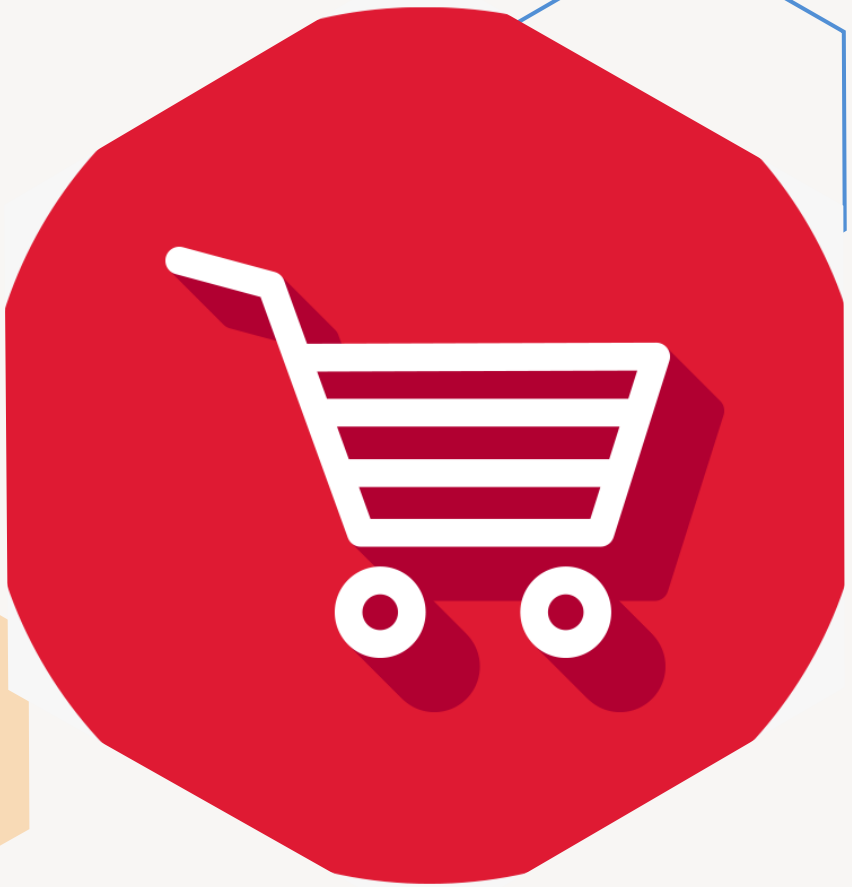
# ONLINE RETAIL ANALYSIS

BHUVANESWARI KAPULURU



# Agenda

Problem statement

Data

Clean data

Feature engineering

Visualisation

# Problem statement

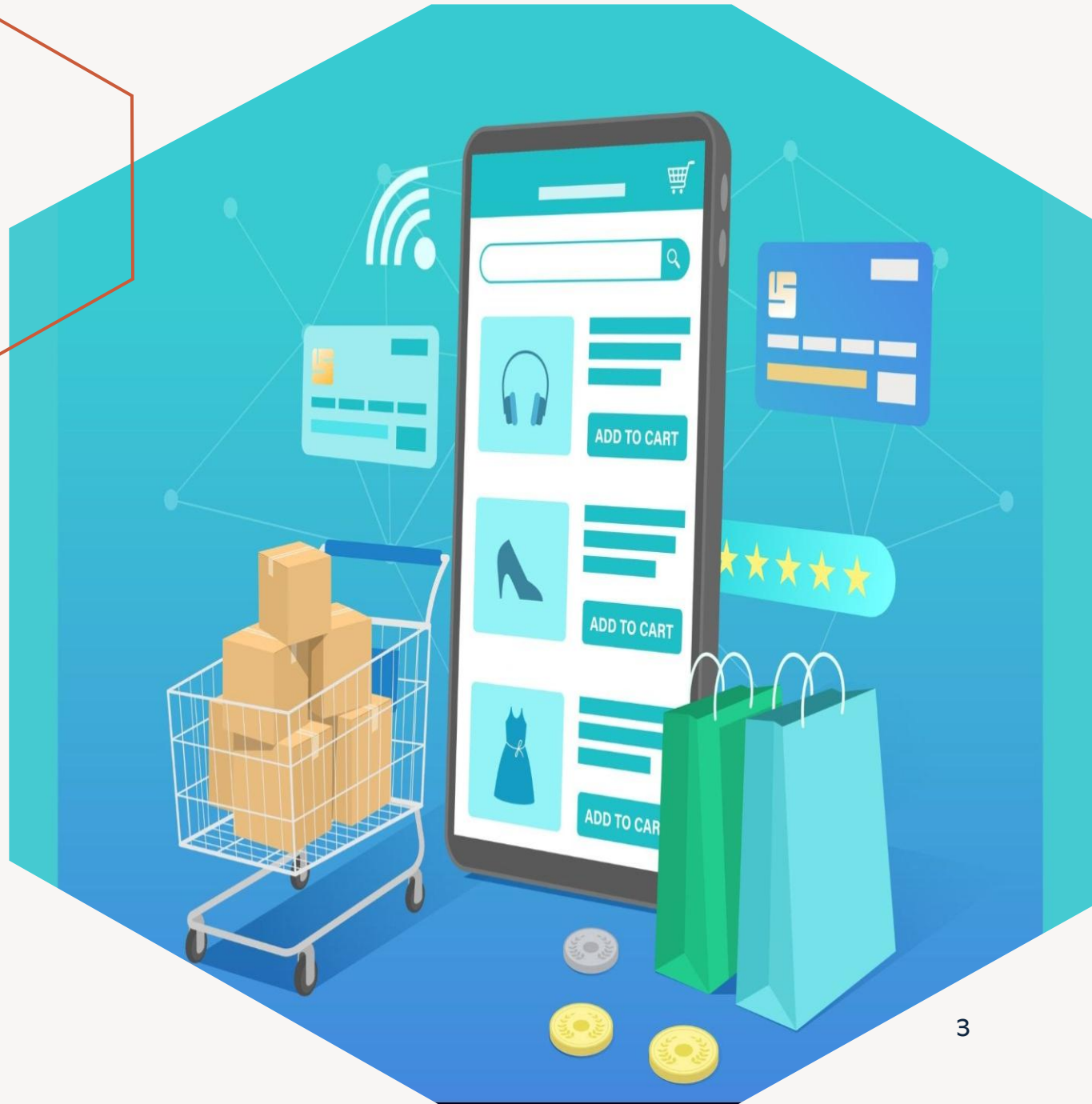
•The retail dataset contains thousands of transactions across multiple countries.

