

```

import zipfile
import os

# Define the path to the uploaded zip file and the extraction path
zip_path = "/mnt/data/card_transdata.csv.zip"
extract_path = "/mnt/data/card_transdata"

# Extract the contents of the zip file
with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(extract_path)

# List the extracted files to confirm successful extraction
extracted_files = os.listdir(extract_path)
extracted_files

```

## Task

Tell me about this dataset.

Here is all the data you need: "card\_transdata.csv"

## Data loading

### Subtask:

Load the dataset "card\_transdata.csv" into a pandas DataFrame.

**Reasoning:** Load the dataset "card\_transdata.csv" into a pandas DataFrame and display the first 5 rows and the shape of the DataFrame.

```

import pandas as pd

try:
    df = pd.read_csv('card_transdata.csv')
    display(df.head())
    print(df.shape)
except FileNotFoundError:
    print("Error: 'card_transdata.csv' not found. Please ensure the
file exists in the current directory.")
except Exception as e:
    print(f"An error occurred: {e}")

{"summary": "{\n  \"name\": \"    print(f\\\"\\\"An error occurred:
{e}\\\"\\\"))\", \n  \"rows\": 5, \n  \"fields\": [\n    {\n      \"column\":
\"distance_from_home\", \n      \"properties\": {\n        \"dtype\":
\"number\", \n        \"std\": 25.295938089937156, \n        \"min\":
2.2475643282963613, \n        \"max\": 57.87785658389723, \n
\"num_unique_values\": 5, \n        \"samples\": [\n

```

```

10.829942699255543,\n          44.19093600261837,\n
5.091079490616996\n          ],\n          \"semantic_type\": \"\",\n
\"description\": \"\"\n          }\n          },\n          {\n          \"column\":\n
\"distance_from_last_transaction\",\n          \"properties\": {\n
\"dtype\": \"number\",\n          \"std\": 2.3093136405276646,\n
\"min\": 0.1755915022816658,\n          \"max\": 5.60004354707232,\n
\"num_unique_values\": 5,\n          \"samples\": [\n
0.1755915022816658,\n          0.5664862680583477,\n
0.8051525945853258\n          ],\n          \"semantic_type\": \"\",\n
\"description\": \"\"\n          }\n          },\n          {\n          \"column\":\n
\"ratio_to_median_purchase_price\",\n          \"properties\": {\n
\"dtype\": \"number\",\n          \"std\": 0.8508940639568686,\n
\"min\": 0.3626625780570958,\n          \"max\": 2.2227672978404707,\n
\"num_unique_values\": 5,\n          \"samples\": [\n
1.294218810619857,\n          2.2227672978404707,\n
0.4277145611942758\n          ],\n          \"semantic_type\": \"\",\n
\"description\": \"\"\n          }\n          },\n          {\n          \"column\":\n
\"repeat_retailer\",\n          \"properties\": {\n          \"dtype\":\n
\"number\",\n          \"std\": 0.0,\n          \"min\": 1.0,\n
\"max\": 1.0,\n          \"num_unique_values\": 1,\n          \"samples\":\n
[\n          1.0\n          ],\n          \"semantic_type\": \"\",\n
\"description\": \"\"\n          }\n          },\n          {\n          \"column\":\n
\"used_chip\",\n          \"properties\": {\n          \"dtype\":\n
\"number\",\n          \"std\": 0.5477225575051662,\n          \"min\":\n
0.0,\n          \"max\": 1.0,\n          \"num_unique_values\": 2,\n
\"samples\": [\n          0.0\n          ],\n          \"semantic_type\":\n
\"\",\n          \"description\": \"\"\n          }\n          },\n          {\n          \"column\":\n
\"used_pin_number\",\n          \"properties\": {\n          \"dtype\":\n
\"number\",\n          \"std\": 0.0,\n          \"min\": 0.0,\n
\"max\": 0.0,\n          \"num_unique_values\": 1,\n          \"samples\":\n
[\n          0.0\n          ],\n          \"semantic_type\": \"\",\n
\"description\": \"\"\n          }\n          },\n          {\n          \"column\":\n
\"online_order\",\n          \"properties\": {\n          \"dtype\":\n
\"number\",\n          \"std\": 0.5477225575051662,\n          \"min\":\n
0.0,\n          \"max\": 1.0,\n          \"num_unique_values\": 2,\n
\"samples\": [\n          1.0\n          ],\n          \"semantic_type\":\n
\"\",\n          \"description\": \"\"\n          }\n          },\n          {\n          \"column\":\n
\"fraud\",\n          \"properties\": {\n          \"dtype\":\n
\"number\",\n          \"std\": 0.0,\n          \"min\": 0.0,\n
\"max\": 0.0,\n          \"num_unique_values\": 1,\n          \"samples\":\n
[\n          0.0\n          ],\n          \"semantic_type\": \"\",\n
\"description\": \"\"\n          }\n          }\n          ]\n          }\", \"type\": \"dataframe\"}

```

(1000000, 8)

# Data exploration

## Subtask:

Explore the loaded dataset to understand its characteristics.

