online_retail

July 16, 2023

1 Portfolio Project: Online Retail Exploratory Data Analysis with Python

1.1 Overview

In this project, I will step into the shoes of an entry-level data analyst at an online retail company, helping interpret real-world data to help make a key business decision.

1.2 Case Study

In this project, i was working with transactional data from an online retail store. The dataset contains information about customer purchases, including product details, quantities, prices, and timestamps. Your task is to explore and analyze this dataset to gain insights into the store's sales trends, customer behavior, and popular products.

By conducting exploratory data analysis, you will identify patterns, outliers, and correlations in the data, allowing you to make data-driven decisions and recommendations to optimize the store's operations and improve customer satisfaction. Through visualizations and statistical analysis, you will uncover key trends, such as the busiest sales months, best-selling products, and the store's most valuable customers. Ultimately, this project aims to provide actionable insights that can drive strategic business decisions and enhance the store's overall performance in the competitive online retail market.

1.3 Project Objectives

1. Describe data to answer key questions to uncover insights
2. Gain valuable insights that will help improve online retail performance
3. Provide analytic insights and data-driven recommendations

1.4 Dataset

The dataset you will be working with is the "Online Retail" dataset. It contains transactional data of an online retail store from 2010 to 2011. The dataset is available as a .xlsx file named `Online Retail.xlsx`. it can also be downloaded here.

The dataset contains the following columns:

- InvoiceNo: Invoice number of the transaction
- StockCode: Unique code of the product
- Description: Description of the product
- Quantity: Quantity of the product in the transaction
- InvoiceDate: Date and time of the transaction
- UnitPrice: Unit price of the product
- CustomerID: Unique identifier of the customer
- Country: Country where the transaction occurred

1.5 Tasks

You may explore this dataset in any way you would like - however if you'd like some help getting started, here are a few ideas:

1. Load the dataset into a Pandas DataFrame and display the first few rows to get an overview of the data.
2. Perform data cleaning by handling missing values, if any, and removing any redundant or unnecessary columns.
3. Explore the basic statistics of the dataset, including measures of central tendency and dispersion.
4. Perform data visualization to gain insights into the dataset. Generate appropriate plots, such as histograms, scatter plots, or bar plots, to visualize different aspects of the data.
5. Analyze the sales trends over time. Identify the busiest months and days of the week in terms of sales.
6. Explore the top-selling products and countries based on the quantity sold.
7. Identify any outliers or anomalies in the dataset and discuss their potential impact on the analysis.
8. Draw conclusions and summarize your findings from the exploratory data analysis.

1.6 Task 1: Load the Data

[48]:
```
import pandas as pd
import numpy as np
import seaborn as sns                          #visualisation
import matplotlib.pyplot as plt                #visualisation
%matplotlib inline
sns.set(color_codes=True)
```

[64]:
```
df = pd.read_excel("Online Retail.xlsx")
# To display the top 5 rows
df.head(5)
```

[64]:
```
  InvoiceNo StockCode                          Description  Quantity  \
0    536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6
1    536365     71053                  WHITE METAL LANTERN         6
2    536365    84406B       CREAM CUPID HEARTS COAT HANGER         8
```

```
3    536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6
4    536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6

          InvoiceDate  UnitPrice  CustomerID         Country
0 2010-12-01 08:26:00       2.55     17850.0  United Kingdom
1 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
2 2010-12-01 08:26:00       2.75     17850.0  United Kingdom
3 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
4 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
```

[62]:
```
df.tail(5)
```

[62]:
```
       InvoiceNo StockCode                      Description  Quantity  \
541904    581587     22613      PACK OF 20 SPACEBOY NAPKINS        12
541905    581587     22899     CHILDREN'S APRON DOLLY GIRL          6
541906    581587     23254    CHILDRENS CUTLERY DOLLY GIRL          4
541907    581587     23255  CHILDRENS CUTLERY CIRCUS PARADE         4
541908    581587     22138    BAKING SET 9 PIECE RETROSPOT          3

               InvoiceDate  UnitPrice  CustomerID Country
541904 2011-12-09 12:50:00       0.85     12680.0  France
541905 2011-12-09 12:50:00       2.10     12680.0  France
541906 2011-12-09 12:50:00       4.15     12680.0  France
541907 2011-12-09 12:50:00       4.15     12680.0  France
541908 2011-12-09 12:50:00       4.95     12680.0  France
```