•Management lacks a clear understanding of:

•Customer behavior trends

•Purchase patterns across months and countries

•Customer retention and churn risks



# Steps In Data Analysis

1.Load data

2.Handle Missing Data

3.Feature engineering

4.visualisation

5.Rfm analysis

6.Churn analysis

# Data loading

# Data loading

```
[ ] !wget https://archive.ics.uci.edu/static/public/352/online+retail.zip
```

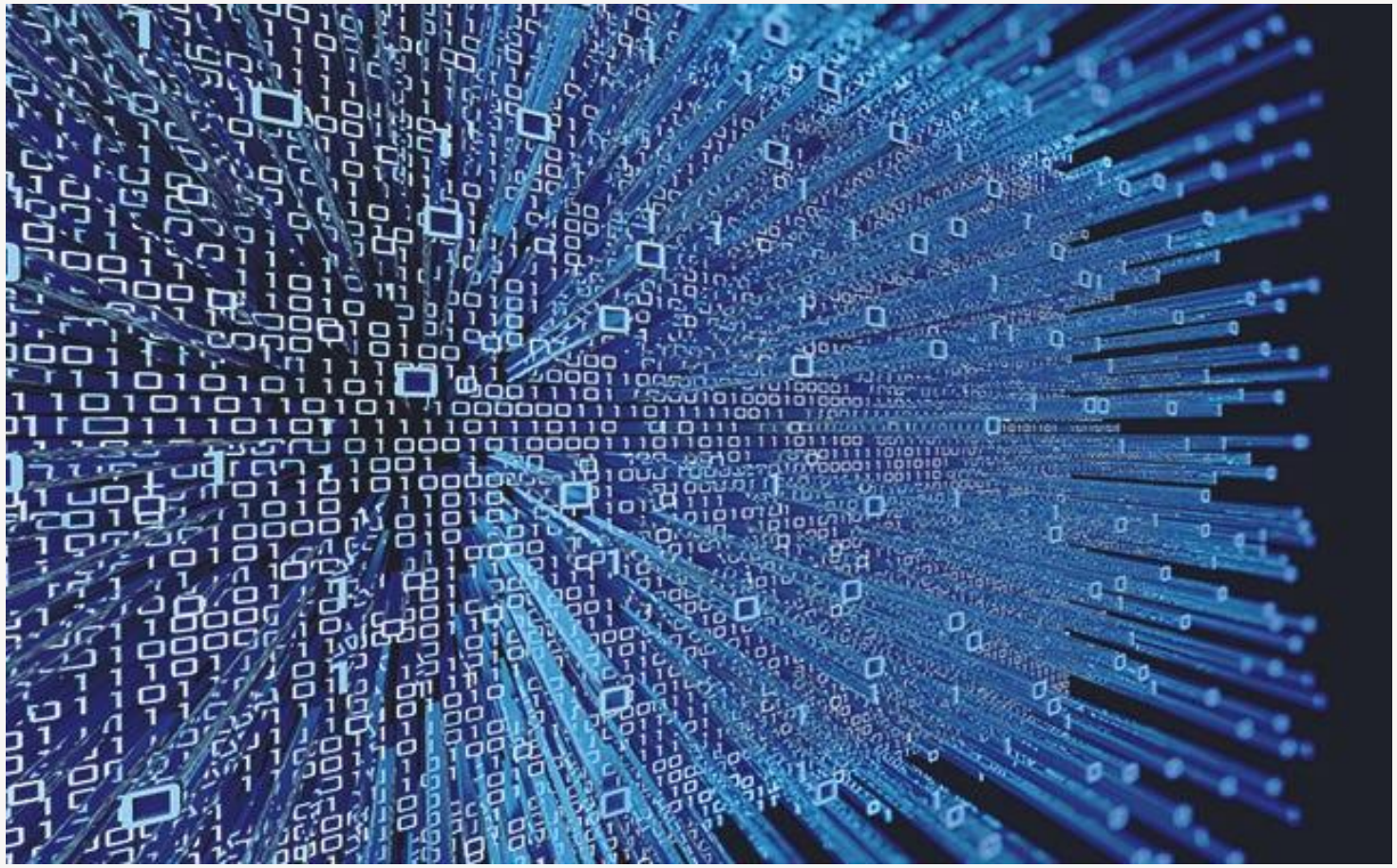
```
--2025-06-20 14:39:55--  https://archive.ics.uci.edu/static/public/352/online+retail.zip
Resolving archive.ics.uci.edu (archive.ics.uci.edu)... 128.195.10.252
Connecting to archive.ics.uci.edu (archive.ics.uci.edu)|128.195.10.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified
Saving to: ‘online+retail.zip’

online+retail.zip       [    <=>             ]  22.62M  31.4MB/s    in 0.7s

2025-06-20 14:39:56 (31.4 MB/s) - ‘online+retail.zip’ saved [23715478]
```

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

```
[ ] !unzip online+retail.zip
```

```
Archive:  online+retail.zip
 extracting: Online Retail.xlsx
```