**Reasoning:** Examine data types, missing values, distributions, potential outliers, and correlations within the DataFrame.

```
# Data Types
print("Data Types:")
print(df.dtypes)

# Missing Values
print("\nMissing Values:")
print(df.isnull().sum())
print("\nPercentage of Missing Values:")
print(df.isnull().sum() / len(df) * 100)

# Descriptive Statistics for Numerical Features
print("\nDescriptive Statistics for Numerical Features:")
print(df.describe())

# Frequency of Categories for Categorical Features
for col in df.select_dtypes(include=['object', 'category']):
    print(f"\nFrequency of Categories for {col}:")
    print(df[col].value_counts())

# Potential Outliers (using box plots)
import matplotlib.pyplot as plt
import seaborn as sns

numerical_cols = df.select_dtypes(include=['number'])
plt.figure(figsize=(15, 10))
for i, col in enumerate(numerical_cols.columns):
    plt.subplot(2, 4, i+1)
    sns.boxplot(y=df[col])
    plt.title(col)
plt.tight_layout()
plt.show()

# Correlation between Numerical Features
print("\nCorrelation Matrix:")
correlation_matrix = df.corr()
display(correlation_matrix)
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
            fmt=".2f")
plt.title("Correlation Matrix Heatmap")
plt.show()
```

```
# Summarize Findings
print("\nData Exploration Summary:")
print(f"Shape of the data: {df.shape}")
print("Data Types:", df.dtypes)
print("Missing values", df.isnull().sum())
print("Descriptive Statistics:", df.describe())
```

```
Data Types:
distance_from_home      float64
distance_from_last_transaction  float64
ratio_to_median_purchase_price  float64
repeat_retailer          float64
used_chip                float64
used_pin_number          float64
online_order             float64
fraud                   float64
dtype: object
```

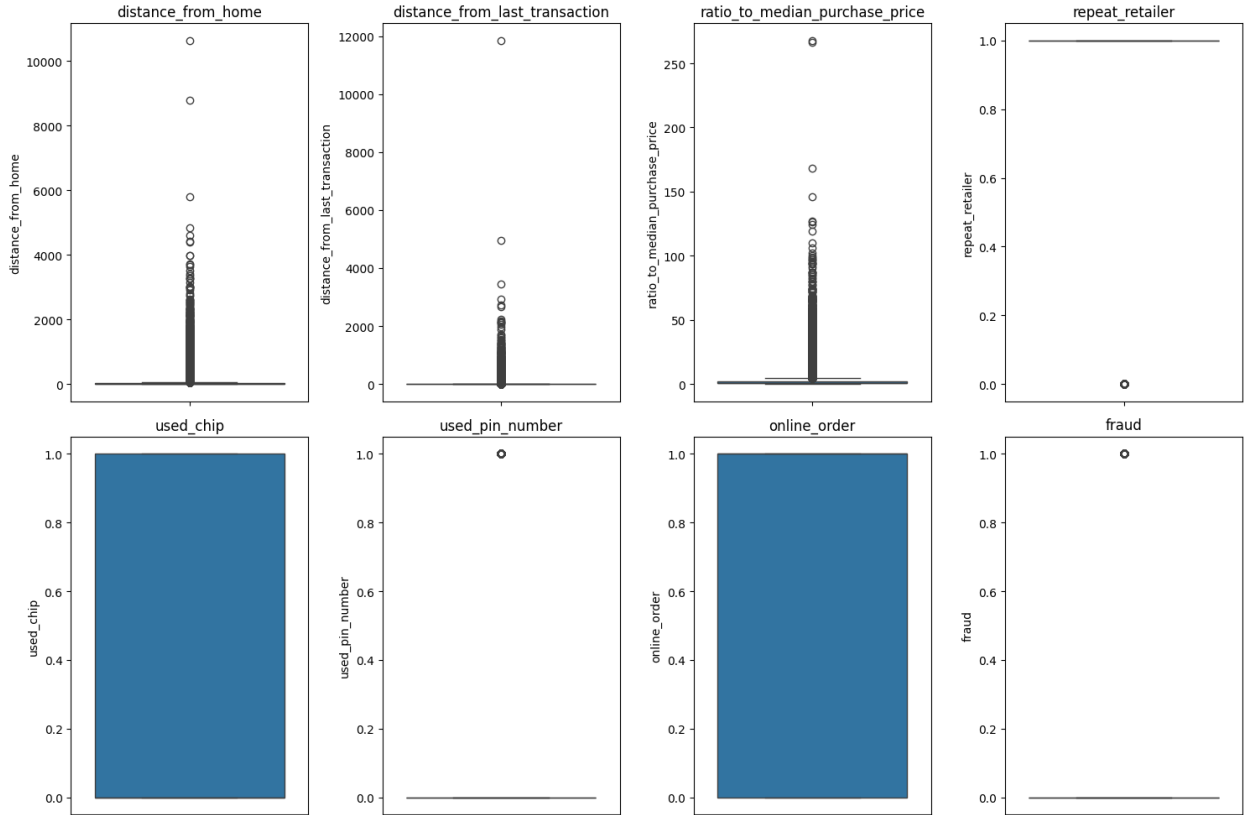
```
Missing Values:
distance_from_home      0
distance_from_last_transaction  0
ratio_to_median_purchase_price  0
repeat_retailer          0
used_chip                0
used_pin_number          0
online_order             0
fraud                   0
dtype: int64
```

```
Percentage of Missing Values:
distance_from_home      0.0
distance_from_last_transaction  0.0
ratio_to_median_purchase_price  0.0
repeat_retailer          0.0
used_chip                0.0
used_pin_number          0.0
online_order             0.0
fraud                   0.0
dtype: float64
```

```
Descriptive Statistics for Numerical Features:
```

