Data-Driven Strategies for Enhancing the Retail Transaction Experience

Data Analysis

Y. Adithya Vardhan Reddy

The Steps of Data Analysis: From Raw Data to Insightful Reports

Data analysis is a fascinating journey that transforms raw data into actionable insights. Here's a breakdown of the key steps involved:

1. Data Collection:

Identify your source: Where will you get your data? Databases, surveys, APIs, web scraping, or manual entry are some options.
Choose the right format: Ensure your data is in a format compatible with your analysis tools (e.g., CSV, Excel, JSON).
Gather relevant data: Collect all data points necessary to answer your research questions.

2. Data Cleaning:

Correct errors: Identify and fix typos, inconsistencies, and formatting issues.
Handle missing values: Decide how to address missing data points (e.g., imputation, exclusion).
Standardize data formats: Ensure consistent data types, units, and labels across variables.

3. Data Preparation:

Data transformation: Apply transformations to make your data analysis-friendly (e.g., scaling, encoding categorical variables).
Data selection: Choose the relevant subset of data based on your research questions.

4. Exploratory Data Analysis:

Dive deeper: Explore your data through visualizations, descriptive statistics, and hypothesis testing.
Identify patterns and trends: Look for relationships, outliers, and unexpected behaviors.
Formulate hypotheses: Based on your findings, refine your research questions and potential outcomes.

5. Data Visualization:

Tell a story with data: Create clear, informative, and engaging visualizations to communicate your findings.
Choose the right chart type: Bar charts, line graphs, scatter plots, and heatmaps are some common options.
Highlight key insights: Use visual cues to draw attention to important patterns and relationships.

6. Reporting:

Summarize your findings: Write a concise and informative report explaining your analysis, results, and conclusions.
Interpret your findings: Explain what your data means in the context of your research questions.
Communicate effectively: Tailor your report to your audience, using clear language and relevant visuals.

1. Data Collection:

- Loaded two datasets, Retail_Data_Response.csv and Retail_Data_Transactions.csv, using pd.read_csv.
- Displayed the first few rows of each dataset using df1.head() and df2.head().

Out[5]:

	customer_id	response
0	CS1112	0
1	CS1113	0
2	CS1114	1
3	CS1115	1
4	CS1116	1

Out[4]:

	customer_id	trans_date	tran_amount
0	CS5295	11-Feb-13	35
1	CS4768	15-Mar-15	39
2	CS2122	26-Feb-13	52
3	CS1217	16-Nov-11	99
4	CS1850	20-Nov-13	78

2. Data Cleaning:

- Checked data types and missing values in both datasets using df1.info() and df2.info().
- Merged the two datasets using the "customer_id" column with an inner join.
- Removed duplicate rows and dropped any remaining missing values using df = df.drop_duplicates().dropna().
- Checked the summary statistics of the cleaned dataset using df.describe().

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6884 entries, 0 to 6883
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   customer_id  6884 non-null   object
 1   response     6884 non-null   int64 
dtypes: int64(1), object(1)
memory usage: 107.7+ KB
```

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 125000 entries, 0 to 124999
Data columns (total 3 columns):
 #   Column       Non-Null Count   Dtype 
---  ------       --------------   ----- 
 0   customer_id  125000 non-null  object
 1   trans_date   125000 non-null  object
 2   tran_amount  125000 non-null  int64 
dtypes: int64(1), object(2)
memory usage: 2.9+ MB
```

```
df = df1.merge(df2, on="customer_id", how="inner")
```

	customer_id	response	trans_date	tran_amount
0	CS1112	0	14-Jan-15	39
1	CS1112	0	16-Jul-14	90
2	CS1112	0	29-Apr-14	63
3	CS1112	0	04-Dec-14	59
4	CS1112	0	08-Apr-12	56
...	...	...	...	...
124964	CS9000	0	12-May-12	53
124965	CS9000	0	08-May-14	20
124966	CS9000	0	28-Feb-15	34
124967	CS9000	0	01-Jun-12	37
124968	CS9000	0	11-Dec-12	49

124969 rows × 4 columns

3. Data Preparation:

- Checked data types of the cleaned dataset using df.dtypes.
- Displayed the last few rows of the dataset using df.tail().
- Checked for null values in the dataset using df.isnull().sum().
- Converted "trans_date" to datetime and "response" to 'int64' using pd.to_datetime and astype('int64') respectively.
- Standardized numerical columns using StandardScaler.

```
df.isnull().sum()
```

```
customer_id    0
response       0
trans_date     0
tran_amount    0
dtype: int64
```

```
num = df.select_dtypes(include=['number']).columns
num
```

```
Index(['response', 'tran_amount'], dtype='object')
```