# 2.Handling missing values



# Identifying null values

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

| | 0 |
|---|---|
| InvoiceNo | 0 |
| StockCode | 0 |
| Description | 1454 |
| Quantity | 0 |
| InvoiceDate | 0 |
| UnitPrice | 0 |
| CustomerID | 135080 |
| Country | 0 |

dtype: int64

```
df[["StockCode","Description"]][df.Description.isnull()==True]
```

| | StockCode | Description |
|---|---|---|
| 622 | 22139 | NaN |
| 1970 | 21134 | NaN |
| 1971 | 22145 | NaN |
| 1972 | 37509 | NaN |
| 1987 | 85226A | NaN |
| ... | ... | ... |
| 535322 | 84581 | NaN |
| 535326 | 23406 | NaN |
| 535332 | 21620 | NaN |
| 536981 | 72817 | NaN |
| 538554 | 85175 | NaN |

1454 rows × 2 columns

# Handled null values

```
d2.drop("Description_y",axis=1,inplace=True)
d2
```

| | InvoiceNo | StockCode | Description_x | Quantity | InvoiceDate | UnitPrice | CustomerID | Country | count |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 536365 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 08:26:00 | 2.55 | 17850.0 | United Kingdom | 2302 |
| **1** | 536373 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 09:02:00 | 2.55 | 17850.0 | United Kingdom | 2302 |
| **2** | 536375 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 09:32:00 | 2.55 | 17850.0 | United Kingdom | 2302 |
| **3** | 536390 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 64 | 2010-12-01 10:19:00 | 2.55 | 17511.0 | United Kingdom | 2302 |
| **4** | 536394 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 32 | 2010-12-01 10:39:00 | 2.55 | 13408.0 | United Kingdom | 2302 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **541792** | 548999 | 84743C | damages | -26 | 2011-04-05 14:34:00 | 0.00 | NaN | United Kingdom | 1 |
| **541793** | 554311 | 84743C | damages | -16 | 2011-05-23 15:28:00 | 0.00 | NaN | United Kingdom | 1 |
| **541794** | 543899 | 84803A | PINK ALLIUM ARTIFICIAL FLOWER | 3 | 2011-02-14 12:11:00 | 1.69 | NaN | EIRE | 1 |
| **541795** | 542731 | 84795C | OCEAN STRIPE HAMMOCK | 2 | 2011-01-31 15:27:00 | 7.95 | 13600.0 | United Kingdom | 1 |
| **541796** | 542784 | 84795C | OCEAN STRIPE HAMMOCK | 3 | 2011-02-01 10:04:00 | 0.00 | NaN | United Kingdom | 1 |

541797 rows × 9 columns

```
d2[d2.Description_x.isnull()==True]
d2.Description_x.isnull().sum()
d2.isnull().sum()
```

| | 0 |
|---|---|
| InvoiceNo | 0 |
| StockCode | 0 |
| Description_x | 0 |
| Quantity | 0 |
| InvoiceDate | 0 |
| UnitPrice | 0 |
| CustomerID | 134968 |
| Country | 0 |

# 3.Feature engineering

# Adding total price column

```
d2['total_price']=d2['Quantity']*d2['UnitPrice']
d2
```

| | InvoiceNo | StockCode | Description_x | Quantity | InvoiceDate | UnitPrice | CustomerID | Country | total_price |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 536365 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 08:26:00 | 2.55 | 17850.0 | United Kingdom | 15.30 |
| **1** | 536373 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 09:02:00 | 2.55 | 17850.0 | United Kingdom | 15.30 |
| **2** | 536375 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 09:32:00 | 2.55 | 17850.0 | United Kingdom | 15.30 |
| **3** | 536390 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 64 | 2010-12-01 10:19:00 | 2.55 | 17511.0 | United Kingdom | 163.20 |
| **4** | 536394 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 32 | 2010-12-01 10:39:00 | 2.55 | 13408.0 | United Kingdom | 81.60 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **541792** | 548999 | 84743C | damages | -26 | 2011-04-05 14:34:00 | 0.00 | NaN | United Kingdom | -0.00 |
| **541793** | 554311 | 84743C | damages | -16 | 2011-05-23 15:28:00 | 0.00 | NaN | United Kingdom | -0.00 |
| **541794** | 543899 | 84803A | PINK ALLIUM ARTIFICIAL FLOWER | 3 | 2011-02-14 12:11:00 | 1.69 | NaN | EIRE | 5.07 |
| **541795** | 542731 | 84795C | OCEAN STRIPE HAMMOCK | 2 | 2011-01-31 15:27:00 | 7.95 | 13600.0 | United Kingdom | 15.90 |
| **541796** | 542784 | 84795C | OCEAN STRIPE HAMMOCK | 3 | 2011-02-01 10:04:00 | 0.00 | NaN | United Kingdom | 0.00 |