[51]:
```
df.dtypes
```

[51]:
```
InvoiceNo              object
StockCode              object
Description            object
Quantity                int64
InvoiceDate    datetime64[ns]
UnitPrice             float64
CustomerID            float64
Country                object
dtype: object
```

2 Task 2: Clean the Data

[52]:
```
df=df[['CustomerID','InvoiceNo','StockCode','Quantity','UnitPrice','Description','Country']]
```

[53]:
```
df.head(10)
```

[53]:
```
   CustomerID InvoiceNo StockCode  Quantity  UnitPrice  \
0     17850.0    536365    85123A         6       2.55
1     17850.0    536365     71053         6       3.39
2     17850.0    536365    84406B         8       2.75
3     17850.0    536365    84029G         6       3.39
4     17850.0    536365    84029E         6       3.39
5     17850.0    536365     22752         2       7.65
6     17850.0    536365     21730         6       4.25
7     17850.0    536366     22633         6       1.85
8     17850.0    536366     22632         6       1.85
9     13047.0    536367     84879        32       1.69

                           Description         Country
0   WHITE HANGING HEART T-LIGHT HOLDER  United Kingdom
1                  WHITE METAL LANTERN  United Kingdom
2       CREAM CUPID HEARTS COAT HANGER  United Kingdom
3  KNITTED UNION FLAG HOT WATER BOTTLE  United Kingdom
4       RED WOOLLY HOTTIE WHITE HEART.  United Kingdom
5         SET 7 BABUSHKA NESTING BOXES  United Kingdom
6    GLASS STAR FROSTED T-LIGHT HOLDER  United Kingdom
7                HAND WARMER UNION JACK  United Kingdom
8             HAND WARMER RED POLKA DOT  United Kingdom
9        ASSORTED COLOUR BIRD ORNAMENT  United Kingdom
```

[54]:
```python
df.shape
```

[54]:
```
(541909, 7)
```

[55]:
```python
duplicate_rows_df = df[df.duplicated()]
print("number of duplicate rows: ", duplicate_rows_df.shape)
```

```
number of duplicate rows:  (5269, 7)
```

[56]:
```python
df.count()
```

[56]:
```
CustomerID     406829
InvoiceNo      541909
StockCode      541909
Quantity       541909
UnitPrice      541909
Description    540455
Country        541909
dtype: int64
```

[57]:
```python
df = df.drop_duplicates()
df.head(5)
```

```
[57]:    CustomerID InvoiceNo StockCode  Quantity  UnitPrice  \
      0     17850.0    536365    85123A         6       2.55
      1     17850.0    536365     71053         6       3.39
      2     17850.0    536365    84406B         8       2.75
      3     17850.0    536365    84029G         6       3.39
      4     17850.0    536365    84029E         6       3.39

                                 Description         Country
      0   WHITE HANGING HEART T-LIGHT HOLDER  United Kingdom
      1                  WHITE METAL LANTERN  United Kingdom
      2       CREAM CUPID HEARTS COAT HANGER  United Kingdom
      3  KNITTED UNION FLAG HOT WATER BOTTLE  United Kingdom
      4       RED WOOLLY HOTTIE WHITE HEART.  United Kingdom
```

```
[58]: df.count()
```

```
[58]: CustomerID     401603
      InvoiceNo      536640
      StockCode      536640
      Quantity       536640
      UnitPrice      536640
      Description    535186
      Country        536640
      dtype: int64
```

```
[59]: print(df.isnull().sum())
```

```
CustomerID     135037
InvoiceNo           0
StockCode           0
Quantity            0
UnitPrice           0
Description      1454
Country             0
dtype: int64
```

```
[60]: df = df.dropna()     # Dropping the missing values.
      df.count()
```

```
[60]: CustomerID     401603
      InvoiceNo      401603
      StockCode      401603
      Quantity       401603
      UnitPrice      401603
      Description    401603
      Country        401603
      dtype: int64
```