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

```
6884
```

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

```
customer_id
CS4424    39
CS4320    38
CS3799    36
CS2620    35
CS3013    35
          ..
CS7224     4
CS7333     4
CS8559     4
CS8504     4
CS7716     4
Name: count, Length: 6884, dtype: int64
```

```
df["customer_id"].value_counts().count()
```

```
6884
```

```
scaler = StandardScaler()
scaler.fit(df[num])
df[num] = scaler.transform(df[num])
df[num]
```

	response	tran_amount
0	-0.352926	-1.137176
1	-0.352926	1.093804
2	-0.352926	-0.087303
3	-0.352926	-0.262282
4	-0.352926	-0.393516
...	...	...
124964	-0.352926	-0.524750
124965	-0.352926	-1.968325
124966	-0.352926	-1.355899
124967	-0.352926	-1.224665
124968	-0.352926	-0.699729

124963 rows × 2 columns

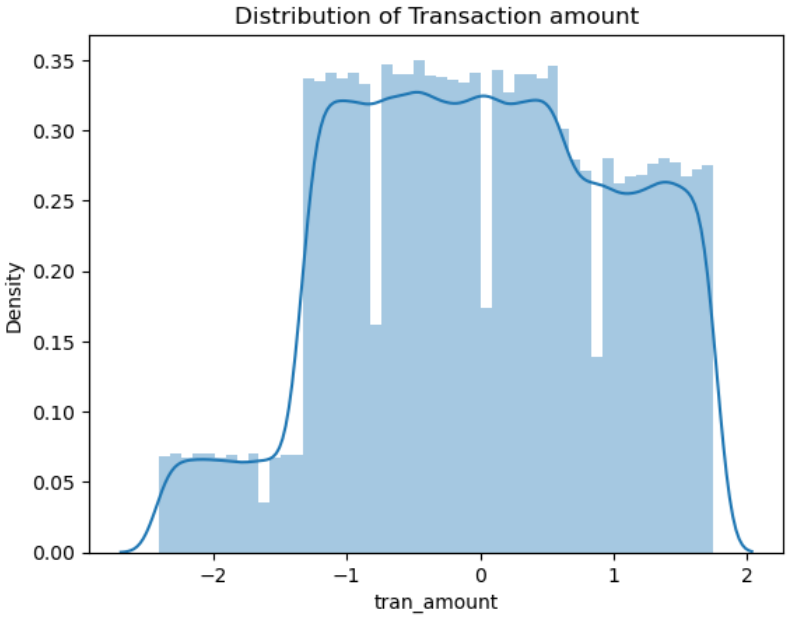
4. Data Analysis

- Explored the distribution of "tran_amount" and "response" using sns.distplot.
- Visualized the distribution of "tran_amount" using a boxplot and "response" using a histogram.
- Created a pair plot to visualize relationships between numerical variables.
- Generated a bar plot of the top 5 customers with the highest number of orders.
- Analyzed monthly sales by grouping data by month and summing transaction amounts.

Exploring the Transaction Amount Distribution

Objective: To visualize the frequency distribution of transaction amounts in your dataset. This will reveal if the amounts are clustered around certain values, skewed towards higher or lower values, or follow a specific distribution like normal or exponential.

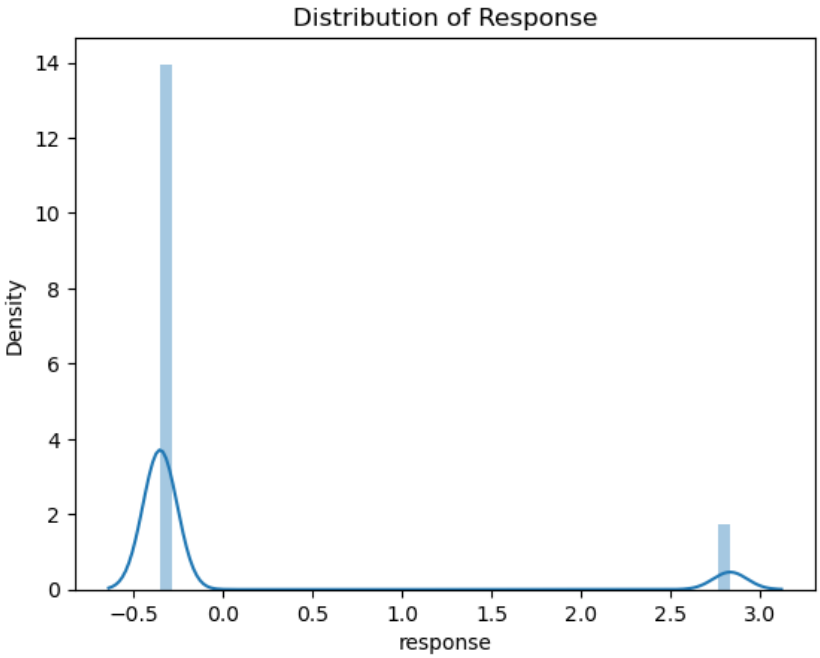
Description: This graph will likely be a histogram or density plot, showing the number of transactions (or density) at different transaction amount levels. Look for features like central tendency (peak), skewness (longer tail towards one side), and outliers.