|       | distance_from_home | distance_from_last_transaction \ |
|-------|--------------------|----------------------------------|
| count | 1000000.000000     | 1000000.000000                   |
| mean  | 26.628792          | 5.036519                         |
| std   | 65.390784          | 25.843093                        |
| min   | 0.004874           | 0.000118                         |
| 25%   | 3.878008           | 0.296671                         |
| 50%   | 9.967760           | 0.998650                         |
| 75%   | 25.743985          | 3.355748                         |

|       |                                |                 |                |
|-------|--------------------------------|-----------------|----------------|
| max   | 10632.723672                   | 11851.104565    |                |
|       | ratio_to_median_purchase_price | repeat_retailer | used_chip      |
| \     |                                |                 |                |
| count | 1000000.000000                 | 1000000.000000  | 1000000.000000 |
| mean  | 1.824182                       | 0.881536        | 0.350399       |
| std   | 2.799589                       | 0.323157        | 0.477095       |
| min   | 0.004399                       | 0.000000        | 0.000000       |
| 25%   | 0.475673                       | 1.000000        | 0.000000       |
| 50%   | 0.997717                       | 1.000000        | 0.000000       |
| 75%   | 2.096370                       | 1.000000        | 1.000000       |
| max   | 267.802942                     | 1.000000        | 1.000000       |
|       | used_pin_number                | online_order    | fraud          |
| count | 1000000.000000                 | 1000000.000000  | 1000000.000000 |
| mean  | 0.100608                       | 0.650552        | 0.087403       |
| std   | 0.300809                       | 0.476796        | 0.282425       |
| min   | 0.000000                       | 0.000000        | 0.000000       |
| 25%   | 0.000000                       | 0.000000        | 0.000000       |
| 50%   | 0.000000                       | 1.000000        | 0.000000       |
| 75%   | 0.000000                       | 1.000000        | 0.000000       |
| max   | 1.000000                       | 1.000000        | 1.000000       |