[47]:
```
print(df.isnull().sum())
```

```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

3 Task3: Data visualization & Analysis

[65]:
```
TotalAmount = df['Quantity'] * df['UnitPrice']
df.insert(loc=5,column='TotalAmount',value=TotalAmount)
```

[66]:
```
new_df =␣
 ↪df[['CustomerID','InvoiceNo','StockCode','Quantity','TotalAmount','InvoiceDate','Country']]

new_df2 = df.copy()
```

[67]:
```
new_df.head()
```

[67]:
```
   CustomerID InvoiceNo StockCode  Quantity  TotalAmount         InvoiceDate  \
0     17850.0    536365    85123A         6        15.30 2010-12-01 08:26:00
1     17850.0    536365     71053         6        20.34 2010-12-01 08:26:00
2     17850.0    536365    84406B         8        22.00 2010-12-01 08:26:00
3     17850.0    536365    84029G         6        20.34 2010-12-01 08:26:00
4     17850.0    536365    84029E         6        20.34 2010-12-01 08:26:00

          Country
0  United Kingdom
1  United Kingdom
2  United Kingdom
3  United Kingdom
4  United Kingdom
```

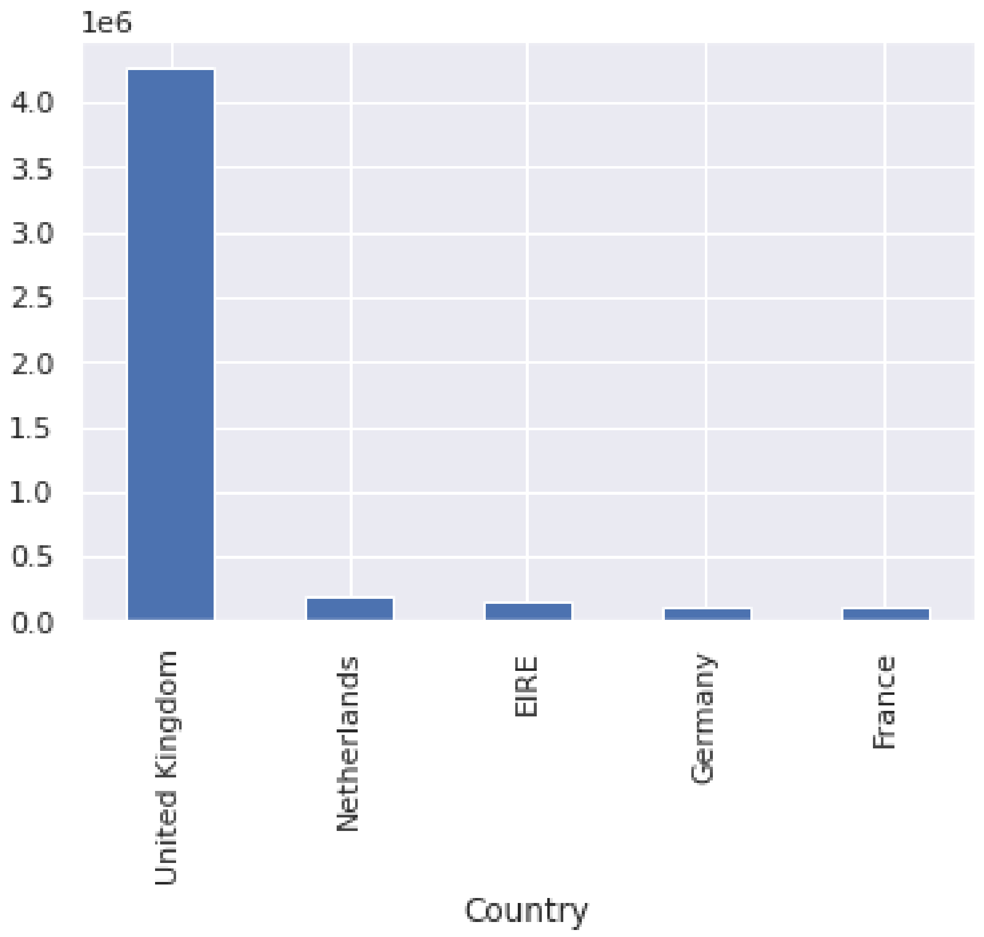
3.0.1 Exploratory Data Analysis(EDA)

[69]:
```
country_price = new_df.groupby('Country')['Quantity'].sum().
 ↪sort_values(ascending = False)
country_price
# Grouping countries by TotalAmount of sales
```