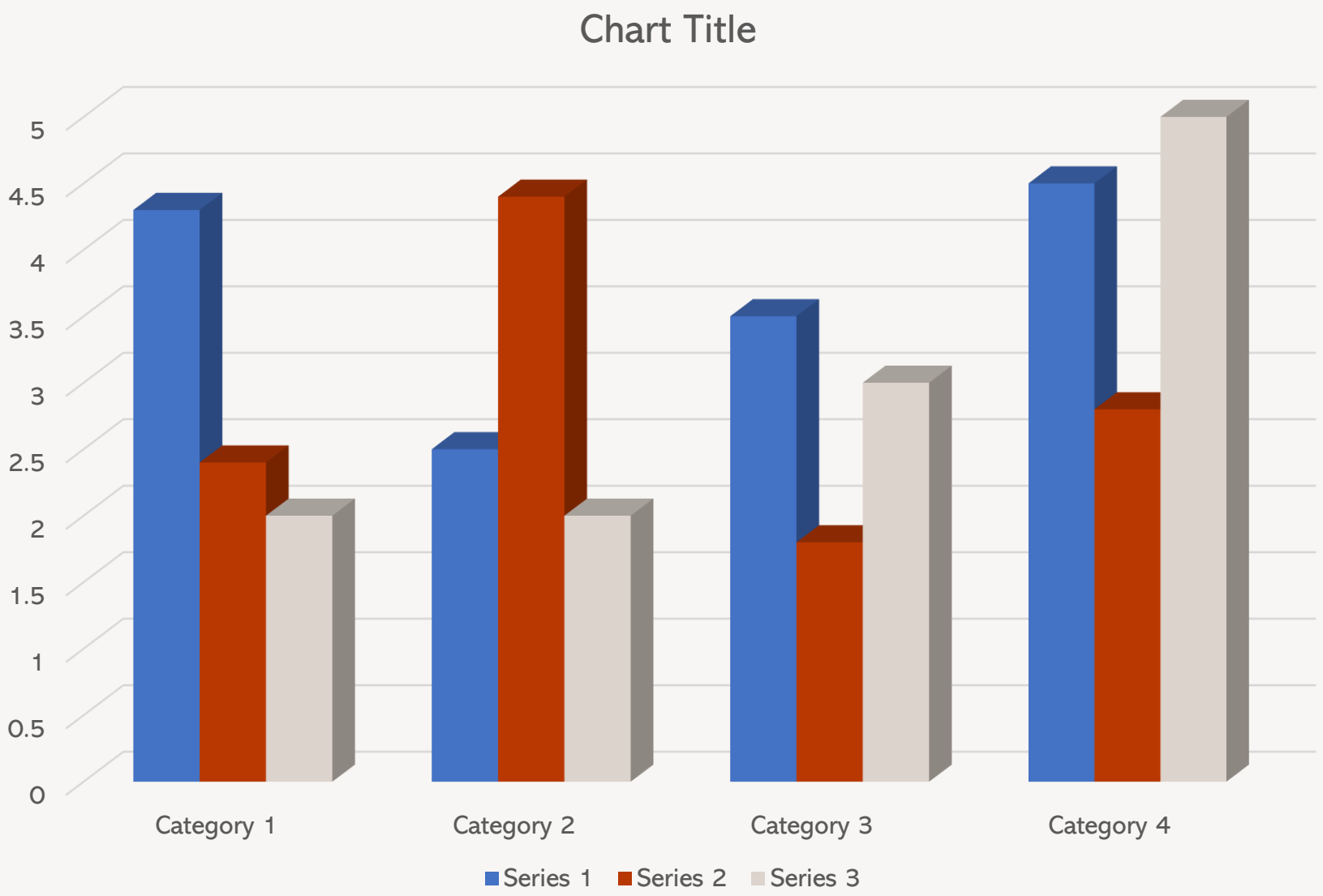
541797 rows × 9 columns

# Adding month column

```
d2['month']=d2['InvoiceDate'].dt.month
d2.head(3)
```

| | InvoiceNo | StockCode | Description_x | Quantity | InvoiceDate | UnitPrice | CustomerID | Country | total_price | month |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 536365 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 08:26:00 | 2.55 | 17850.0 | United Kingdom | 15.3 | 12 |
| 1 | 536373 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 09:02:00 | 2.55 | 17850.0 | United Kingdom | 15.3 | 12 |
| 2 | 536375 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 2010-12-01 09:32:00 | 2.55 | 17850.0 | United Kingdom | 15.3 | 12 |

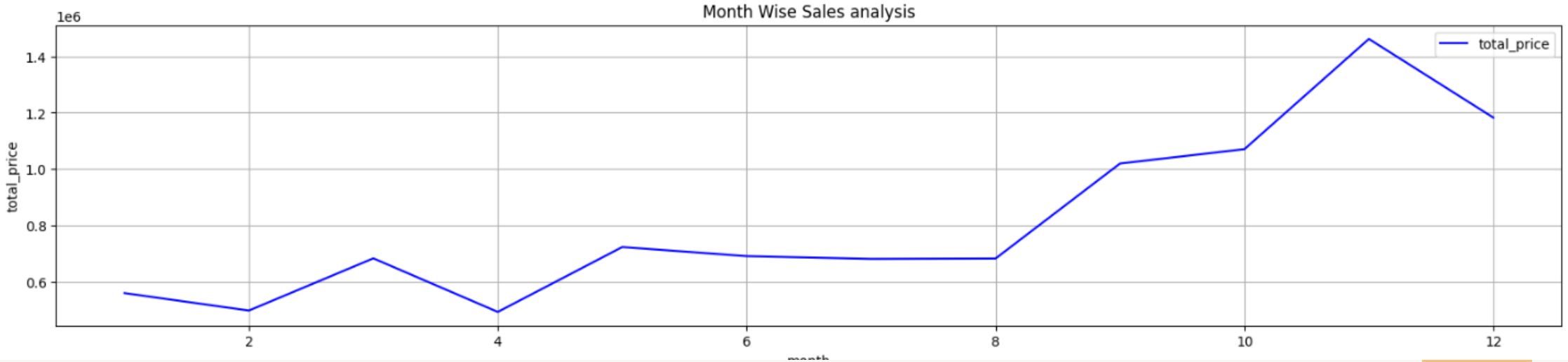
# 4.Visualisation



# Month wise sales analysis

```
from matplotlib import pyplot as plt
plt.figure(figsize=(20,4))
plt.plot(m.index,m.values,color="blue",label="total_price")
plt.title("Month Wise Sales analysis")
plt.xlabel("month")
plt.ylabel("total_price")
plt.grid()
plt.legend()
```

