UNIFIED MENTOR
YOUR SKILL, SUCCESS & JOURNEY

Project Title	Daily Transactions
Tools	Visual Studio code / jupyter notebook
Domain	Finance Analyst
Project Difficulties level	intermediate

Dataset : Dataset is available in the given link. You can download it at your convenience.

Click here to download data set

About Dataset

The "Daily Transactions" dataset contains information on dummy transactions made by an individual on a daily basis. The dataset includes data on the products that were purchased, the amount spent on each product, the date and time of each transaction, the payment mode of each transaction, and the source of each record (Expense/Income).

This dataset can be used to analyze purchasing behavior and money management, forecasting expenses, and optimizing savings and budgeting strategies. The dataset is well-suited for data analysis and machine learning applications,it can be used to train predictive models and make data-driven decisions.

Column Descriptors

- **Date:** The date and time when the transaction was made
- **Mode:** The payment mode used for the transaction
- **Category:** Each record is divided into a set of categories of transactions
- **Subcategory:** Categories are further broken down into Subcategories of transactions
- **Note:** A brief description of the transaction made
- **Amount:** The transactional amount
- **Income/Expense:** The indicator of each transaction representing either expense or income
- **Currency:** All transactions are recorded in official currency of India

Example: You can get the basic idea how you can create a project from here

Sure! Let's outline a financial analyst project that involves working with a dataset of daily transactions. We'll include steps to clean the data, perform analysis, and generate a report with code examples in Python using popular libraries like Pandas, NumPy, Matplotlib, and Seaborn.

1. Project Overview

Objective:

- Analyze daily financial transactions to identify trends, patterns, and insights.
- Generate a comprehensive report with visualizations.

2. Dataset Description

- **Date**: Date of the transaction.
- **Transaction_ID**: Unique identifier for each transaction.
- **Account_ID**: Unique identifier for the account.
- **Category**: Category of the transaction (e.g., Sales, Purchase, Salary).
- **Amount**: Amount of money involved in the transaction.
- **Type**: Type of transaction (Credit or Debit).

3. Steps to Complete the Project

Step 1: Import Libraries and Load Data

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Load the dataset
df = pd.read_csv('daily_transactions.csv')

# Display the first few rows of the dataset
df.head()
```

Step 2: Data Cleaning

- Handle missing values.
- Correct data types.
- Remove duplicates.

```
# Check for missing values
df.isnull().sum()

# Fill or drop missing values
df['Category'].fillna('Unknown', inplace=True)
df.dropna(subset=['Date', 'Transaction_ID', 'Amount'], inplace=True)

# Convert data types
df['Date'] = pd.to_datetime(df['Date'])
df['Amount'] = df['Amount'].astype(float)

# Remove duplicates
df.drop_duplicates(inplace=True)
```

```
# Verify data types
df.dtypes
```

Step 3: Exploratory Data Analysis (EDA)

- Summary statistics.
- Distribution of transaction amounts.
- Transaction counts by category and type.

```
# Summary statistics
df.describe()

# Distribution of transaction amounts
plt.figure(figsize=(10, 6))
sns.histplot(df['Amount'], bins=50, kde=True)
plt.title('Distribution of Transaction Amounts')
plt.xlabel('Amount')
plt.ylabel('Frequency')
plt.show()

# Transaction counts by category
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='Category', order=df['Category'].value_counts().index)
plt.title('Transaction Counts by Category')
plt.xlabel('Category')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()

# Transaction counts by type
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Type')
plt.title('Transaction Counts by Type')
plt.xlabel('Type')
plt.ylabel('Count')
plt.show()
```

Step 4: Time Series Analysis

- Trend analysis.
- Monthly and daily trends.

```
# Resample data to monthly frequency
monthly_data = df.resample('M', on='Date').sum()

plt.figure(figsize=(14, 7))
plt.plot(monthly_data.index, monthly_data['Amount'], marker='o')
plt.title('Monthly Transaction Amounts')
plt.xlabel('Month')
plt.ylabel('Total Amount')
plt.grid(True)
plt.show()

# Daily trends
daily_data = df.groupby(df['Date'].dt.date).sum()

plt.figure(figsize=(14, 7))
plt.plot(daily_data.index, daily_data['Amount'], marker='o')
plt.title('Daily Transaction Amounts')
plt.xlabel('Date')
plt.ylabel('Total Amount')
plt.grid(True)
plt.show()
```

Step 5: Correlation Analysis

- Analyze the correlation between transaction categories and amounts.

```
# Create a pivot table for correlation analysis
pivot_table = df.pivot_table(index='Date', columns='Category', values='Amount',
aggfunc='sum', fill_value=0)