```
[69]: Country
      United Kingdom          4263829
      Netherlands              200128
      EIRE                     142637
      Germany                  117448
      France                   110480
      Australia                 83653
      Sweden                    35637
      Switzerland               30325
      Spain                     26824
      Japan                     25218
      Belgium                   23152
      Norway                    19247
      Portugal                  16180
      Finland                   10666
      Channel Islands            9479
      Denmark                    8188
      Italy                      7999
      Cyprus                     6317
      Singapore                  5234
      Austria                    4827
      Hong Kong                  4769
      Israel                     4353
      Poland                     3653
      Unspecified                3300
      Canada                     2763
      Iceland                    2458
      Greece                     1556
      USA                        1034
      United Arab Emirates        982
      Malta                       944
      Lithuania                   652
      Czech Republic              592
      European Community          497
      Lebanon                     386
      Brazil                      356
      RSA                         352
      Bahrain                     260
      Saudi Arabia                 75
      Name: Quantity, dtype: int64
```

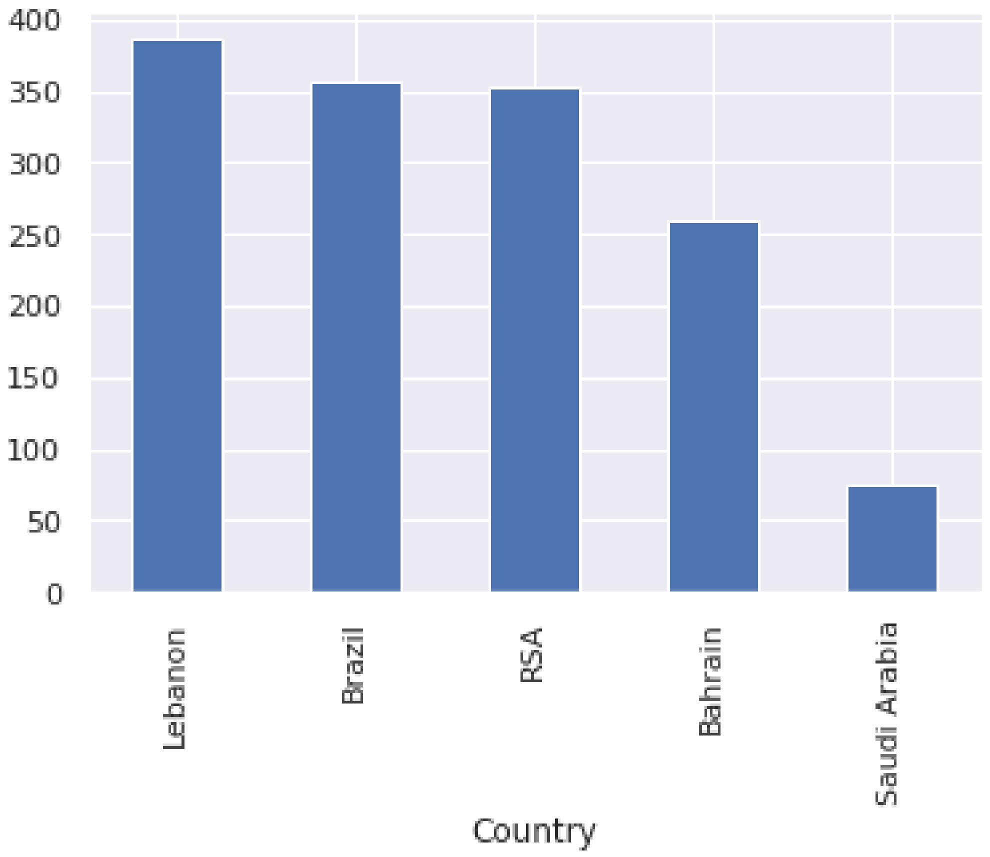
```
[70]: country_price[:5].plot(kind = 'bar')
      # Top 5 Companies with high number of purchase
```

