].quantile(0.01)

upper = df['Account_Balance'].quantile(0.99)


df['Account_Balance'] = df['Account_Balance'].clip(lower, upper)
```

You keep the row
You just adjust extreme values
Useful to reduce noise without losing data

---

2. Replace Outliers with Percentile Values

This method **replaces the outlier with a percentile number instead of capping**.

Example:

Use the **median (50th percentile)** or **75th percentile** instead.

Let's say outlier value = ₹10,00,000
50th percentile (median) = ₹48,000

Replace:

- Outlier → 48,000

Code Example (IQR method to detect + replace):

```
Q1 = df['Account_Balance'].quantile(0.25)

Q3 = df['Account_Balance'].quantile(0.75)

IQR = Q3 - Q1


lower = Q1 - 1.5 * IQR

upper = Q3 + 1.5 * IQR


median_value = df['Account_Balance'].median()


df.loc[(df['Account_Balance'] < lower) | (df['Account_Balance'] > upper),

    'Account_Balance'] = median_value
```

Instead of deleting
You put a stable statistical value
Good for skewed distributions

Simple Difference

Method	What Happens?	Good For
Winsorization (Cap)	Values are clipped to a threshold	Normal-like distributions
Replace with Percentile	Outlier becomes median/percentile	Skewed data, financial data

Distribution Analysis

Histogram

- Account Balance
- Credit Score
- Monthly Income

Normal Distribution Curve

Use:

```
sns.distplot(df['Credit_Score'], fit=norm)
```

Skewness & Kurtosis

Skewness

```
df['Account_Balance'].skew()
```

Interpretation:

- 0 = perfect symmetric
- 0 = right skew (long tail right)
- <0 = left skew

Kurtosis

```
df['Account_Balance'].kurt()
```

Interpretation:

- 3 = leptokurtic (heavy tails)
- 3 = mesokurtic (normal)
- <3 = platykurtic (light tails)

Visualizations (All Graph Types)

📊 **Univariate**

- Histogram
- Boxplot
- Density Plot
- Barplot

📉 **Bivariate**

- Scatter plot (Income vs Credit Score)
- Heatmap correlation
- Boxplot (Loan Status vs Balance)

📈 **Time/Trend (if needed)**

- Transaction count distribution

Final Insights You Can Include in Report

✓ Customers with low credit score tend to have more late payments
✓ Certain cities have higher average loan approvals
✓ Account balance is heavily right-skewed due to outliers
✓ Missing values were mainly in numerical columns
✓ Strong relationship between Monthly Income and Account Balance
✓ Outliers were capped to stabilize analysis

Deliverables for Your Project

1. Cleaned Dataset

After imputation & outlier handling.

2. Statistical Analysis Report

PDF report explaining all metrics.

3. Visualization Dashboard

Using Power BI / Tableau / Python.

4. Jupyter Notebook

Complete EDA + Statistical Methods.

Project Explanation

What I did — step by step

1. **Loaded** the messy dataset you have (bank_messy_dataset.csv).
2. **Standardized** categorical labels (gender, account type, city, loan status).
3. **Converted** numeric-like columns (Account_Balance, Credit_Score, Monthly_Income, Loan_Amount) to numeric types
4. **Imputed missing values**:
   - Age → median
   - Account_Balance → median
   - Credit_Score → mean
   - Monthly_Income → median
     (These choices are often pragmatic — median for skewed money values, mean for scores when distribution is close to normal.)
5. **Detected outliers** using the **IQR method** (per-column). I saved the counts and lower/upper thresholds per column.
6. **Created two treatments**:
   - **Winsorized** dataset: clipped each numeric column to the 1st and 99th percentiles (keeps extreme rows but limits their effect).
   - **Median-replaced** dataset: replaced IQR-identified outliers with the column median.
7. **Computed skewness & kurtosis** (for Account_Balance, Credit_Score, Monthly_Income, Transaction_Count, Late_Payments) for the original, imputed, winsorized and median-replaced datasets to compare how distributions changed.
8. **Saved descriptive stats** (before/after) and correlation matrix.
9. **Plotted histograms** (original vs winsorized vs median-replaced) for key numeric columns and saved PNGs.

Key numeric findings (high level)

- Many numeric columns were skewed (Account_Balance and Monthly_Income especially).
- IQR method flagged several outliers in Account_Balance and Credit_Score (counts written in quick_report.txt).
- **Winsorization** reduced extreme tails and moved skewness closer to zero in many columns.
- **Median-replacement** tends to shrink variability more (reduces std dev) and can change median less than winsorization in some cases.
- Correlations (see numeric_corr_matrix.csv) can change slightly after outlier handling — expected because extremes influence correlation.