# Calculate correlation matrix
correlation_matrix = pivot_table.corr()
```

```
# Plot correlation heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap of Transaction Categories')
plt.show()
```

Step 6: Generate Report

- Summarize findings and visualizations.

4. Report

Summary

The financial transactions dataset was analyzed to identify key trends and insights. The data cleaning process involved handling missing values, correcting data types, and removing duplicates. Exploratory Data Analysis (EDA) revealed the distribution of transaction amounts, transaction counts by category and type, and significant patterns over time. Time series analysis highlighted monthly and daily transaction trends. Correlation analysis identified relationships between different transaction categories.

Key Findings

- The distribution of transaction amounts showed a right-skewed pattern with most transactions clustered around lower values.
- Sales and Purchases were the most common transaction categories.
- Credit transactions were more frequent than Debit transactions.
- Monthly transaction trends revealed seasonal patterns with peaks in certain months.
- Correlation analysis indicated strong relationships between certain transaction categories.

Visualizations

- Distribution of Transaction Amounts
- Transaction Counts by Category and Type
- Monthly and Daily Transaction Amounts
- Correlation Heatmap of Transaction Categories

This project provides valuable insights into daily financial transactions, helping to inform decision-making and strategic planning.

Would you like more details on any specific part of the project or any additional analysis?

Example: You can get the basic idea how you can create a project from here

Sample code with output

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create
```

```
a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
/kaggle/input/daily-transactions-dataset/Daily Household Transactions.csv
```

In [2]:

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

In [3]:

```
df = pd.read_csv("/kaggle/input/daily-transactions-dataset/Daily Household Transactions.csv")
```

In [4]:

```
df.head() #check the first 5 rows of the dataset
```

Out[4]:

	Date	Mode	Category	Subcategory	Note	Amount	Income/Expense	Currency
0	20/09/2018 12:04:08	Cash	Transportation	Train	2 Place 5 to Place 0	30.0	Expense	INR
1	20/09/2018 12:03:15	Cash	Food	snacks	Idli medu Vada mix 2 plates	60.0	Expense	INR
2	19/09/2018	Saving Bank account 1	subscription	Netflix	1 month subscription	199.0	Expense	INR
3	17/09/2018 23:41:17	Saving Bank account 1	subscription	Mobile Service Provider	Data booster pack	19.0	Expense	INR
4	16/09/2018	Cash	Festiv	Ganesh	Ganesh idol	25	Expens	INR

	17:15:08		als	Pujan		1.0	e	

In [5]:

```
df.shape #get the number of rows and columns in the dataset
```

Out[5]:

```
(2461, 8)
```

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2461 entries, 0 to 2460
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Date            2461 non-null   object
 1   Mode            2461 non-null   object
 2   Category        2461 non-null   object
 3   Subcategory     1826 non-null   object
 4   Note            1940 non-null   object
 5   Amount          2461 non-null   float64
 6   Income/Expense  2461 non-null   object
 7   Currency        2461 non-null   object
dtypes: float64(1), object(7)
memory usage: 153.9+ KB
```

```
In [7]:
```

```
df.isnull().sum() #get the null values
```

```
Out[7]:
```

```
Date                0
Mode                0
Category            0
Subcategory       635
Note              521
Amount              0
Income/Expense      0
Currency            0