```
[70]: <matplotlib.axes._subplots.AxesSubplot at 0x7f7d4be3e1d0>
```



```
[71]: # 5 Compaies with least number of purchase
      country_price[33:].plot(kind = 'bar')
```

```
[71]: <matplotlib.axes._subplots.AxesSubplot at 0x7f7d543e5bd0>
```



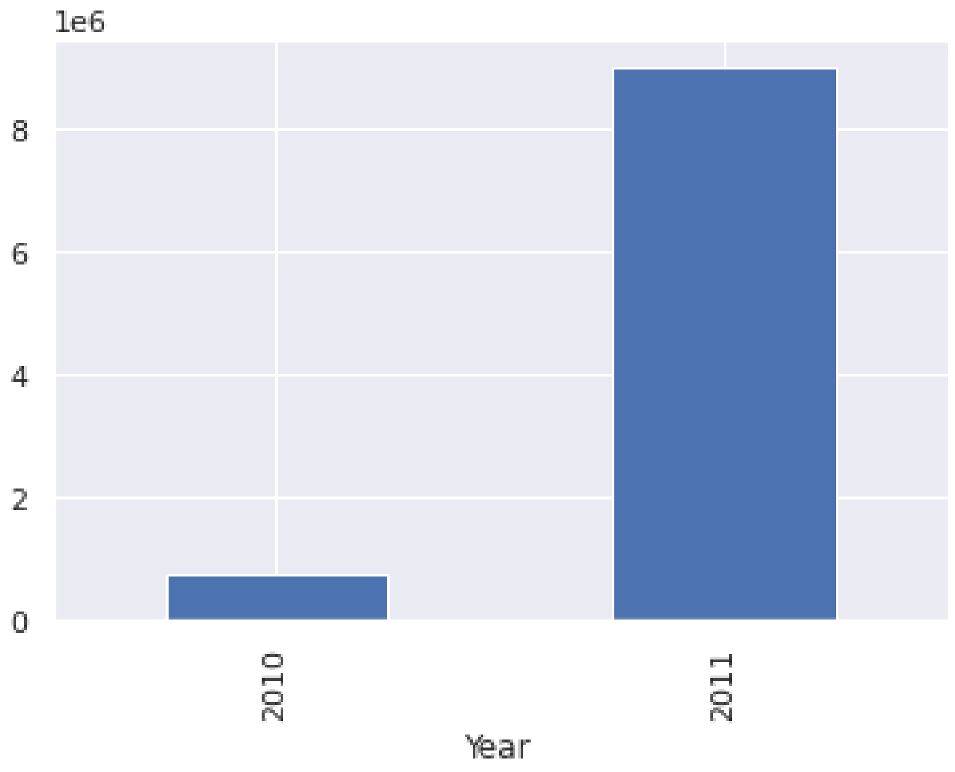
[72]:
```
# Adding year feature to the dataset

timest = new_df['InvoiceDate'].dt.year

new_df['Year'] = timest

new_df.head()
```

[72]:
```
   CustomerID InvoiceNo StockCode  Quantity  TotalAmount         InvoiceDate  \
0     17850.0    536365    85123A         6        15.30 2010-12-01 08:26:00
1     17850.0    536365     71053         6        20.34 2010-12-01 08:26:00
2     17850.0    536365    84406B         8        22.00 2010-12-01 08:26:00
3     17850.0    536365    84029G         6        20.34 2010-12-01 08:26:00
4     17850.0    536365    84029E         6        20.34 2010-12-01 08:26:00

          Country  Year
0  United Kingdom  2010
1  United Kingdom  2010
2  United Kingdom  2010
3  United Kingdom  2010
```

