```
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
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error
import warnings
warnings.filterwarnings('ignore')
```

```
import pandas as pd

df = pd.read_csv('Sales Transaction v.4a.csv')
df.head()
```

	TransactionNo	Date	ProductNo	ProductName	Price	Quantity	CustomerNo	Country
0	581482	12/9/2019	22485	Set Of 2 Wooden Market Crates	21.47	12.0	17490.0	United Kingdom
1	581475	12/9/2019	22596	Christmas Star Wish List Chalkboard	10.65	36.0	13069.0	United Kingdom
2	581475	12/9/2019	23235	Storage Tin Vintage Leaf	11.53	12.0	13069.0	United Kingdom
3	581475	12/9/2019	23272	Tree T-Light Holder Willie Winkie	10.65	12.0	13069.0	United Kingdom
4	581475	12/9/2019	23239	Set Of 4 Knick Knack Tins Poppies	11.94	6.0	13069.0	United Kingdom

```
import pandas as pd

data = pd.read_csv('Sales Transaction v.4a.csv')

print(data.head())
```

```
   TransactionNo       Date ProductNo                         ProductName  \
0         581482  12/9/2019     22485        Set Of 2 Wooden Market Crates
1         581475  12/9/2019     22596  Christmas Star Wish List Chalkboard
2         581475  12/9/2019     23235             Storage Tin Vintage Leaf
3         581475  12/9/2019     23272    Tree T-Light Holder Willie Winkie
4         581475  12/9/2019     23239    Set Of 4 Knick Knack Tins Poppies

   Price  Quantity  CustomerNo         Country
0  21.47        12     17490.0  United Kingdom
1  10.65        36     13069.0  United Kingdom
2  11.53        12     13069.0  United Kingdom
3  10.65        12     13069.0  United Kingdom
4  11.94         6     13069.0  United Kingdom
```

```
data.rename(columns={
    'TransactionNo': 'InvoiceNo',
    'Date': 'InvoiceDate',
    'ProductNo': 'ProductID',
    'Price': 'UnitPrice',
    'Quantity': 'Quantity',
    'CustomerNo': 'CustomerID',
    'Country': 'Country'
}, inplace=True)


data['Quantity'] = pd.to_numeric(data['Quantity'], errors='coerce')
data['UnitPrice'] = pd.to_numeric(data['UnitPrice'], errors='coerce')

data = data.dropna(subset=['CustomerID'])

data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'], errors='coerce')

data['TotalPrice'] = data['Quantity'] * data['UnitPrice']

print(data.isnull().sum())

print(data.head())
```

```
InvoiceNo      0
InvoiceDate    0
ProductID      0
ProductName    0
UnitPrice      0
```

```
Quantity       0
CustomerID     0
Country        0
TotalPrice     0
dtype: int64
  InvoiceNo InvoiceDate ProductID                          ProductName  \
0    581482  2019-12-09     22485        Set Of 2 Wooden Market Crates
1    581475  2019-12-09     22596  Christmas Star Wish List Chalkboard
2    581475  2019-12-09     23235             Storage Tin Vintage Leaf
3    581475  2019-12-09     23272    Tree T-Light Holder Willie Winkie
4    581475  2019-12-09     23239    Set Of 4 Knick Knack Tins Poppies

   UnitPrice  Quantity  CustomerID         Country  TotalPrice
0      21.47        12     17490.0  United Kingdom      257.64
1      10.65        36     13069.0  United Kingdom      383.40
2      11.53        12     13069.0  United Kingdom      138.36
3      10.65        12     13069.0  United Kingdom      127.80
4      11.94         6     13069.0  United Kingdom       71.64
```

```
TODAY = data['InvoiceDate'].max() + pd.Timedelta(days=1)

rfm = data.groupby('CustomerID').agg({
    'InvoiceDate': lambda x: (TODAY - x.max()).days,  # Recency
    'InvoiceNo': 'count',                             # Frequency
    'TotalPrice': 'sum'                               # Monetary (Total Purchase)
})

rfm.rename(columns={
    'InvoiceDate': 'Recency',
    'InvoiceNo': 'Frequency',
    'TotalPrice': 'Monetary'
}, inplace=True)

rfm.head()
```

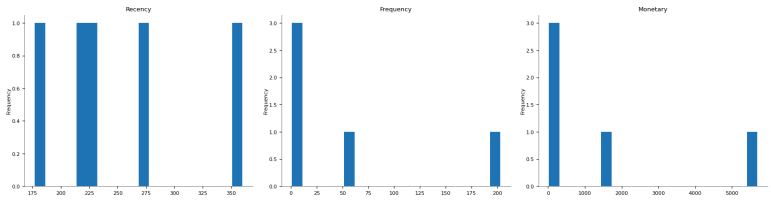
1 to 5 of 5 entries Filter

CustomerID	Recency	Frequency	Monetary
12004.0	228	56	1509.6
12006.0	219	1	24.76
12008.0	277	203	5689.57
12013.0	360	1	69.96000000000001
12024.0	177	5	149.51999999999998

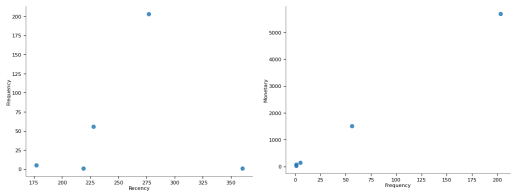
Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