dtype: int64
```

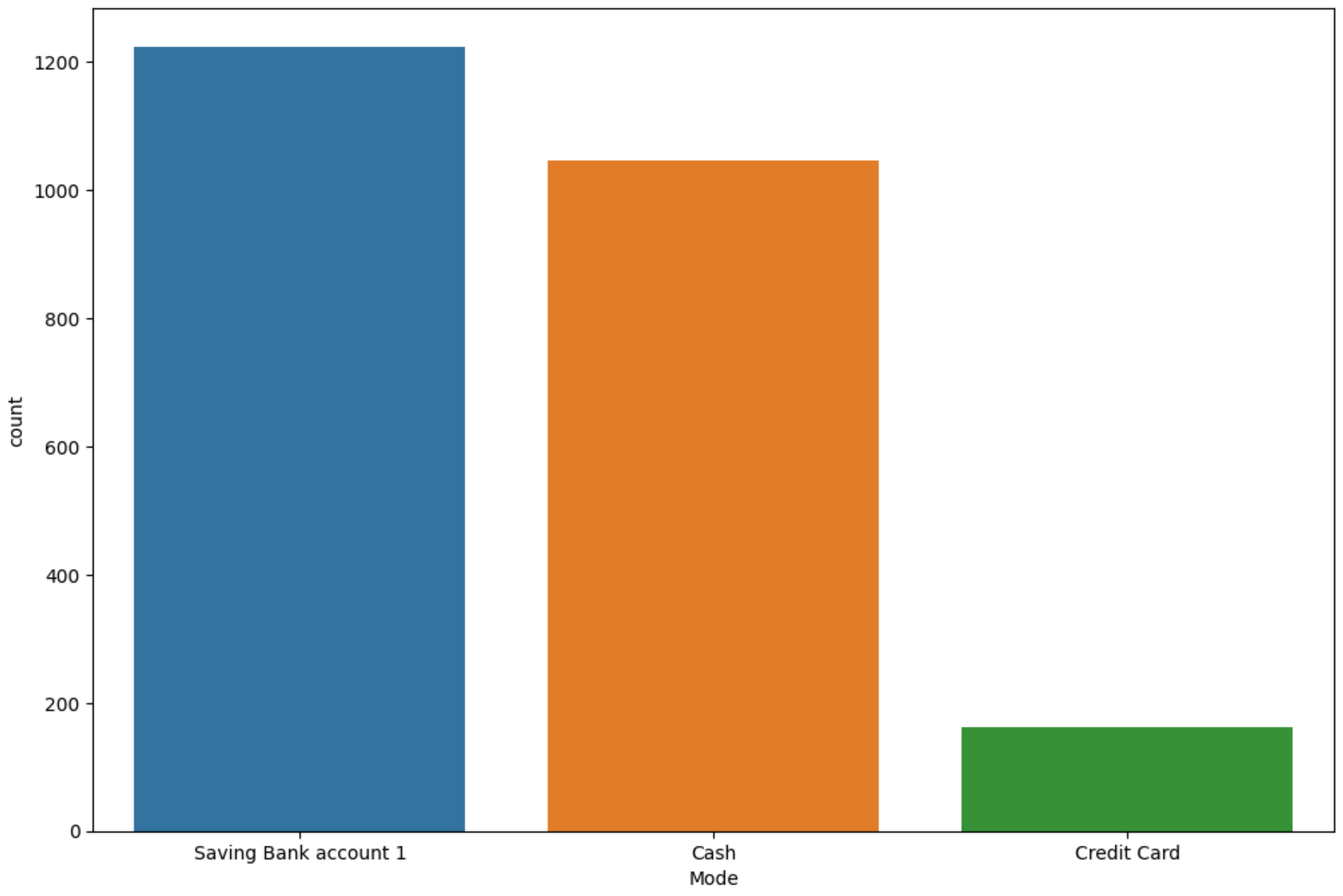
```
In [8]:
```

```
df["Mode"].value_counts()
```

```
Out[8]:
```

```
Mode
Saving Bank account 1    1223
Cash                     1046
Credit Card               162
Equity Mutual Fund B       11
Share Market Trading        5
Saving Bank account 2       5
Recurring Deposit           3
Debit Card                  2
Equity Mutual Fund C        1
Equity Mutual Fund A        1
Equity Mutual Fund D        1
Fixed Deposit               1

Name: count, dtype: int64
```

```
In [9]:
plt.figure(figsize = (12,8))
sns.countplot(data = df, x = "Mode", order =
df["Mode"].value_counts().iloc[:3].index)
plt.show()
```



In [10]:

```
df["Category"].value_counts()
```

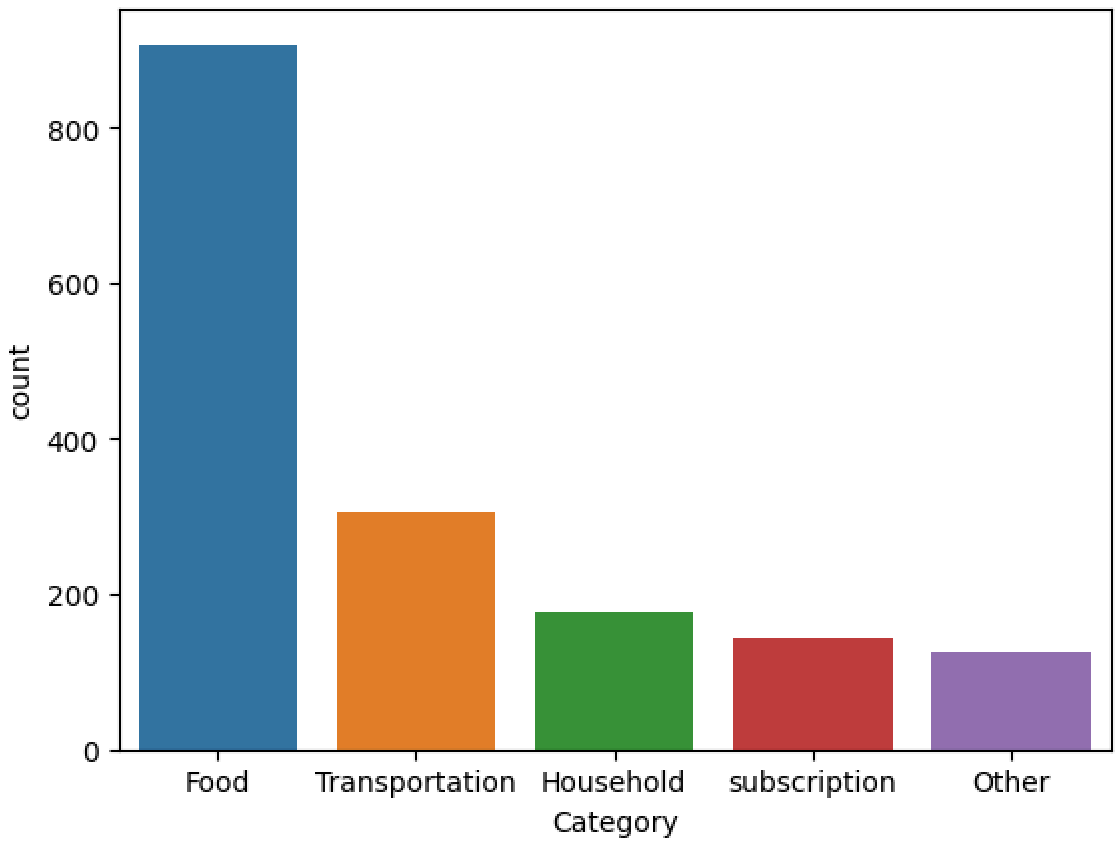
Out[10]:

```
Category
Food                     907
Transportation           307
Household                176
subscription             143
Other                    126
```

