```
# IMPORTING ALL THE NECESSARY LIBRARIES AND MODULES WHICH CAN BE USED IN THIS PROJECT

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, confus
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
import xgboost as xgb
from sklearn.neural_network import MLPClassifier
```

```
df = pd.read_csv('/content/blood.csv')
df
```

	Recency	Frequency	Monetary	Time	Class
0	2	50	12500	99	1
1	0	13	3250	28	1
2	1	17	4000	36	1
3	2	20	5000	45	1
4	1	24	6000	77	0
...	...	...	...	...	...
743	23	2	500	38	0
744	21	2	500	52	0
745	23	3	750	62	0
746	39	1	250	39	0
747	72	1	250	72	0

748 rows × 5 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.head()
```

	Recency	Frequency	Monetary	Time	Class
0	2	50	12500	99	1
1	0	13	3250	28	1
2	1	17	4000	36	1
3	2	20	5000	45	1
4	1	24	6000	77	0

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.shape
```

```
(748, 5)
```

```
df.columns
```

```
Index(['Recency', 'Frequency', 'Monetary', 'Time', 'Class'], dtype='object')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 748 entries, 0 to 747
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Recency    748 non-null    int64
 1   Frequency  748 non-null    int64
 2   Monetary   748 non-null    int64
 3   Time       748 non-null    int64
 4   Class      748 non-null    int64
dtypes: int64(5)
memory usage: 29.3 KB
```

```
df.describe()
```

	Recency	Frequency	Monetary	Time	Class
count	748.000000	748.000000	748.000000	748.000000	748.000000
mean	9.506684	5.516043	1378.676471	34.284759	0.237968
std	8.095396	5.841825	1459.826781	24.380307	0.426124
min	0.000000	1.000000	250.000000	2.000000	0.000000
25%	2.750000	2.000000	500.000000	16.000000	0.000000
50%	7.000000	4.000000	1000.000000	28.000000	0.000000
75%	14.000000	7.000000	1750.000000	50.000000	0.000000
max	74.000000	50.000000	12500.000000	99.000000	1.000000

```
df.nunique()
```

	0
Recency	31
Frequency	33
Monetary	33
Time	79
Class	2

dtype: int64

```
df['Class'].unique()
```

```
array([1, 0])
```

```
df['Class'].value_counts()
```