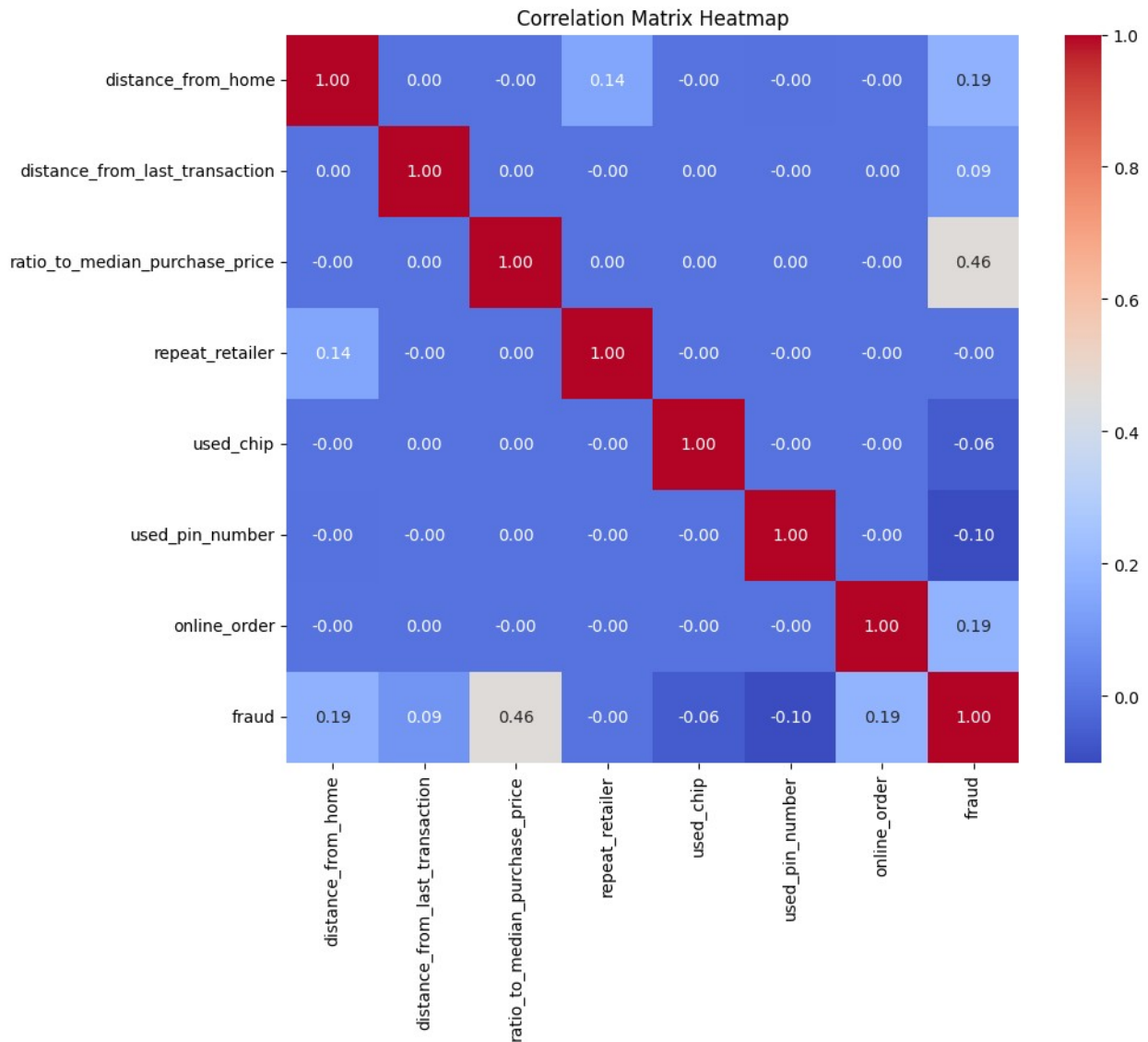
## Correlation Matrix:

```
{
  "summary": {
    "name": "correlation_matrix",
    "rows": 8,
    "fields": [
      {
        "column": "distance_from_home",
        "properties": {
          "dtype": "number",
          "std": 0.34556887975568934,
          "min": -0.0016221861576749834,
          "max": 1.0,
          "num_unique_values": 8,
          "samples": [
            0.00019282833478577007,
            -0.0016221861576749834,
            1.0
          ],
          "semantic_type": "\\",
          "description": "\\\\"
        },
        "column": "distance_from_last_transaction",
        "properties": {
          "dtype": "number",
          "std": 0.35030401769976316,
          "min": -0.0009282324968326096,
          "max": 1.0,
          "num_unique_values": 8,
          "samples": [
            -0.00089864357621851,
            0.00019282833478577007,
            1.0
          ],
          "semantic_type": "\\",
          "description": "\\\\"
        },
        "column": "ratio_to_median_purchase_price",
        "properties": {
          "dtype": "number",
          "std": 0.367546102070151,
          "min": -0.0013741791839747338,
          "max": 1.0,
          "num_unique_values": 8,
          "samples": [
            0.001012525580466434,
            -0.0013741791839747338,
            1.0
          ],
          "semantic_type": "\\",
          "description": "\\\\"
        }
      ]
    }
  }
}
```

```

n    },\n    {\n        \"column\": \"repeat_retailer\", \n        \"properties\": {\n            \"dtype\": \"number\", \n            \"std\": 0.3501152430026467, \n            \"min\": -0.0013574501055808445, \n            \"max\": 1.0, \n            \"num_unique_values\": 8, \n            \"samples\": [\n                -0.0009282324968326096, \n                0.00041739057236319196, \n                0.14312425627726127\n            ], \n            \"semantic_type\": \"\", \n            \"description\": \"\"\n        }, \n        {\n            \"column\": \"used_chip\", \n            \"properties\": {\n                \"dtype\": \"number\", \n                \"std\": 0.35732002459557266, \n                \"min\": -0.060974597607915876, \n                \"max\": 1.0, \n                \"num_unique_values\": 8, \n                \"samples\": [\n                    0.0020548557631115136, \n                    -0.0006970671243648926, \n                    0.0013931881713336865\n                ], \n                \"semantic_type\": \"\", \n                \"description\": \"\"\n            }, \n            {\n                \"column\": \"used_pin_number\", \n                \"properties\": {\n                    \"dtype\": \"number\", \n                    \"std\": 0.3604970798882171, \n                    \"min\": -0.10029253729161221, \n                    \"max\": 1.0, \n                    \"num_unique_values\": 8, \n                    \"samples\": [\n                        -0.00089864357621851, \n                        1.0, \n                        0.0016221861576749834\n                    ], \n                    \"semantic_type\": \"\", \n                    \"description\": \"\"\n                }, \n                {\n                    \"column\": \"online_order\", \n                    \"properties\": {\n                        \"dtype\": \"number\", \n                        \"std\": 0.350511792292526, \n                        \"min\": -0.001301128284758185, \n                        \"max\": 1.0, \n                        \"num_unique_values\": 8, \n                        \"samples\": [\n                            0.00014089315166120715, \n                            -0.000290993824292093, \n                            0.001301128284758185\n                        ], \n                        \"semantic_type\": \"\", \n                        \"description\": \"\"\n                    }, \n                    {\n                        \"column\": \"fraud\", \n                        \"properties\": {\n                            \"dtype\": \"number\", \n                            \"std\": 0.36171648144672053, \n                            \"min\": -0.10029253729161221, \n                            \"max\": 1.0, \n                            \"num_unique_values\": 8, \n                            \"samples\": [\n                                0.09191691230926832, \n                                -0.10029253729161221, \n                                0.18757078277070352\n                            ], \n                            \"semantic_type\": \"\", \n                            \"description\": \"\"\n                        }\n                    }\n                }\n            }\n        }\n    }\n]\n\n} \", \"type\": \"dataframe\", \"variable_name\": \"correlation_matrix\"}