```
Investment                    103
Health                         94
Family                         71
Recurring Deposit              47
Apparel                        47
Money transfer                 43
Salary                         43
Gift                           30
Public Provident Fund          29
Equity Mutual Fund E           22
Beauty                         22
Gpay Reward                    21
Education                      18
maid                           17
Saving Bank account 1          17
Festivals                      16
Equity Mutual Fund A           14
Equity Mutual Fund F           13
Interest                       12
Dividend earned on Shares      12
Culture                        11
Small cap fund 1               10
Small Cap fund 2               10
Share Market                    8
Maturity amount                 7
```

```
Life Insurance              7
Bonus                       6
Equity Mutual Fund C        6
Petty cash                  6
Tourism                     5
Cook                        4
Rent                        4
Grooming                    4
water (jar /tanker)         3
Saving Bank account 2       3
garbage disposal            2
scrap                       2
Fixed Deposit               2
Self-development            2
Amazon pay cashback         2
Documents                   2
Tax refund                  2
Equity Mutual Fund B        1
Equity Mutual Fund D        1
Social Life                 1

Name: count, dtype: int64
```

```
In [11]:
sns.countplot(data = df, x = "Category", order =
df["Category"].value_counts().iloc[:5].index);
```



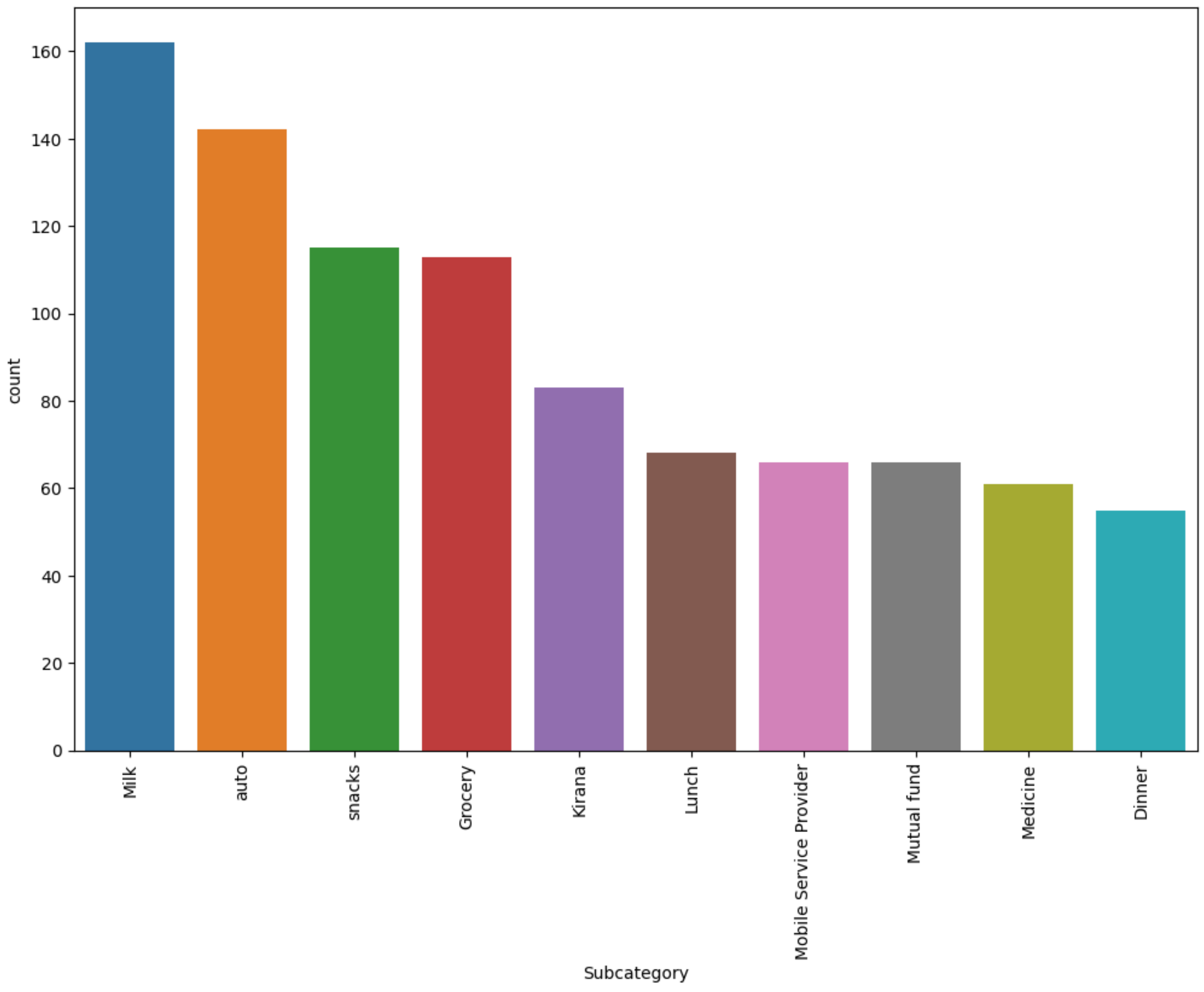
In [12]:

```
df["Subcategory"].unique()
```

Out[12]:

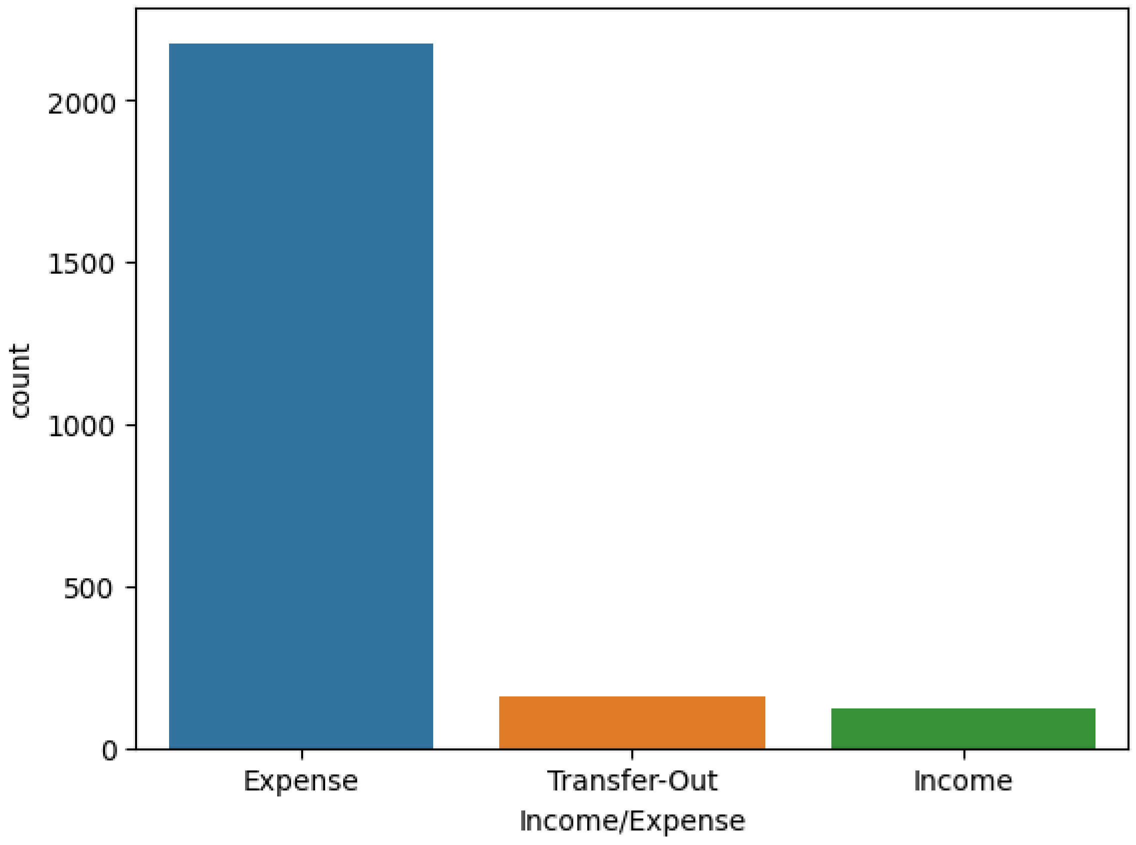
```
array(['Train', 'snacks', 'Netflix', 'Mobile Service Provider',
       'Ganesh Pujan', 'Tata Sky', 'auto', nan, 'Grocery',
'Lunch',
       'Milk', 'Pocket money', 'Laundry', 'breakfast',
'Dinner', 'Sweets',
       'Kirana', 'Ice cream', 'curd', 'Biscuits', 'Rajgira
ladu',
       'Navratri', 'train', 'Tea', 'flour mill', 'Appliances',
       'home decor', 'grooming', 'Health', 'Clothing',
'clothes', 'Home',
       'chocolate', 'Medicine', 'Eating out', 'Movie',
'vegetables',
       'fruits', 'Potato', 'Onions', 'Taxi', 'Hardware',
'Eggs', 'Bread',
       'Petrol', 'Hospital', 'Mahanagar Gas', 'Lab Tests',
'Bus',
       'Travels', 'Kitchen', 'Footwear', 'Entry Fees',
'gadgets',
       'Accessories', 'misc', 'Stationary', 'Newspaper',
'Toiletries',
       'Bike', 'beverage', 'makeup', 'Books', 'Holi',
'Courier',
       'Leisure', 'Updation', 'Amazon Prime', 'Edtech Course',
'Hotstar',
       'Diwali', 'Wifi Internet Service', 'Trip', 'Furniture',
'Water',
       'Cable TV', 'medicine', 'Mutual fund', 'Public Provident
Fund',
       'ropeway', 'RD', 'LIC', 'Saloon', 'gift',
'Rakshabandhan',
       'exam fee', 'Kindle unlimited', 'OTT Platform', 'School
supplies',

       'Audible', 'Makeup'], dtype=object)
```

```
In [13]:
plt.figure(figsize = (12,8))
sns.countplot(data = df, x = "Subcategory", order =
df["Subcategory"].value_counts().iloc[:10].index)
plt.xticks(rotation = 90)
plt.show()
```



In [14]:

```
sns.countplot(data = df, x = "Income/Expense");
```



In [15]:

```
df["Note"].nunique()
```

Out[15]:

```
1057
```

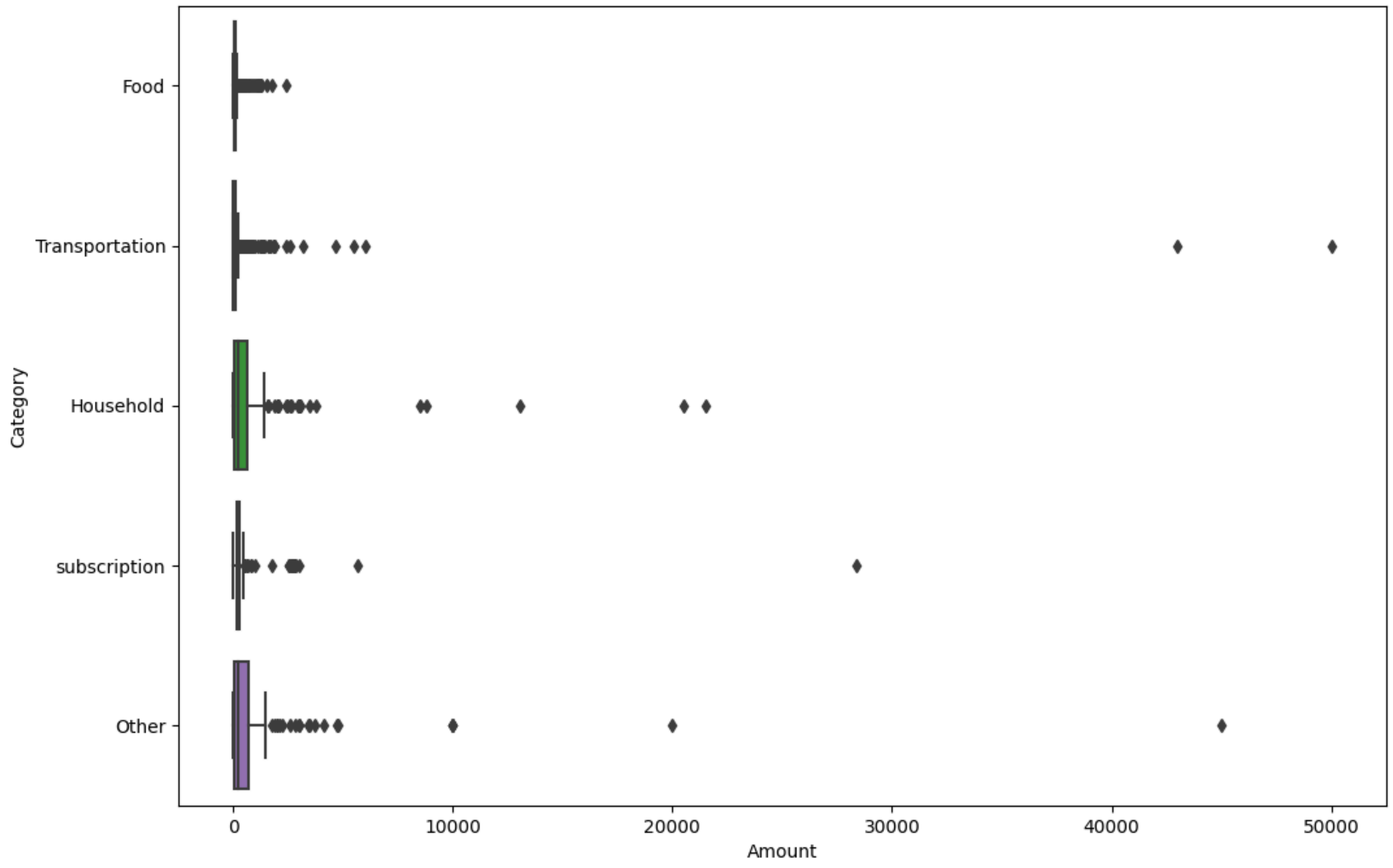
In [16]:

```
df["Currency"].value_counts()
```

Out[16]:

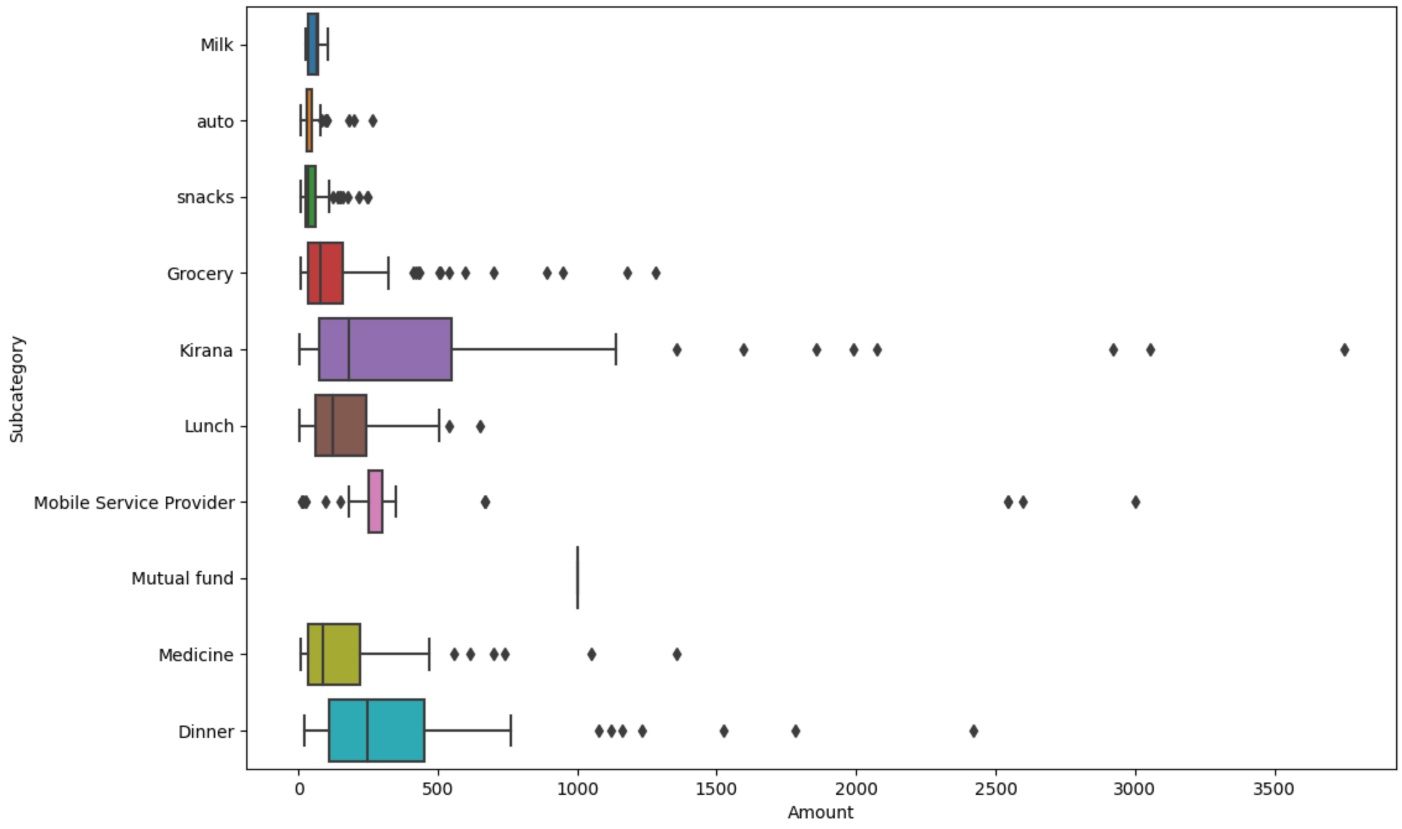
```
Currency
INR    2461

Name: count, dtype: int64
```

```
In [17]:
plt.figure(figsize = (12,8))
sns.boxplot(data = df, x = "Amount", y = "Category", order =
df["Category"].value_counts().iloc[:5].index)
plt.show()
```

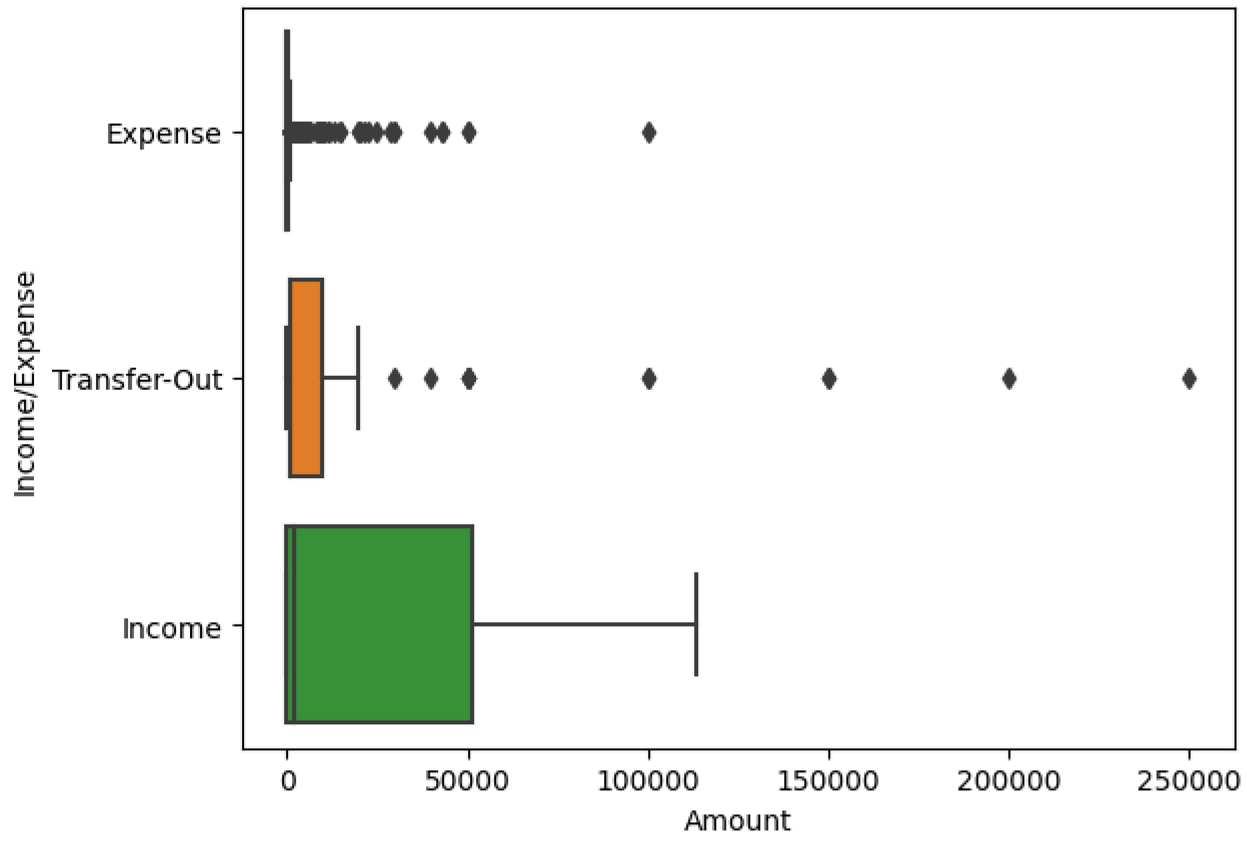