Class	count
0	570
1	178

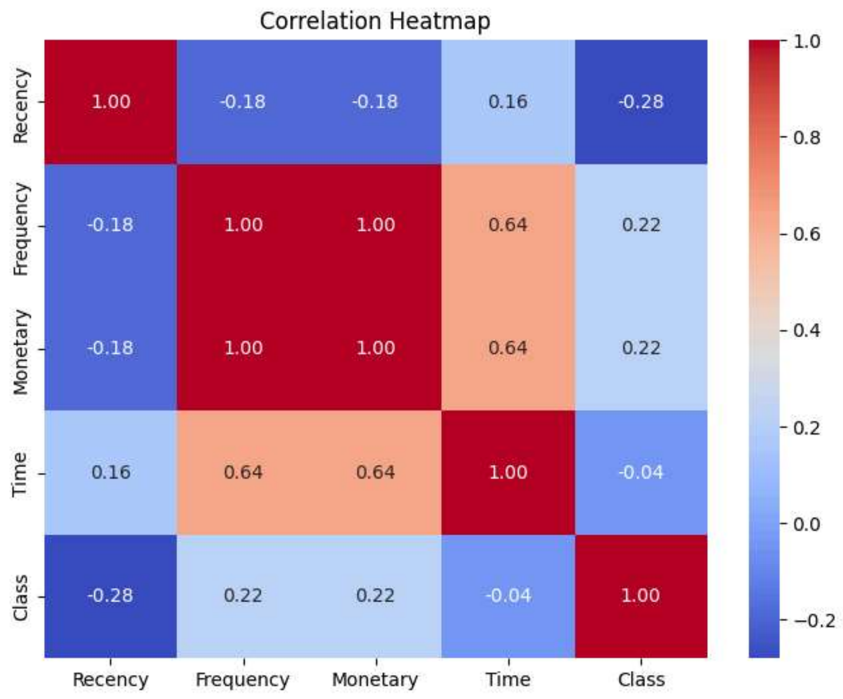
dtype: int64

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

```
              0
  Recency     0
  Frequency   0
  Monetary    0
  Time        0
  Class       0

dtype: int64
```

```
Start coding or generate with AI.
```

```python
# Heatmap of correlation between features
plt.figure(figsize=(8, 6))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```



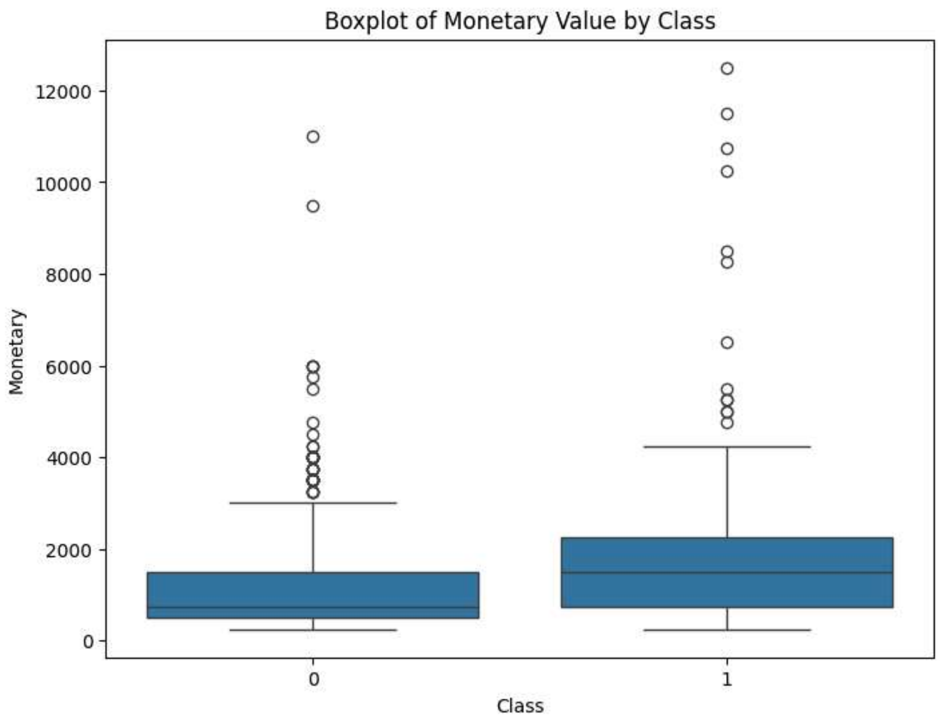
Shows correlations between Recency, Frequency, Monetary, Time, and Class.

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```python
# Boxplot of Monetary values by Class
plt.figure(figsize=(8, 6))
sns.boxplot(x='Class', y='Monetary', data=df)
plt.title('Boxplot of Monetary Value by Class')
plt.show()
```

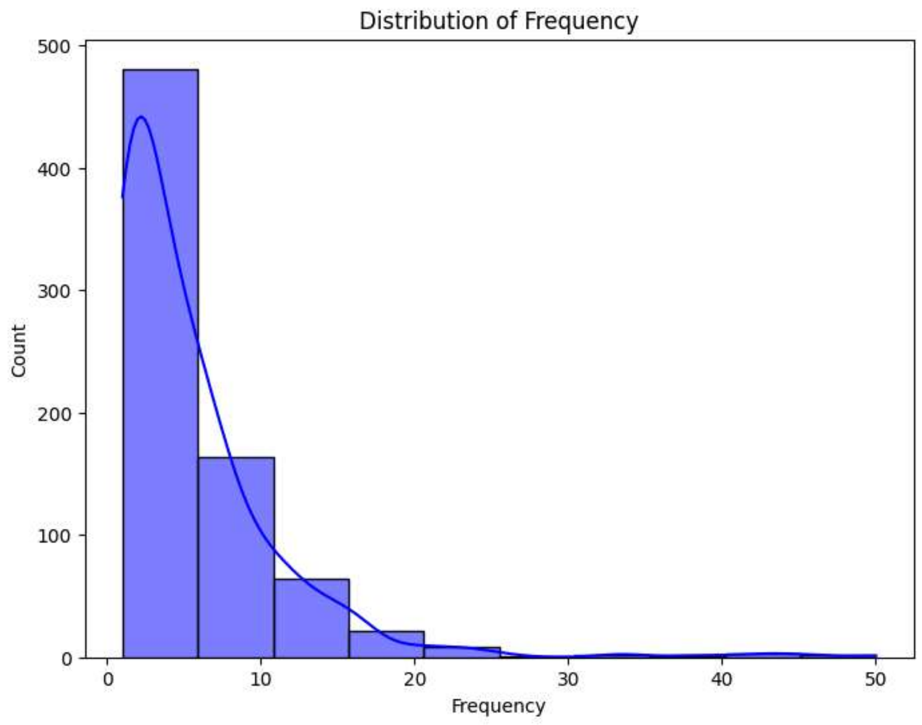


Displays the spread of Monetary values across different classes.