Unveiling Response Time Distribution

Objective: To analyze the distribution of response times in your dataset. This could be related to customer support response times, order processing times, or any other relevant response variable.

Description: Similar to the transaction amount graph, this will likely be a histogram or density plot showing the frequency of responses at different time intervals. Analyze the spread of response times, identify any bottlenecks or peaks, and compare response times across different categories

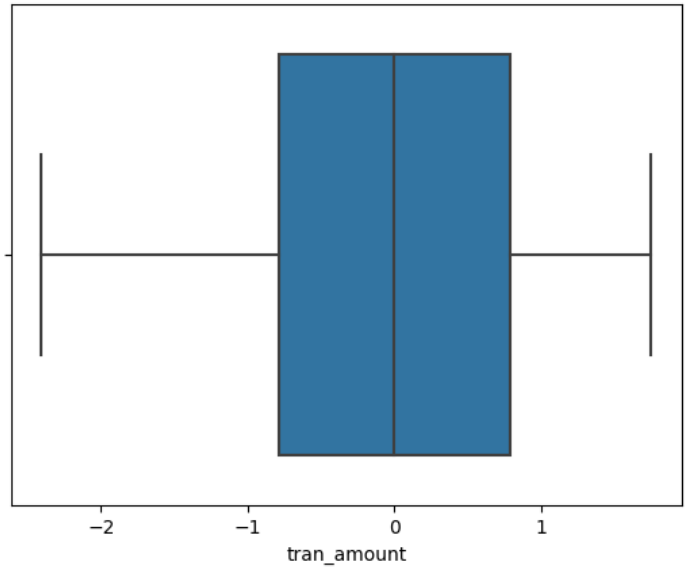


.

Distribution of Transaction Amount in Retail Transactions

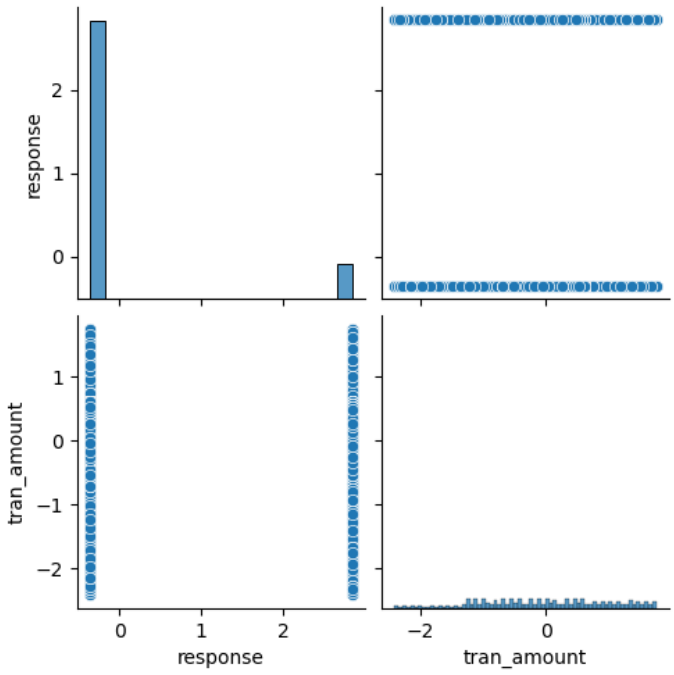
Objective: To visualize the distribution of transaction amounts in the retail transactions dataset, including the median, quartiles, outliers, and range.

Description: The boxplot shows the distribution of transaction amounts in the retail transactions dataset. The box represents the middle 50% of the data (the interquartile range), with the line in the middle of the box representing the median. The whiskers extend from the box to the minimum and maximum values of the data that are not considered outliers. Outliers are shown as individual points beyond the whiskers.