```
4  United Kingdom  2010
```

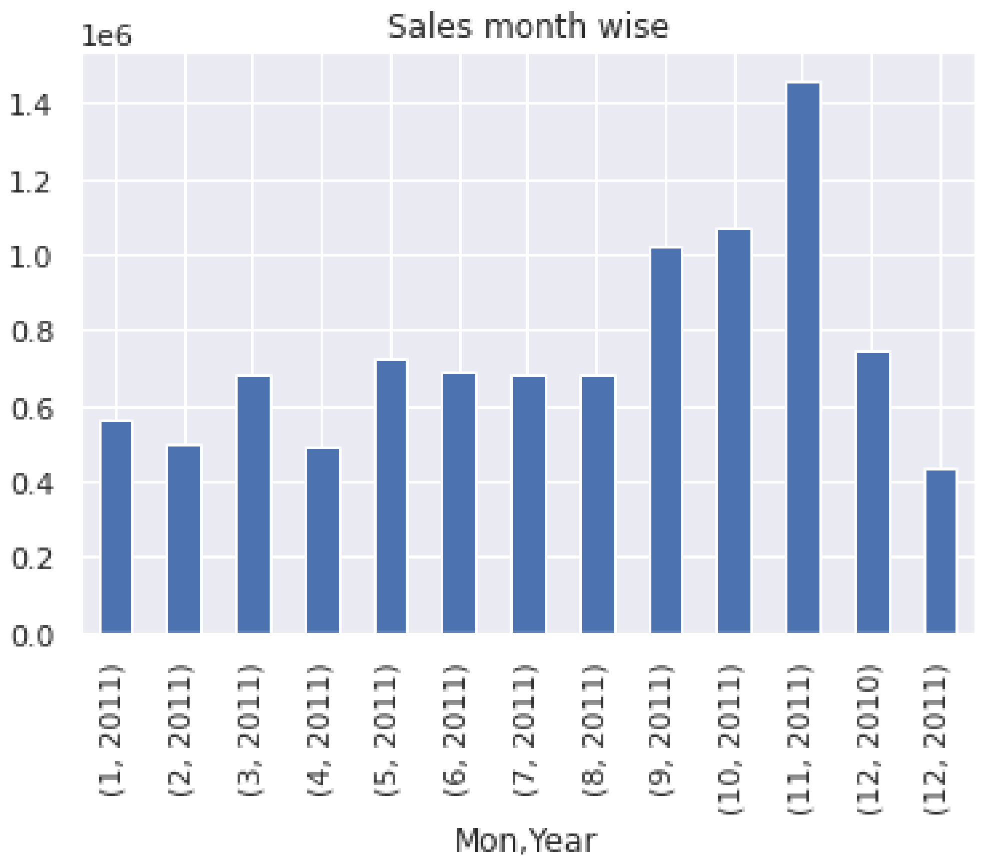
[73]:
```
# Total sales for different years

new_df.groupby('Year')['TotalAmount'].sum().plot(kind = 'bar')
```

[73]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f7d5d6ef210>
```



[74]:
```
# Sales for different months

new_df['Mon'] = new_df['InvoiceDate'].dt.month
new_df['month'] = new_df['InvoiceDate'].dt.month_name()
new_df.groupby(['Mon','Year'])['TotalAmount'].sum().plot(kind = 'bar', title =␣
 ↪'Sales month wise')
```

[74]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f7d4a4b1a90>
```



[75]:
```
# Checking why dec 2011 has a drop comparing to nov 2011
get_2011 = new_df[(new_df['Year'] == 2011)]
get_dec2011 = get_2011[(new_df['month'] == 'December')]
get_dec2011 = get_dec2011['InvoiceDate'].dt.date.unique()
get_dec2011
```

[75]:
```
array([datetime.date(2011, 12, 1), datetime.date(2011, 12, 2),
       datetime.date(2011, 12, 4), datetime.date(2011, 12, 5),
       datetime.date(2011, 12, 6), datetime.date(2011, 12, 7),
       datetime.date(2011, 12, 8), datetime.date(2011, 12, 9)],
      dtype=object)
```

3.1 Answer :

3.2 Performance Analysis

Sales Performance can be seen with

Number of sales every month

Number of sales every year

We see that in 2010 we have sales only for dec and in 2011 we have sales for all months

We can see that from September to Novembor we have very good sales

We could see that DEC 2010 we have sales of 748957 and in DEC 2011 we have sales of 433668 which is a huge drop when analyzed further found out that We have only data upto 9th on dec 2011, so we find a sales drop in the month of dec 2011

4 Task 4:Analysis

potential areas of improvement for the business

[76]:
```
new_df.head()
```

[76]:
```
   CustomerID InvoiceNo StockCode  Quantity  TotalAmount         InvoiceDate  \
0     17850.0    536365    85123A         6        15.30 2010-12-01 08:26:00
1     17850.0    536365     71053         6        20.34 2010-12-01 08:26:00
2     17850.0    536365    84406B         8        22.00 2010-12-01 08:26:00
3     17850.0    536365    84029G         6        20.34 2010-12-01 08:26:00
4     17850.0    536365    84029E         6        20.34 2010-12-01 08:26:00