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
# Distribution plot for Frequency
plt.figure(figsize=(8, 6))
sns.histplot(df['Frequency'], kde=True, bins=10, color='blue')
plt.title('Distribution of Frequency')
plt.xlabel('Frequency')
plt.ylabel('Count')
plt.show()
```

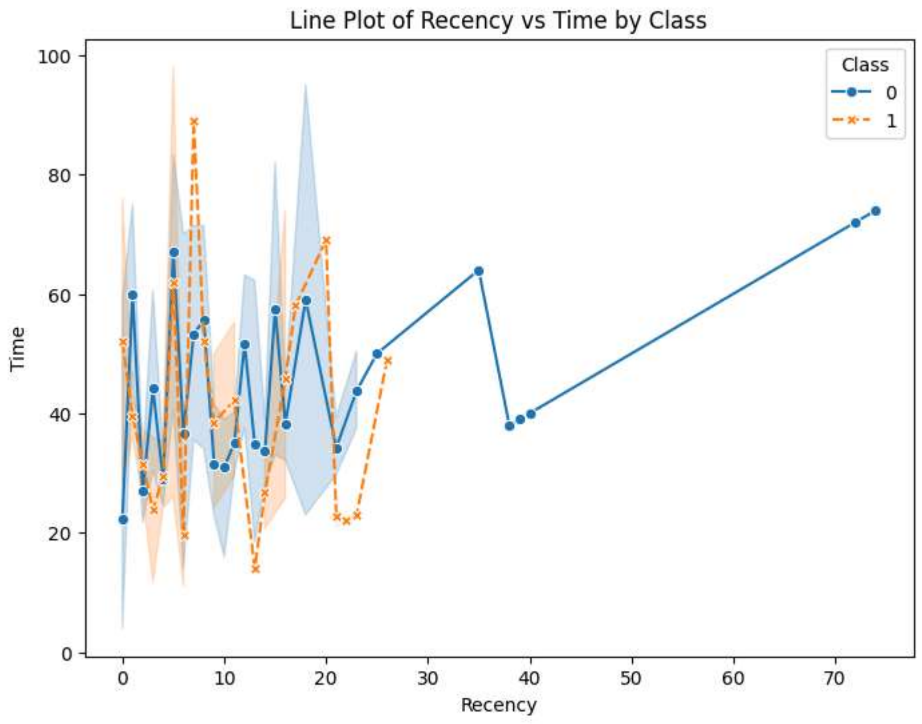


Shows the distribution of Frequency values in the dataset.

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
# Line plot for Recency vs Time
plt.figure(figsize=(8, 6))
sns.lineplot(x='Recency', y='Time', hue='Class', style='Class', markers=True, data=df)
plt.title('Line Plot of Recency vs Time by Class')
plt.show()
```

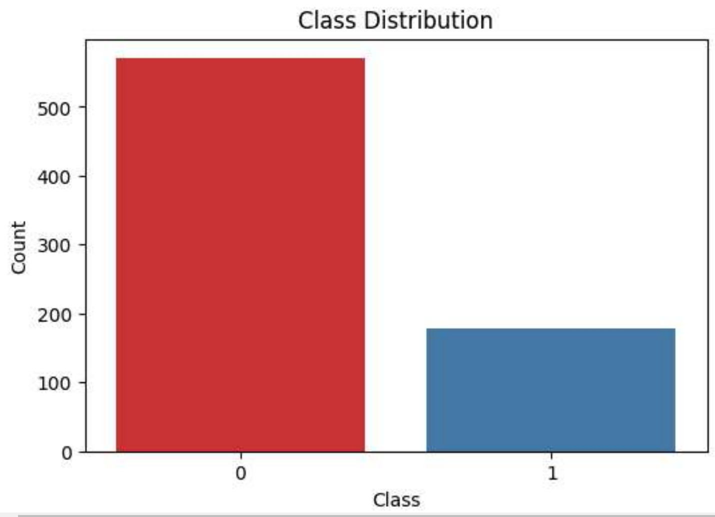


Visualizes the relationship between Recency and Time based on Class.

```
Start coding or generate with AI.
```

```
# Class distribution count plot
plt.figure(figsize=(6, 4))
sns.countplot(data=df, x='Class', palette='Set1')
plt.title('Class Distribution')
plt.xlabel('Class')
plt.ylabel('Count')
plt.show()

