Bank Loan Data Preprocessing & Outlier Detection

Problem Title:

"Bank Loan Application Data Preprocessing and Outlier Detection"

NAME: Prerana M

EMAIL: preranam8792@gmail.com

DATE: 2nd Sep 2025

Description:

This project focuses on cleaning, preprocessing, and analyzing a bank loan dataset using Python and Pandas. The goal is to prepare raw loan application data for further modeling by handling missing values, detecting duplicates, standardizing categorical variables, and identifying outliers using the IQR method. The project also detects potentially high-risk loans and allows downloading the cleaned dataset for further analysis or machine learning tasks.

Index:

1. Introduction

2. Problem Statement

3. Objectives

4. Dataset Description

5. Use-Case Explanation

6. UML Diagrams

7. Code Implementation & Explanation

8. Output Screenshots with explanation

9. Conclusion

10. Bibliography

1.Introduction:

Data preprocessing is one of the most important steps in any data science or machine learning pipeline. Before applying algorithms, the dataset must be cleaned and prepared to remove missing values, inconsistencies, and outliers. This project focuses on preprocessing a **Bank Loan Application Dataset** to ensure that the data is reliable and ready for further analysis or modeling.

2.Problem Statement:

The project focuses on cleaning, preprocessing, and analyzing a loan application dataset to detect inconsistencies, missing values, and outliers. The final goal is to prepare the dataset for further modeling.

3. Objectives:

- To inspect and clean the dataset by handling missing values and duplicates.

- To standardize categorical values for consistency.

- To detect and highlight outliers using the **IQR (Interquartile Range) method**.

- To filter high-risk records where **LoanAmount > 2 × Income**.

- To group and aggregate data for insights (e.g., average LoanAmount by EmploymentStatus).

- To export a cleaned dataset for future modeling.

4. Dataset Description:

- **ApplicationID**: Unique identifier for each application.

- **Income**: Applicant's monthly income.

- **CreditScore**: Creditworthiness score of applicant.

- **LoanAmount**: Loan amount requested.

- **LoanStatus**: Approval or rejection of loan.

- **EmploymentStatus**: Employment details of the applicant.

Dataset issues:

- Missing values in Income, CreditScore, and LoanAmount.

- Duplicate entries in ApplicationID.

- Outliers in Income and LoanAmount.

5. Use-Case Explanation:

- **Data Cleaning**: Ensure dataset quality by handling missing values.

- **Duplicate Removal**: Keep only unique applications.

- **Categorical Standardization**: Ensure LoanStatus is uniform (Approved/Rejected).

- **Outlier Detection**: Apply IQR method to identify extreme values in Income and LoanAmount.

- **Risk Filtering**: Detect applicants with very high LoanAmount relative to Income.

- **Aggregation**: Summarize average loan amounts by EmploymentStatus for better insights.

6. UML Diagrams:

- **Use Case Diagram**: Shows interactions between user and system (data loading, cleaning, outlier detection).

- **Activity Diagram**: Steps of preprocessing (load → inspect → clean → detect outliers → export).
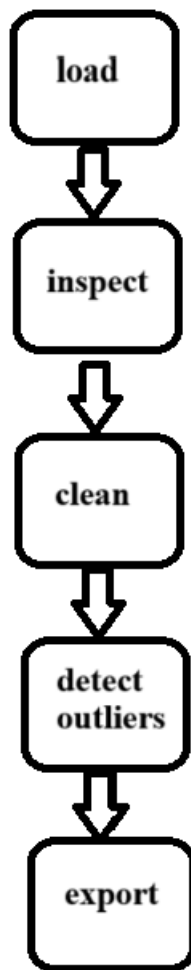
- **Data Flow Diagram (DFD)**: Flow of dataset from raw input → cleaning process → cleaned output.

7. Explanation of the Code:

This project uses **Streamlit** (a Python framework for interactive data apps) along with **Pandas** and **NumPy** to preprocess bank loan data and detect outliers. Below is a step-by-step explanation of the implementation:

```python
import streamlit as st
import pandas as pd
import numpy as np
st.set_page_config(page_title="Bank Loan Data Preprocessing", layout="wide")
```

- streamlit → for the interactive web interface.