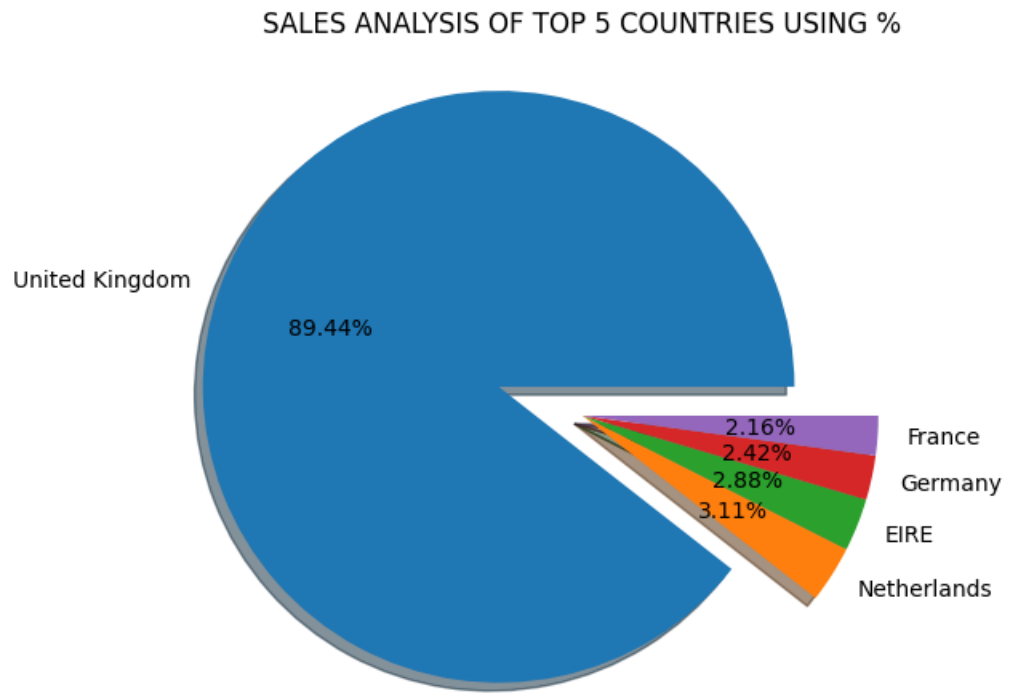
```
<matplotlib.legend.Legend at 0x7a8646b00b10>
```



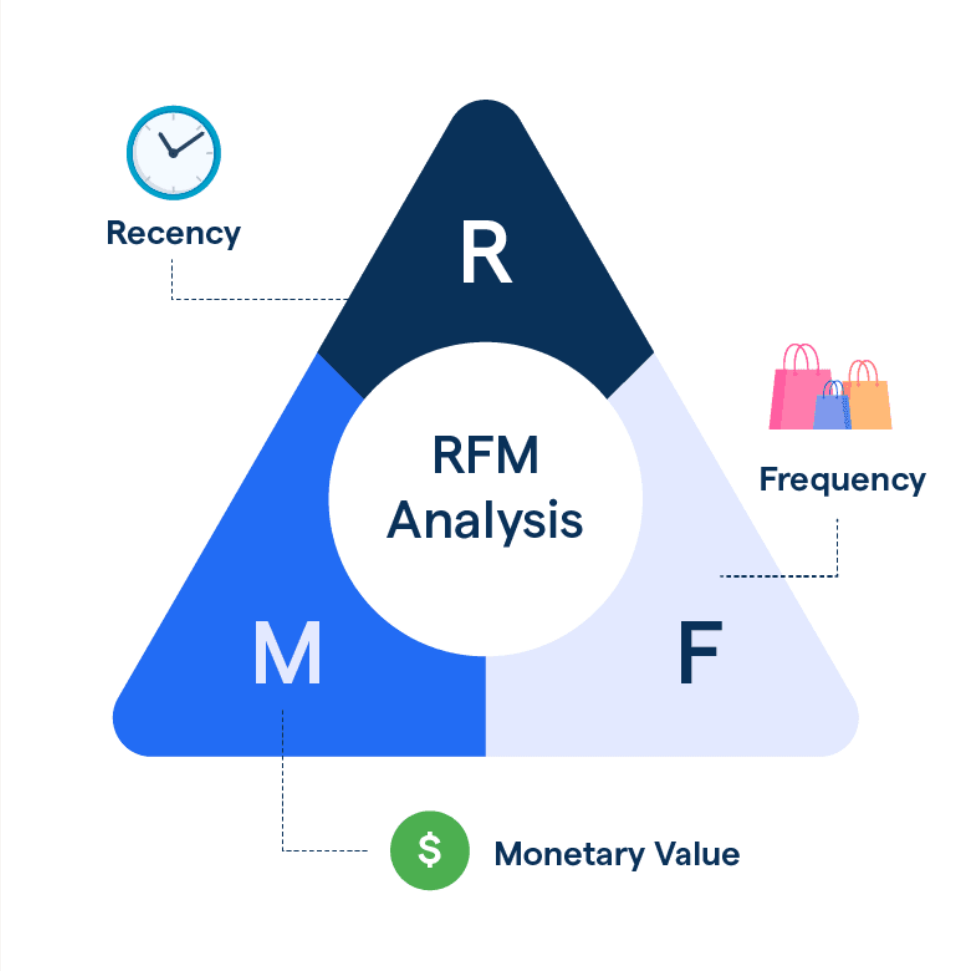
# Sales analysis of top 5 countries in %

```
max_con=d2.groupby("Country").total_price.sum().sort_values(ascending=False).head(5)
plt.figure(figsize=(12,6))
plt.pie(max_con,labels=max_con.index,autopct="%.2f%%",shadow=True,explode=(0.3,0,0,0,0))
plt.title("SALES ANALYSIS OF TOP 5 COUNTRIES USING %")
```

```
Text(0.5, 1.0, 'SALES ANALYSIS OF TOP 5 COUNTRIES USING %')
```



# 5.Rfm analysis



# Rfm and Rfm segments

```
last_date=d2['InvoiceDate'].max().date()
current_date=last_date+pd.Timedelta(days=1)
current_date
```

```
datetime.date(2011, 12, 10)
```

```
rm=d2.groupby('CustomerID').agg({'InvoiceDate':lambda x:(current_date-x.dt.date.max()).days,
                                 'InvoiceNo':'count',
                                 'total_price':'sum'})
rm.columns=['recency','frequency','monetary']
rm
```

| | recency | frequency | monetary |
|---|---|---|---|
| CustomerID | | | |
| **12346.0** | 326 | 2 | 0.00 |
| **12347.0** | 3 | 182 | 4310.00 |
| **12348.0** | 76 | 31 | 1797.24 |
| **12349.0** | 19 | 73 | 1757.55 |
| **12350.0** | 311 | 17 | 334.40 |
| **...** | ... | ... | ... |
| **18280.0** | 278 | 10 | 180.60 |
| **18281.0** | 181 | 7 | 80.82 |
| **18282.0** | 8 | 13 | 176.60 |
| **18283.0** | 4 | 756 | 2094.88 |

```
rm['r seg']=pd.qcut(rm['recency'],4,labels=[4,3,2,1])
rm['f seg']=pd.qcut(rm['frequency'],4,labels=[1,2,3,4])
rm['m seg']=pd.qcut(rm['monetary'],4,labels=[1,2,3,4])
rm['rfm']=rm['r seg'].astype(str)+rm['f seg'].astype(str)

rm
```