```

```
<ipython-input-46-356403bd7901>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.countplot(data=df, x='Class', palette='Set1')
```

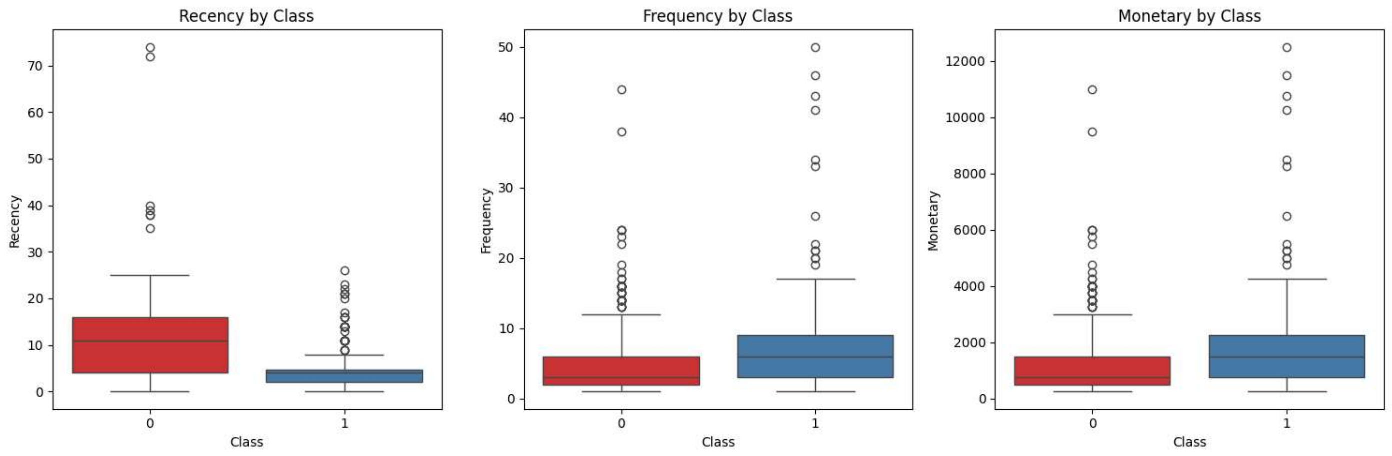


visualize how the data is distributed across different classes (0 and 1) using a countplot.

```
Start coding or generate with AI.
```

```
# Boxplot for Recency, Frequency, and Monetary by Class
fig, axes = plt.subplots(1, 3, figsize=(15, 5))

# Recency Boxplot by Class
sns.boxplot(x='Class', y='Recency', data=df, ax=axes[0], palette='Set1')
axes[0].set_title('Recency by Class')

# Frequency Boxplot by Class
sns.boxplot(x='Class', y='Frequency', data=df, ax=axes[1], palette='Set1')
axes[1].set_title('Frequency by Class')

# Monetary Boxplot by Class
sns.boxplot(x='Class', y='Monetary', data=df, ax=axes[2], palette='Set1')
axes[2].set_title('Monetary by Class')

plt.tight_layout()
plt.show()