- pandas → for data manipulation.

- numpy → for handling numerical operations.

- st.set_page_config → sets page title and wide layout for better readability.

```python
st.title(" Bank Loan Application Data Preprocessing & Outlier Detection")
st.markdown("Upload your CSV file and preprocess the data step by step.")


uploaded_file = st.file_uploader("Upload CSV", type=["csv"])
```

- Displays project title and description.

- Allows user to upload a **CSV file** containing raw loan application data.

```python
df = pd.read_csv(uploaded_file)
st.subheader(" Raw Data (Full Details)")
st.dataframe(df)
```

- Reads uploaded CSV into a Pandas DataFrame.
- Displays the **full dataset** in an interactive table.

```python
for col in ["Income", "CreditScore", "LoanAmount"]:
    if col in df.columns:
        df[col] = pd.to_numeric(df[col], errors="coerce")
```

```python
st.write(df.isnull().sum())    # Show missing values per column
df["Income"].fillna(df["Income"].median(), inplace=True)
df["CreditScore"].fillna(round(df["CreditScore"].mean()), inplace=True)
df = df.dropna(subset=["LoanAmount"])
```

- Ensures key columns (Income, CreditScore, LoanAmount) are numeric.

- Converts invalid entries into NaN (missing values).
- Shows count of missing values and affected rows.

  Fills:

- Income → with **median**.

- CreditScore → with **mean**.

- Drops rows where LoanAmount is missing (critical field).

```
df = df.drop_duplicates(subset=["ApplicationID"], keep="first")
```

```
df["LoanStatus"] = df["LoanStatus"].astype(str).str.strip().str.lower().map(
    lambda x: "Approved" if str(x).startswith("app") else "Rejected"
)
```

```
def detect_outliers(series):
    q1 = series.quantile(0.25)
    q3 = series.quantile(0.75)
    iqr = q3 - q1
    lower = q1 - 1.5 * iqr
    upper = q3 + 1.5 * iqr
    return lower, upper
```

- Removes duplicate loan applications based on ApplicationID.
- Keeps only the first occurrence.

- Ensures **LoanStatus** has consistent values (Approved / Rejected).

- Removes inconsistencies like extra spaces or capitalization issues.

- For `Income` and `LoanAmount`, calculates **Interquartile Range (IQR)**.
- Identifies values **outside (Q1 − 1.5×IQR, Q3 + 1.5×IQR)** as outliers.
- Marks them with `True/False` and highlights in **red**.

```
high_risk = df[df["LoanAmount"] > 2 * df["Income"]]
```

```
group_avg = df.groupby("EmploymentStatus")["LoanAmount"].mean().reset_index()
```

```
cleaned_csv = df.to_csv(index=False).encode("utf-8")
st.download_button(
    label="Download cleaned CSV",
    data=cleaned_csv,
    file_name="cleaned_loan_data.csv",
    mime="text/csv",
)
```

```
st.success("✅ Preprocessing completed!")
```

- Identifies applicants where LoanAmount > 2 × Income.
- These are considered **high-risk loans**.

- Groups data by **Employment Status**.

- Calculates **average LoanAmount** per group.

- Converts cleaned DataFrame to CSV format.
- Provides **download button** for users to save the cleaned dataset.
- Displays success message once all steps are complete.

8.Output Screenshots with explanation:

📊 **Bank Loan Application Data Preprocessing & Outlier Detection**

Upload your CSV file and preprocess the data step by step.

Upload CSV

| ☁ | Drag and drop file here
Limit 200MB per file • CSV | Browse files |

Please upload a CSV file to begin.

- This is the output screen where it will ask to upload the CSV file.