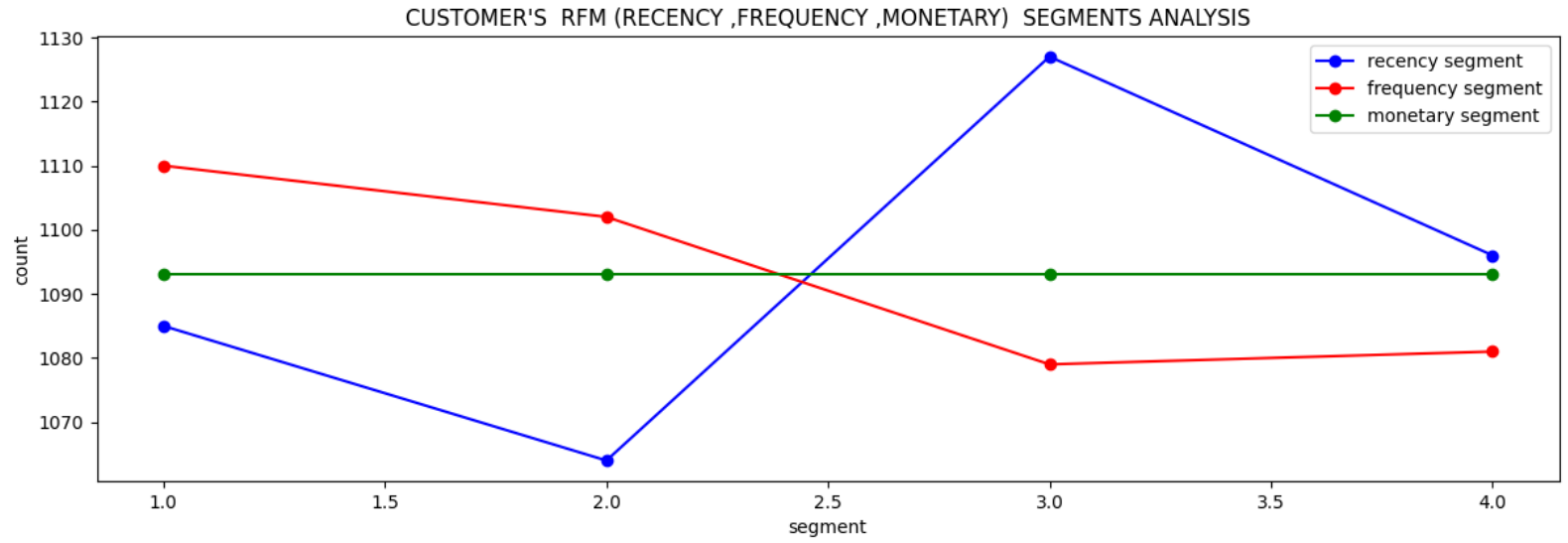
| | recency | frequency | monetary | r seg | f seg | m seg | rfm |
|---|---|---|---|---|---|---|---|
| CustomerID | | | | | | | |
| **12346.0** | 326 | 2 | 0.00 | 1 | 1 | 1 | 11 |
| **12347.0** | 3 | 182 | 4310.00 | 4 | 4 | 4 | 44 |
| **12348.0** | 76 | 31 | 1797.24 | 2 | 2 | 4 | 22 |
| **12349.0** | 19 | 73 | 1757.55 | 3 | 3 | 4 | 33 |
| **12350.0** | 311 | 17 | 334.40 | 1 | 1 | 2 | 11 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **18280.0** | 278 | 10 | 180.60 | 1 | 1 | 1 | 11 |
| **18281.0** | 181 | 7 | 80.82 | 1 | 1 | 1 | 11 |
| **18282.0** | 8 | 13 | 176.60 | 4 | 1 | 1 | 41 |
| **18283.0** | 4 | 756 | 2094.88 | 4 | 4 | 4 | 44 |
| **18287.0** | 43 | 70 | 1837.28 | 3 | 3 | 4 | 33 |

4372 rows × 7 columns

# Rfm segments analysis using graph

```
plt.figure(figsize=(12,4))
plt.plot(r_group.index,r_group.values,label="recency segment",color="blue",marker='o')
plt.plot(f_group.index,f_group.values,label="frequency segment",color="red",marker='o')
plt.plot(m_group.index,m_group.values,label="monetary segment",color="green",marker='o')
plt.tight_layout()
plt.title("CUSTOMER'S  RFM (RECENCY ,FREQUENCY ,MONETARY)  SEGMENTS ANALYSIS")
plt.legend()
plt.xlabel("segment")
plt.ylabel("count")
```