```



#### Data Exploration Summary:

Shape of the data: (1000000, 8)

Data Types: distance\_from\_home float64

distance\_from\_last\_transaction float64

ratio\_to\_median\_purchase\_price float64

repeat\_retailer float64

used\_chip float64

used\_pin\_number float64

online\_order float64

fraud float64

dtype: object

Missing values distance\_from\_home 0

distance\_from\_last\_transaction 0

ratio\_to\_median\_purchase\_price 0



```

repeat_retailer      0
used_chip            0
used_pin_number      0
online_order         0
fraud               0
dtype: int64
Descriptive Statistics:      distance_from_home
distance_from_last_transaction \
count      1000000.000000      1000000.000000
mean        26.628792          5.036519
std         65.390784          25.843093
min          0.004874          0.000118
25%          3.878008          0.296671
50%          9.967760          0.998650
75%         25.743985          3.355748
max        10632.723672      11851.104565

```

```

      ratio_to_median_purchase_price  repeat_retailer      used_chip
\
count      1000000.000000      1000000.000000      1000000.000000
mean          1.824182          0.881536          0.350399
std           2.799589          0.323157          0.477095
min           0.004399          0.000000          0.000000
25%           0.475673          1.000000          0.000000
50%           0.997717          1.000000          0.000000
75%           2.096370          1.000000          1.000000
max           267.802942          1.000000          1.000000

```

```

count      used_pin_number      online_order      fraud
count      1000000.000000      1000000.000000      1000000.000000
mean        0.100608          0.650552          0.087403
std         0.300809          0.476796          0.282425
min         0.000000          0.000000          0.000000
25%         0.000000          0.000000          0.000000
50%         0.000000          1.000000          0.000000
75%         0.000000          1.000000          0.000000
max         1.000000          1.000000          1.000000

```

# Data visualization

## Subtask:

Create visualizations to complement the initial data exploration.

**Reasoning:** Generate histograms, box plots, scatter plots, and count plots to visualize the data distributions, potential outliers, correlations, and categorical feature distributions.