🔍 Raw Data (Full Details)

	ApplicationID	Name	Age	EmploymentStatus	Income	CreditScore	LoanAmount	LoanStatus
0	A001	John Smith	29	Employed	45000	720	15000	Approved
1	A002	Sarah Johnson	35	Self-Employed	60000	680	25000	Approved
2	A003	Michael Lee	42	Employed	52000	None	18000	Rejected
3	A004	Emily Davis	28	Unemployed	None	650	8000	Rejected
4	A005	David Wilson	50	Employed	85000	710	40000	Approved
5	A006	Linda Martinez	31	Employed	47000	600	25000	Rejected
6	A007	James Anderson	38	Self-Employed	120000	730	60000	Approved
7	A008	Mary Thomas	45	Employed	95000	710	100000	Approved
8	A009	Robert Jackson	52	Retired	30000	500	20000	Rejected
9	A010	Patricia White	33	Employed	48000	650	25000	Approved

- This is the raw data which has taken the data from the uploaded CSV file.

🛠 Handle Missing Values

Missing values (count per column):

	0
ApplicationID	0
Name	0
Age	0
EmploymentStatus	0
Income	1
CreditScore	1
LoanAmount	1
LoanStatus	0

🔎 Rows with Missing Values:

	ApplicationID	Name	Age	EmploymentStatus	Income	CreditScore	LoanAmount	LoanStatus
2	A003	Michael Lee	42	Employed	52000	None	18000	Rejected
3	A004	Emily Davis	28	Unemployed	None	650	8000	Rejected
12	A012	Alex Brown	26	Employed	38000	640	None	Rejected

✅ Removed 1 duplicate rows based on `ApplicationID`.

- This shows the missing values and along the rows with missing values.

🚨 Outlier Detection (IQR method)

Outliers in **Income**: 1

🔎 Highlighted Outliers in **Income** (outside -15000.00 - 145000.00):

	Income
3	47500.000000
4	85000.000000
5	47000.000000
6	120000.000000
7	95000.000000
8	30000.000000
9	48000.000000
10	45000.000000
13	300000.000000
14	35000.000000

- The outlier detection by using IQR method , this shows the outliers in income by highlighting that row.

Outliers in **LoanAmount**: 1

🔍 Highlighted Outliers in **LoanAmount** (outside -45000.00 - 123000.00):

	LoanAmount
3	8000.000000
4	40000.000000
5	25000.000000
6	60000.000000
7	100000.000000
8	20000.000000
9	25000.000000
10	15000.000000
13	400000.000000
14	90000.000000

- This shows the outliers in Loan Amount by highlighting that row.

⚠️ High-Risk Loans (LoanAmount > 2 × Income)

	ApplicationID	Name	Age	EmploymentStatus	Income	CreditScore	LoanAmount	LoanStatus	Income_is_outlier	LoanAmount_is_outlier	NeedsReview
14	A014	Chris Blue	36	Employed	35000	690	90000	Approved	☐	☐	☐

📌 Average LoanAmount by Employment Status

	EmploymentStatus	LoanAmount
0	Employed	41000
1	Retired	20000
2	Self-Employed	161666.6667
3	Unemployed	8000

💾 Download Cleaned Data

Download cleaned CSV

✅ Preprocessing completed!

- This displays the high-risk loans , the loan amount which is greater than the twice of the income is considered as the high-risk loan.
- It also displays the average loan amount by employment status.
- And it also gives the cleaned data and that can be downloaded.

9.Conclusion:

The project *"Bank Loan Application Data Preprocessing and Outlier Detection"* successfully demonstrates the importance of data preprocessing in real-world datasets. By performing systematic steps such as handling missing values, removing duplicates, standardizing categorical variables, and detecting outliers using the IQR method, the dataset was transformed into a cleaner and more reliable form.

This process not only improves the quality of the data but also ensures that it is ready for further analysis or machine learning tasks. Outlier detection helped in identifying unusual

records that may affect model performance or business decision-making, while data cleaning ensured consistency and accuracy.

Overall, the project highlights the role of data preprocessing as a crucial foundation for any data-driven system. The cleaned and structured dataset can now be used for advanced modeling, prediction, or decision-support applications in the banking domain, making the loan approval process more transparent, efficient, and data-driven.

10.Bibliography:

- Pandas Documentation." https://pandas.pydata.org
- "NumPy Documentation." https://numpy.org/doc
- "Matplotlib Documentation." https://matplotlib.org/stable/contents.html
- "Streamlit Documentation." https://docs.streamlit.io