          Country  Year  Mon     month
0  United Kingdom  2010   12  December
1  United Kingdom  2010   12  December
2  United Kingdom  2010   12  December
3  United Kingdom  2010   12  December
4  United Kingdom  2010   12  December
```

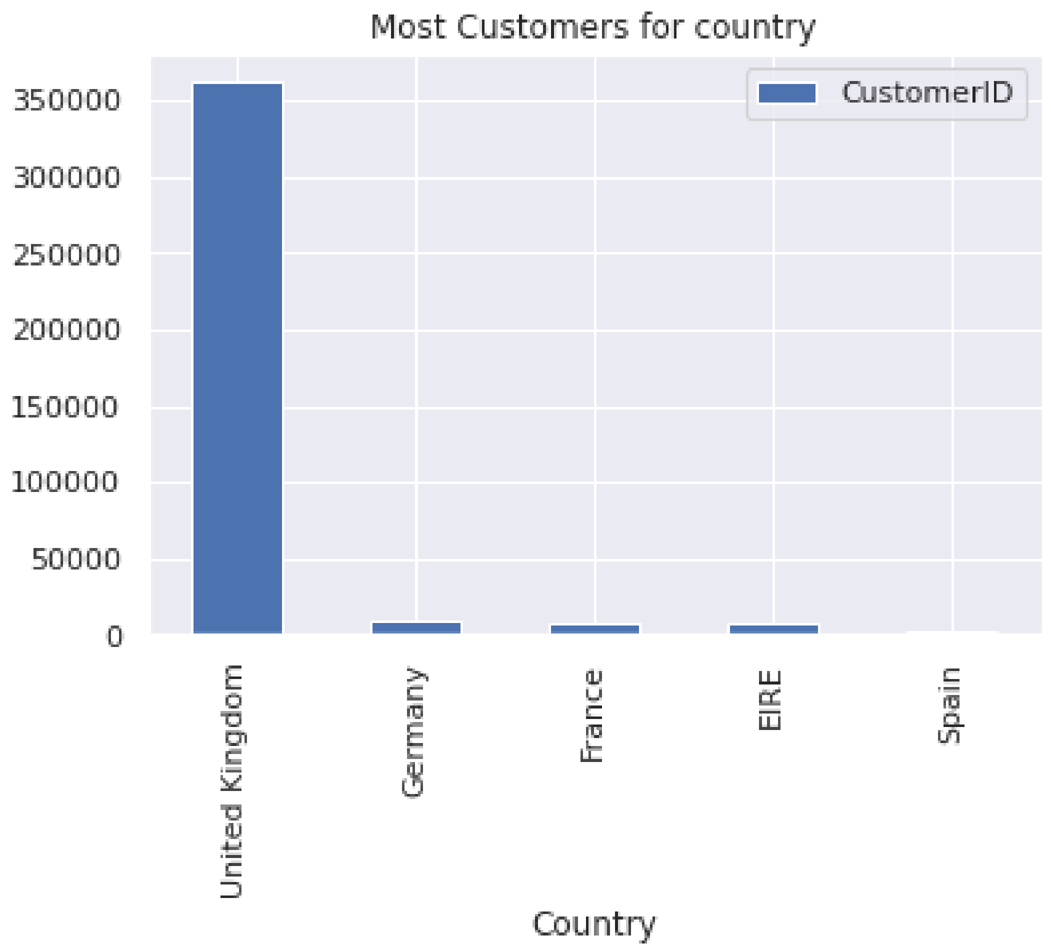
[77]:
```
new_df = new_df.dropna()
new_df.isnull().sum()
```

[77]:
```
CustomerID     0
InvoiceNo      0
StockCode      0
Quantity       0
TotalAmount    0
InvoiceDate    0
Country        0
Year           0
Mon            0
month          0
dtype: int64
```

[78]:
```
#Countries with more number of customers
cus_id = pd.DataFrame(new_df.groupby('Country')['CustomerID'].count().
 ↪sort_values(ascending = False))
```

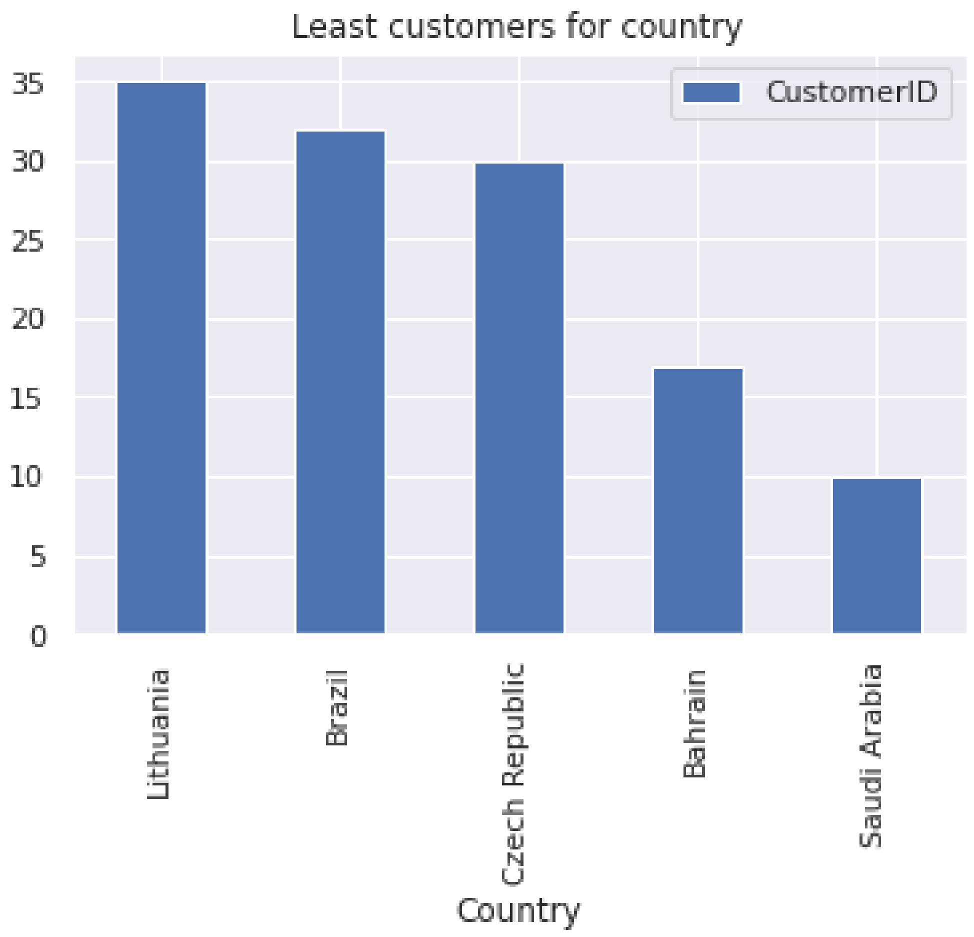
```
cus_id[:5].plot(kind = 'bar', title = 'Most Customers for country')
```

```
[78]: <matplotlib.axes._subplots.AxesSubplot at 0x7f7d4aeb1b50>
```



```
[79]: # Countries with less number of customers
      cus_id[-5:].plot(kind = 'bar', title = 'Least customers for country')
```

```
[79]: <matplotlib.axes._subplots.AxesSubplot at 0x7f7d52018990>
```



5 Answer-

5.0.1 We see that september to december we have very high sales

We can concentrate on improving the sales for the other 8 months

We find very less number of customers in Lithania, Brazil, Czech Republic, Bahrain, Saudi Arabia

We have very less sales for Lebanon, Brazil, RSA, Bahrain, Saudi Arabia.

We can concentrate on improving their sales

```
[ ]:
```

[ ]:
```

```

[ ]:
```

```

[ ]:
```

```