```
In [18]:
```

```
plt.figure(figsize = (12,8))
sns.boxplot(data = df, x = "Amount", y = "Subcategory", order =
df["Subcategory"].value_counts().iloc[:10].index, )
plt.show()
```

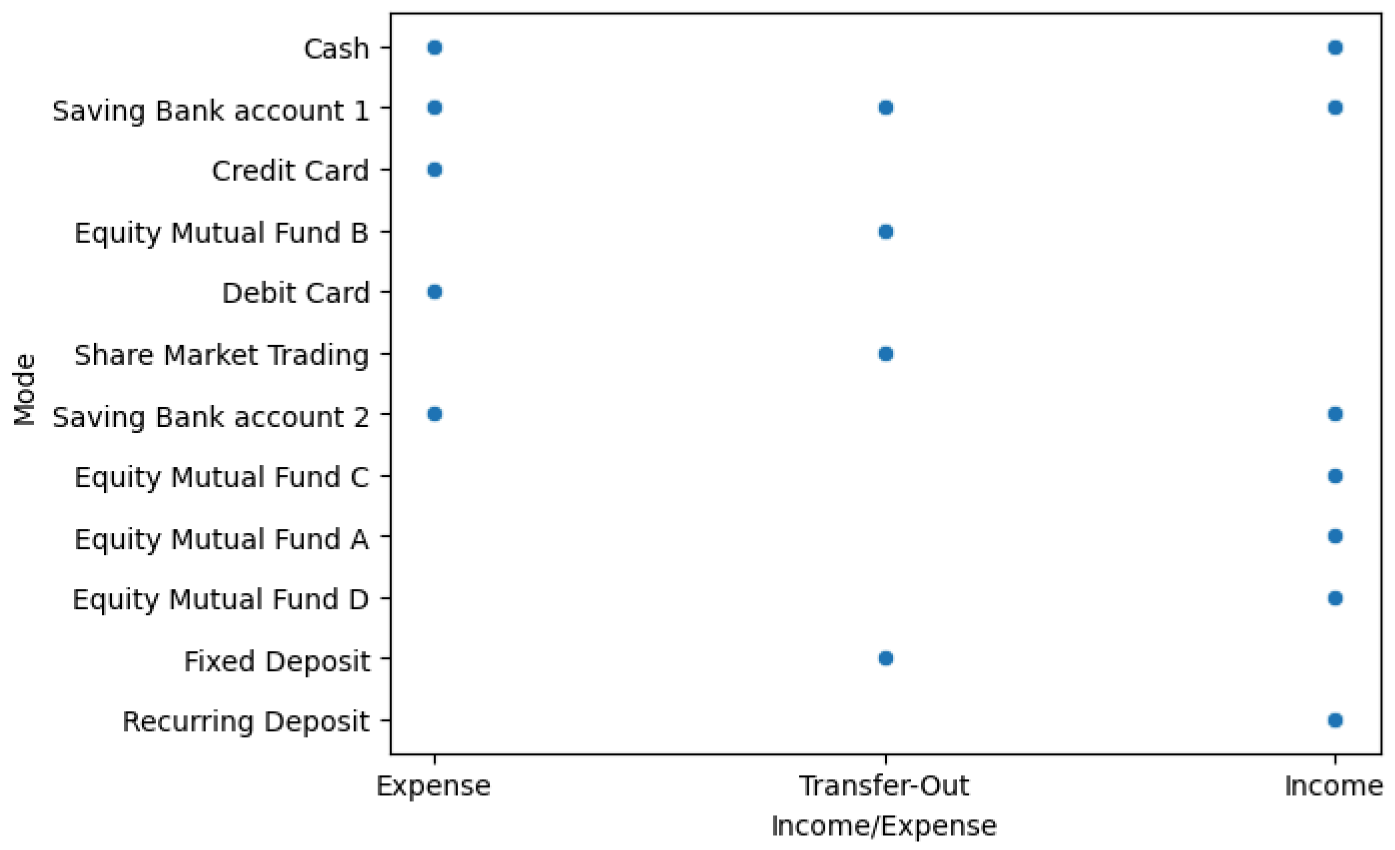


```
In [19]:
sns.boxplot(data = df, x = "Amount", y = "Income/Expense");
```



```
In [20]:
```

```
sns.scatterplot(data = df,x = "Income/Expense", y = "Mode",);
```



```
In [ ]:
```

Reference link