Unveiling the Relationship: Transaction Amount vs. Response Time

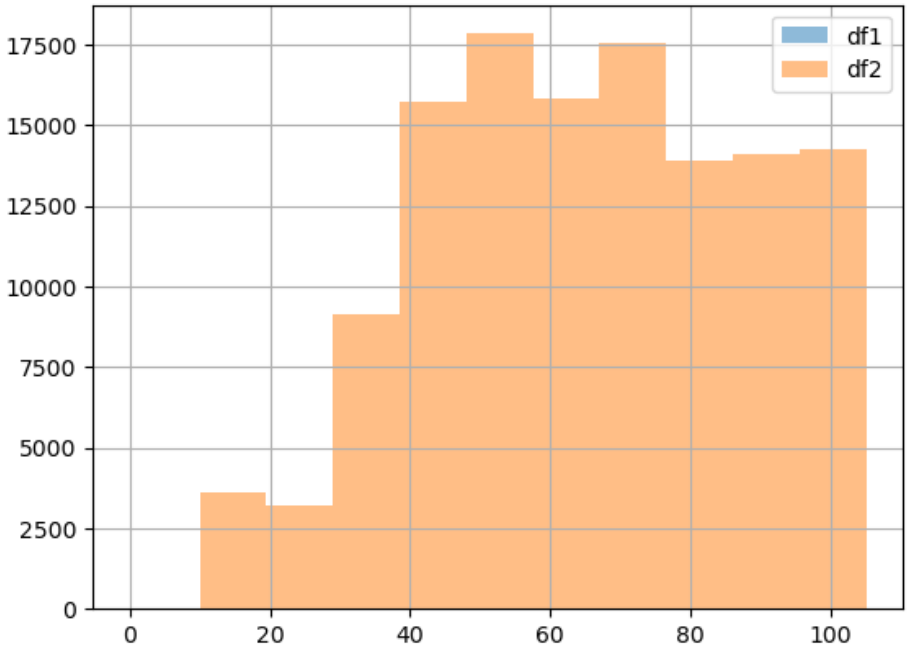
Objective: To investigate if there is a correlation between transaction amount and response time.
Description: A pair-plot displays scatter plots of each variable against every other variable in a matrix format. In this case, the scatter plot will show how response times are distributed across different transaction amounts. Look for any trends or patterns, such as longer response times for higher transaction amounts.



Comparing Price Distributions Across Two Data Sets:

Objective: To visually compare the price distributions of two different datasets. This could be useful for analyzing price changes over time, comparing prices between different product categories, or identifying price outliers.

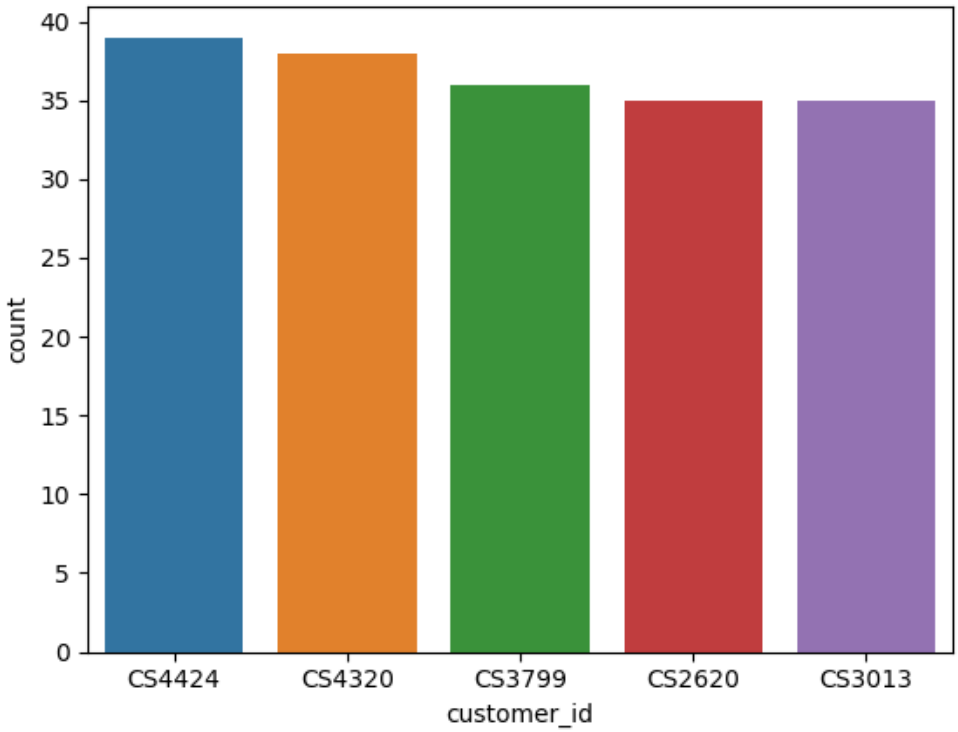
Description: This could be a side-by-side histogram or density plot, or even overlaid boxplots, showing the distribution of prices in each dataset. Compare the central tendency, spread, and shape of the distributions to understand the similarities and differences in pricing across the datasets.



Top Spenders Revealed: Customers with the Most Orders:

Objective: To identify the customers who have placed the most orders in your dataset.

Description: A bar graph will show the number of orders for each customer, sorted from highest to lowest. This helps identify your most loyal customers, analyze their buying behavior, and target them with relevant marketing campaigns.



Reporting:

- Provided visualizations with titles for better understanding of the data.
- Displayed the highest number of orders for the top 5 customers using a bar plot.
- Presented monthly sales with a bar plot.