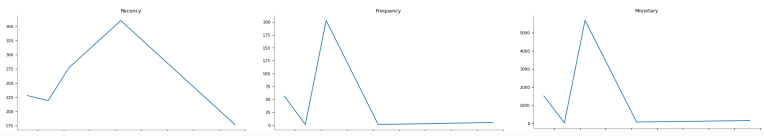
Distributions



2-d distributions



Values



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

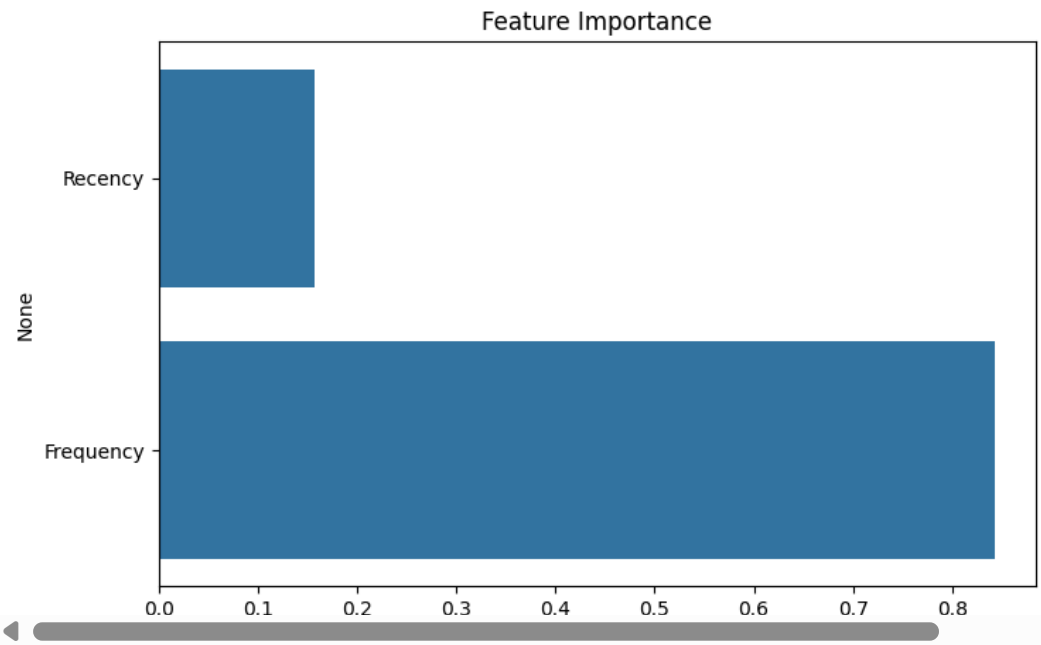
```
X = rfm[['Recency', 'Frequency']]

y = rfm['Monetary']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print("MAE (Mean Absolute Error):", mean_absolute_error(y_test, y_pred))
print("RMSE (Root Mean Squared Error):", np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
MAE (Mean Absolute Error): 12496.292228572464
RMSE (Root Mean Squared Error): 54880.6044020451
```

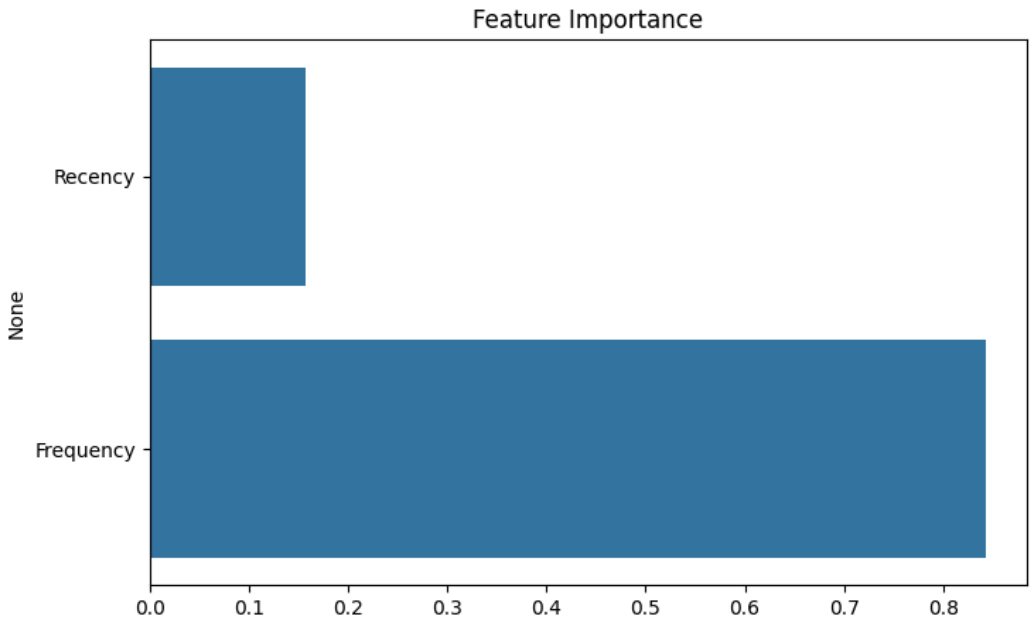
```
importances = model.feature_importances_
features = X.columns

plt.figure(figsize=(8,5))
sns.barplot(x=importances, y=features)
plt.title('Feature Importance')
plt.show()
```




```
predicted_ltv = pd.DataFrame({
    'CustomerID': X_test.index,
    'ActualLTV': y_test,
    'PredictedLTV': y_pred
})

predicted_ltv.to_csv('Predicted_Customer_LTV.csv', index=False)
print("Predicted LTV saved to CSV ✅")
```

```
Predicted LTV saved to CSV ✅
```

```
plt.figure(figsize=(8,5))
sns.barplot(x=importances, y=features)
plt.title('Feature Importance')
plt.show()
```




```
predicted_ltv.to_csv('Predicted_Customer_LTV.csv', index=False)
```

```
from google.colab import files
files.download('Predicted_Customer_LTV.csv')
```