```

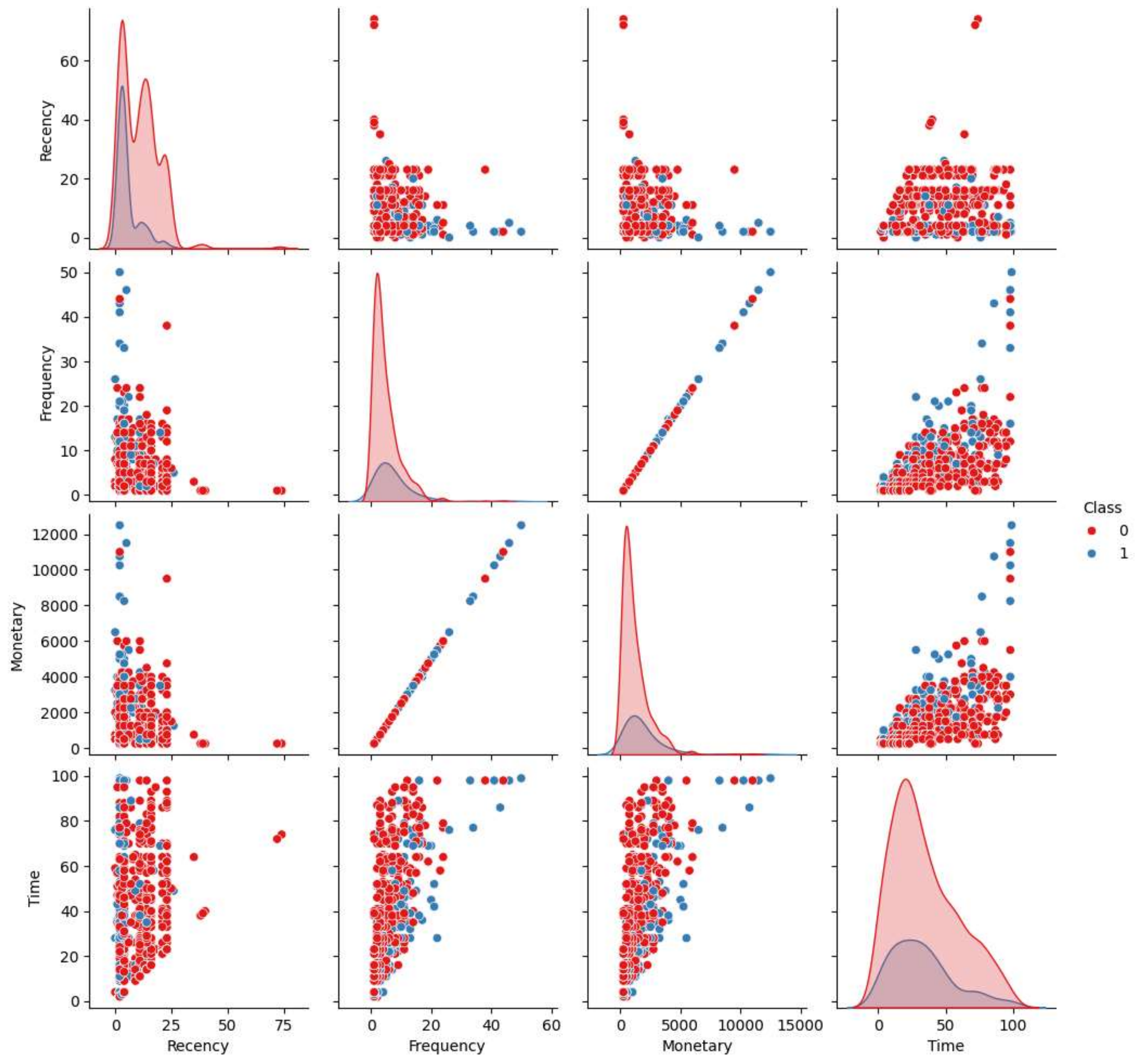
```
<ipython-input-47-6df76138b484>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.boxplot(x='Class', y='Recency', data=df, ax=axes[0], palette='Set1')
<ipython-input-47-6df76138b484>:9: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.boxplot(x='Class', y='Frequency', data=df, ax=axes[1], palette='Set1')
<ipython-input-47-6df76138b484>:13: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.boxplot(x='Class', y='Monetary', data=df, ax=axes[2], palette='Set1')
```



A boxplot can help identify outliers and the distribution of Recency, Frequency, and Monetary across the different classes (0 and 1).

```
Start coding or generate with AI.
```

```
# Pairplot for all numerical features, colored by 'Class'
sns.pairplot(df[['Recency', 'Frequency', 'Monetary', 'Time', 'Class']], hue='Class', palette='Set1')
plt.show()

```



```
Start coding or generate with AI.
```

Conclusion:

The analysis of Recency, Frequency, Monetary, Time, and Class reveals distinct patterns in customer behavior. Customers in Class 1, likely representing loyal or high-value segments, tend to have higher Monetary and Frequency values, as well as more recent interactions (Recency). The correlation analysis suggests that Recency is moderately linked to higher spending and frequent engagement. The boxplots highlight outliers, particularly in Monetary and Frequency, which can help identify high-impact customers. Overall, the visualizations suggest that more engaged customers (higher Recency, Frequency, and Monetary) are more likely to belong to Class 1, offering valuable insights for targeting and segmentation strategies

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```