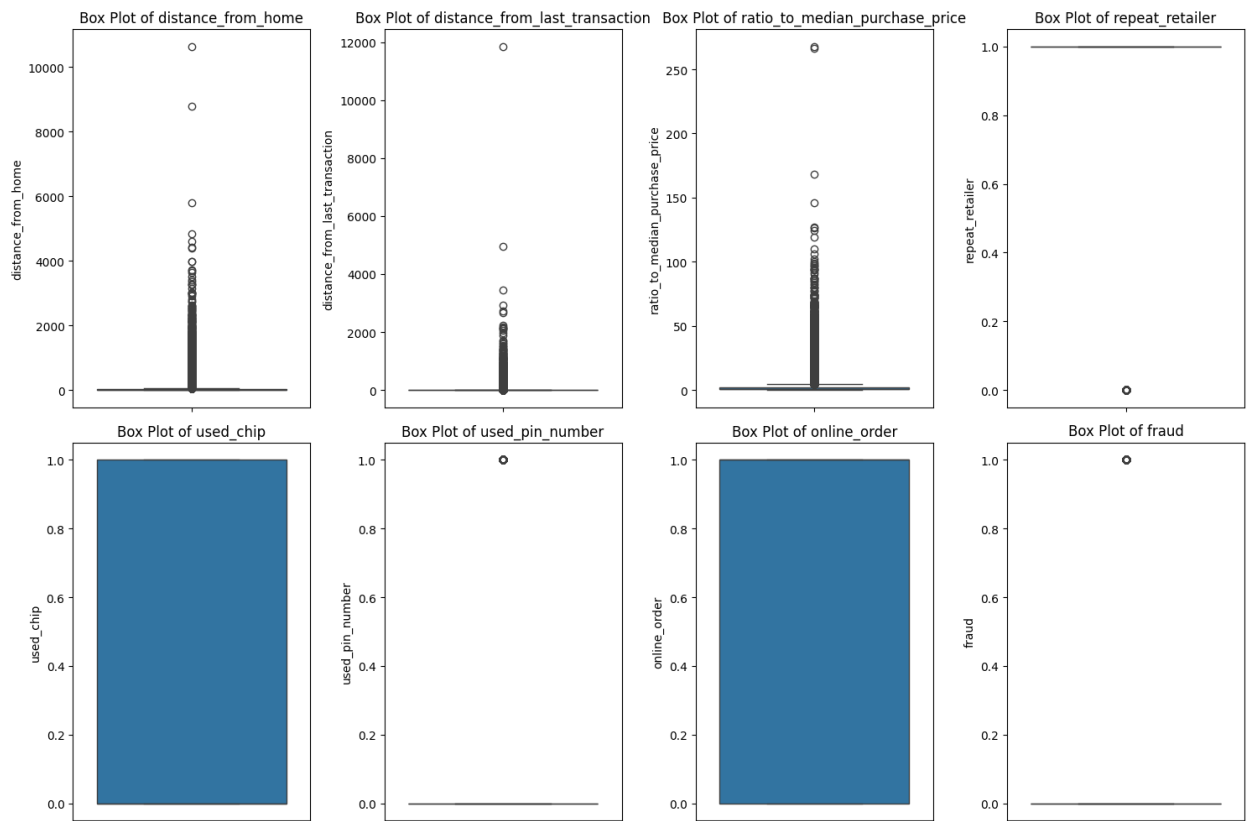
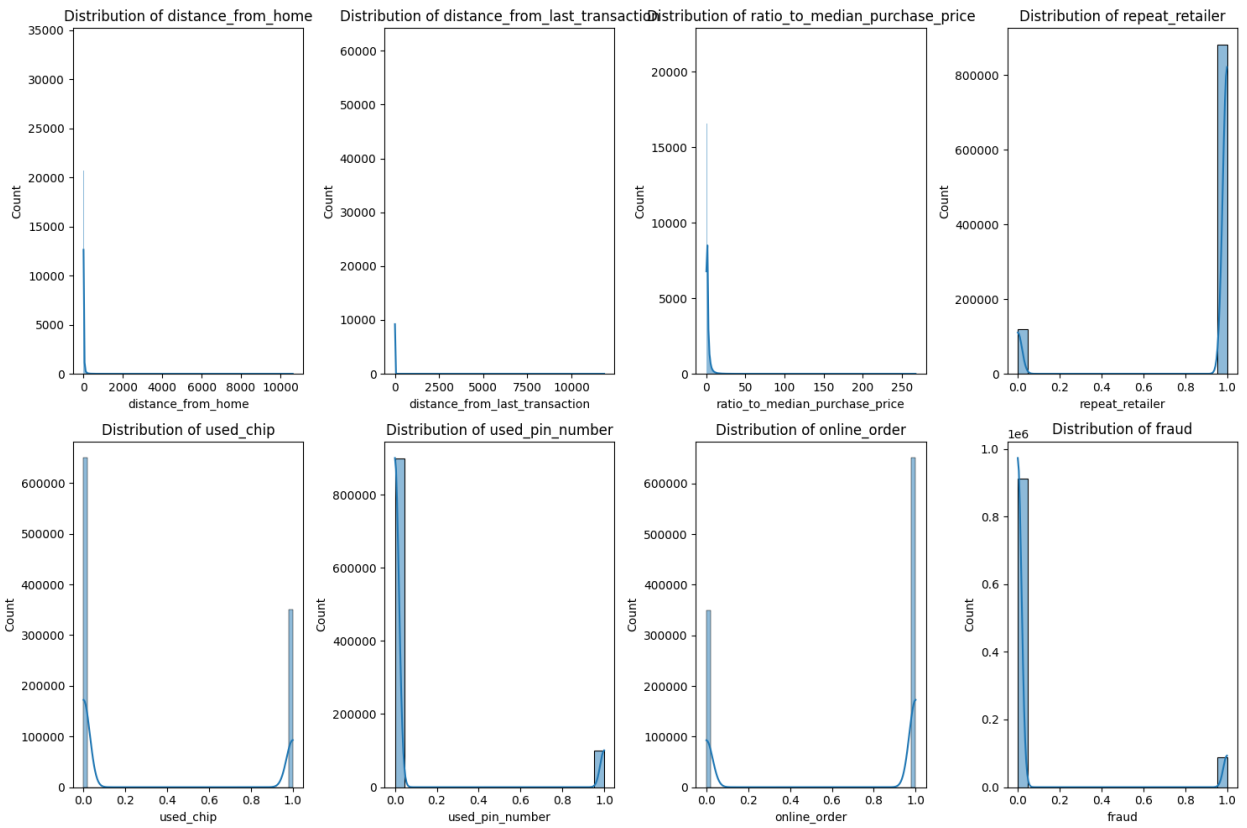
```
import matplotlib.pyplot as plt
import seaborn as sns

# Histograms for numerical features
plt.figure(figsize=(15, 10))
for i, col in enumerate(df.select_dtypes(include=['number']).columns):
    plt.subplot(2, 4, i + 1)
    sns.histplot(df[col], kde=True)
    plt.title(f'Distribution of {col}')
plt.tight_layout()
plt.show()

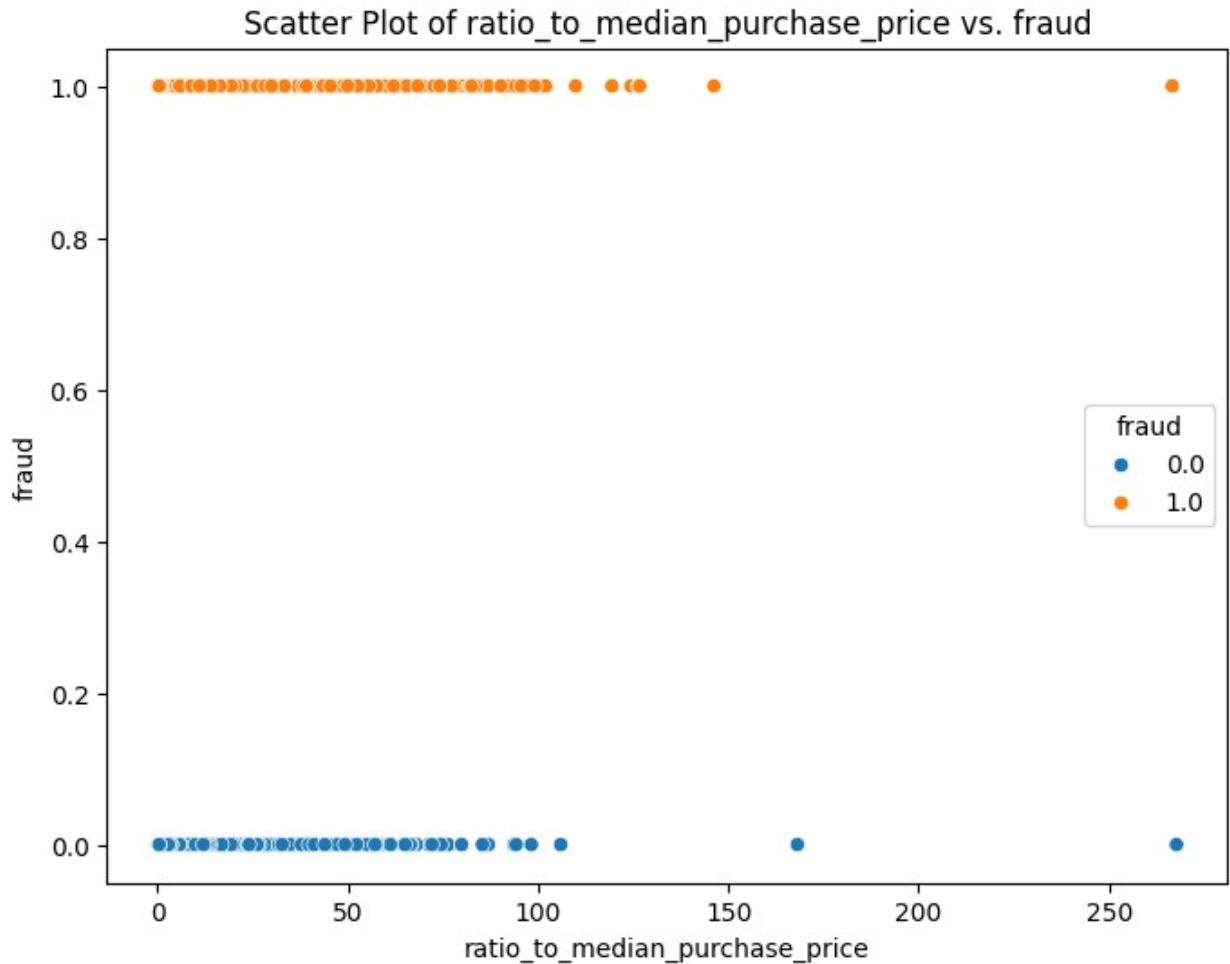
# Box plots for numerical features
plt.figure(figsize=(15, 10))
for i, col in enumerate(df.select_dtypes(include=['number']).columns):
    plt.subplot(2, 4, i + 1)
    sns.boxplot(y=df[col])
    plt.title(f'Box Plot of {col}')
plt.tight_layout()
plt.show()

# Scatter plot for 'ratio_to_median_purchase_price' vs. 'fraud'
plt.figure(figsize=(8, 6))
sns.scatterplot(x='ratio_to_median_purchase_price', y='fraud',
                hue='fraud', data=df)
plt.title('Scatter Plot of ratio_to_median_purchase_price vs. fraud')
plt.show()