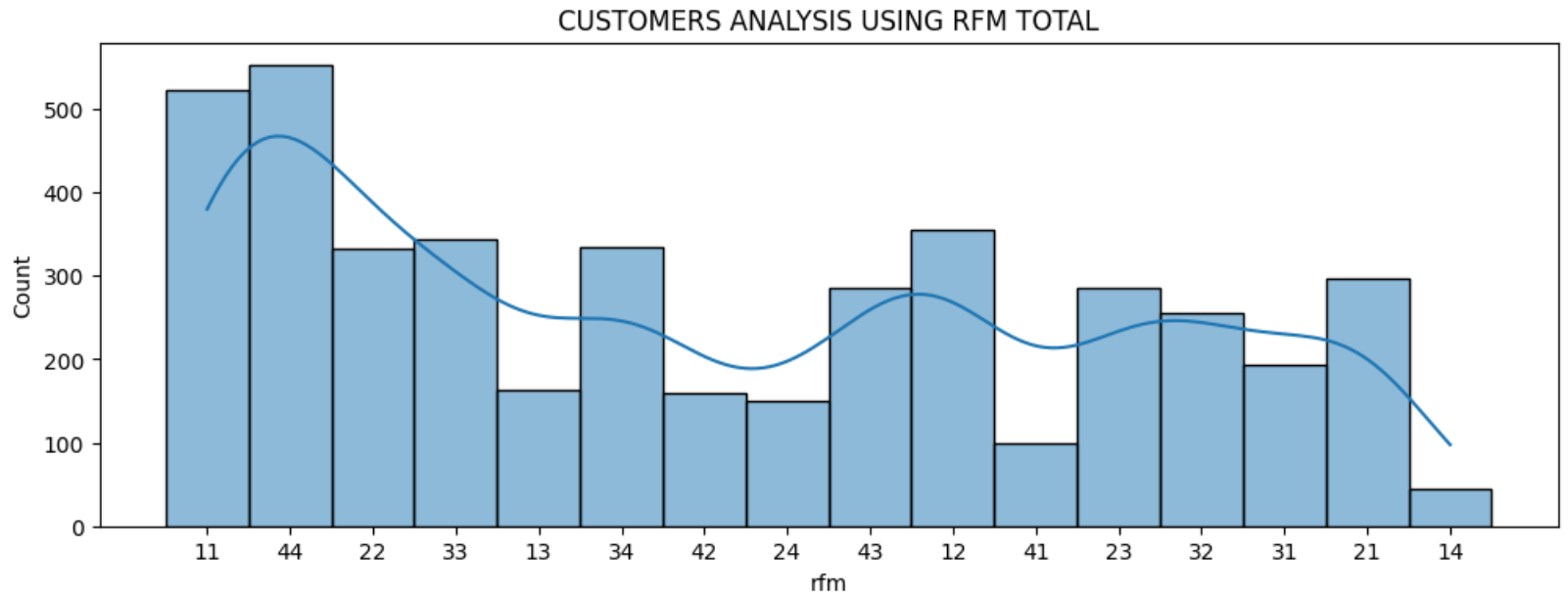
```
Text(99.34722222222221, 0.5, 'count')
```



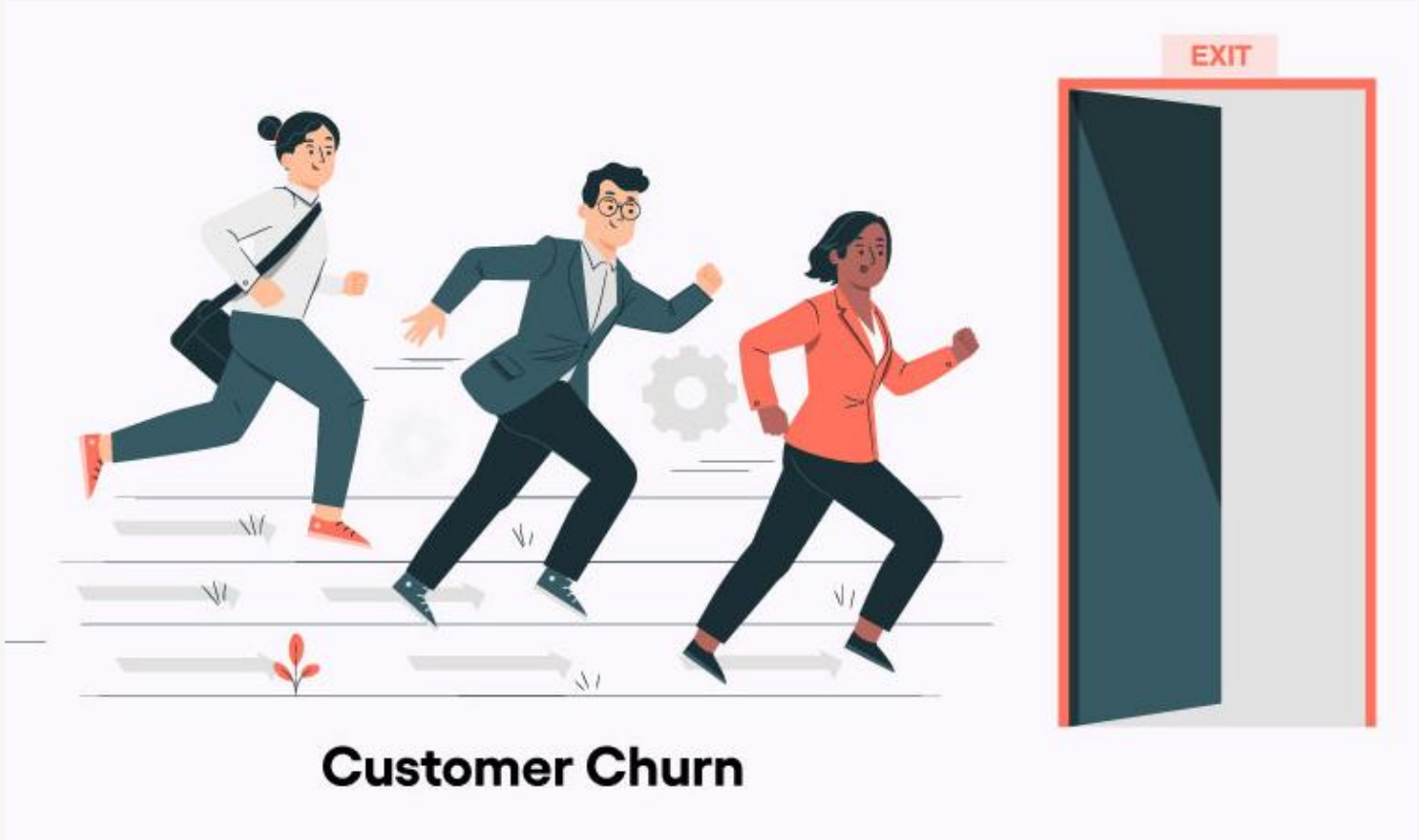
# Customers analysis using rfm total

```
plt.figure(figsize=(12,4))
plt.title('CUSTOMERS ANALYSIS USING RFM TOTAL')
sns.histplot(rm['rfm'],kde=True)
```

```
<Axes: title={'center': 'CUSTOMERS ANALYSIS USING RFM TOTAL'}, xlabel='rfm', ylabel='Count'>
```