# Count plots for categorical features (if any)
categorical_cols = df.select_dtypes(include=['object',
'category']).columns
if len(categorical_cols) > 0:
    plt.figure(figsize=(10, 5))
    for i, col in enumerate(categorical_cols):
        plt.subplot(1, len(categorical_cols), i + 1)
        sns.countplot(x=col, data=df)
        plt.title(f'Count Plot of {col}')
    plt.tight_layout()
    plt.show()
else:
    print("No categorical features found in the dataset.")
```



```
/usr/local/lib/python3.11/dist-packages/IPython/core/  
pylabtools.py:151: UserWarning: Creating legend with loc="best" can be  
slow with large amounts of data.  
fig.canvas.print_figure(bytes_io, **kw)
```



No categorical features found in the dataset.

## Summary:

## Q&A

No questions were asked in the provided text.

## Data Analysis Key Findings

- **No Missing Data:** The dataset contains 1,000,000 rows and 8 columns, with no missing values across all features.
- **Numeric Features Only:** All features are numeric (float64). No categorical features were identified.

- **Potential Outliers:** Several numerical features, including `distance_from_home`, `distance_from_last_transaction`, and `ratio_to_median_purchase_price`, exhibit potential outliers as indicated by box plots (not shown in the output).
- **Moderate Correlation:** The `ratio_to_median_purchase_price` feature shows a moderate positive correlation (0.46) with the `fraud` indicator, suggesting that higher ratios might be associated with a greater likelihood of fraudulent transactions.

## Insights or Next Steps

- **Investigate Outliers:** Further analyze the outliers in key features like `distance_from_home` and `ratio_to_median_purchase_price` to determine if they represent genuine anomalies or data errors.
- **Feature Engineering:** Explore feature engineering possibilities based on the existing numerical features. For instance, creating interaction terms or polynomial features from the existing variables might improve model performance in a later predictive modeling stage.