# 6.Churn analysis



# Churn analysis

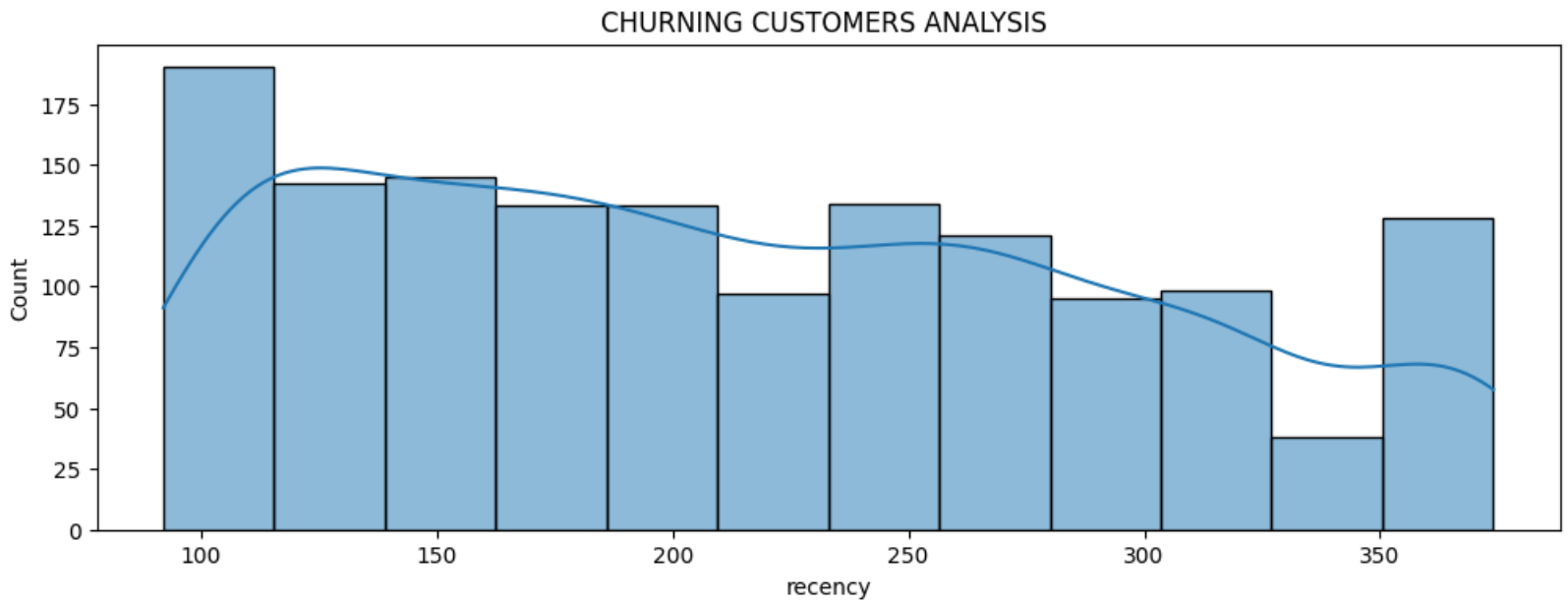
```
churn=rm['recency'][rm.recency >90]
churn.columns=['churning customers']
churn
```

| CustomerID | recency |
|---|---|
| 12346.0 | 326 |
| 12350.0 | 311 |
| 12353.0 | 205 |
| 12354.0 | 233 |
| 12355.0 | 215 |
| ... | ... |
| 18262.0 | 141 |
| 18268.0 | 135 |
| 18269.0 | 359 |
| 18280.0 | 278 |
| 18281.0 | 181 |

1454 rows × 1 columns

```
plt.figure(figsize=(12,4))
plt.title("CHURNING CUSTOMERS ANALYSIS ")
sns.histplot(churn,kde=True)
```

```
<Axes: title={'center': 'CHURNING CUSTOMERS ANALYSIS '}, xlabel='recency', ylabel='Count'>
```



# Overall insights

# Insights

- 🔁 **Majority of customers purchased only once** (Frequency score 1.0), showing low repeat engagement.

- 📈 **Sales increased steadily**, with notable peaks during the 8th to 12th months.

- 🌍 **UK dominates sales**, contributing 89% of total revenue.

- 🕒 **Many customers haven't purchased recently** (Recency score 3.0 – 1130 users), indicating churn risk.

# Insights

- 🚨 **Customers with recency between 100–350 days** are likely to churn and should be prioritized.

- 💰 **Spending remains consistent** across all customer segments (Monetary score).

- ⚖️ **Balanced number of high and low RFM scores** — mix of loyal and disengaged users.

# Recommendations

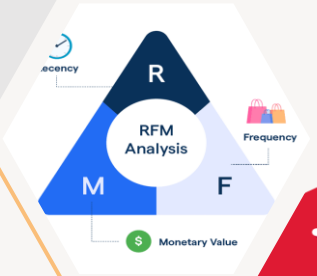
# Recommendations

- Contact churned customers with updates about new features, products, or personalized offers to bring them back.

- Implement a visit-based discount program (e.g., after 5 visits, give a special discount) to improve recency and frequency.

- Use regional retailers or partners in countries like France, Germany, and the Netherlands to grow international reach.

- Invest in marketing and inventory during high-sales months like September to December to leverage year-end demand.


# Thank you

Bhuvaneswari Kapuluru

bhuvaneswari2821@gmail.com