Assignment1

March 7, 2024

[1]:
```
import pandas as pd #for data frames
```

[3]:
```
import matplotlib.pyplot as plt #for graphs
```

```
Matplotlib is building the font cache; this may take a moment.
```

[5]:
```
import seaborn as sns #for graphs
```

[7]:
```
import datetime as dt
```

[11]:
```
data = pd.read_excel("online_retail_II.xlsx",dtype={'CustomerID':␣
↪str,'InvoiceID': str})
```

[12]:
```
df=pd.DataFrame(data)
```

[15]:
```
print(df.head())
```

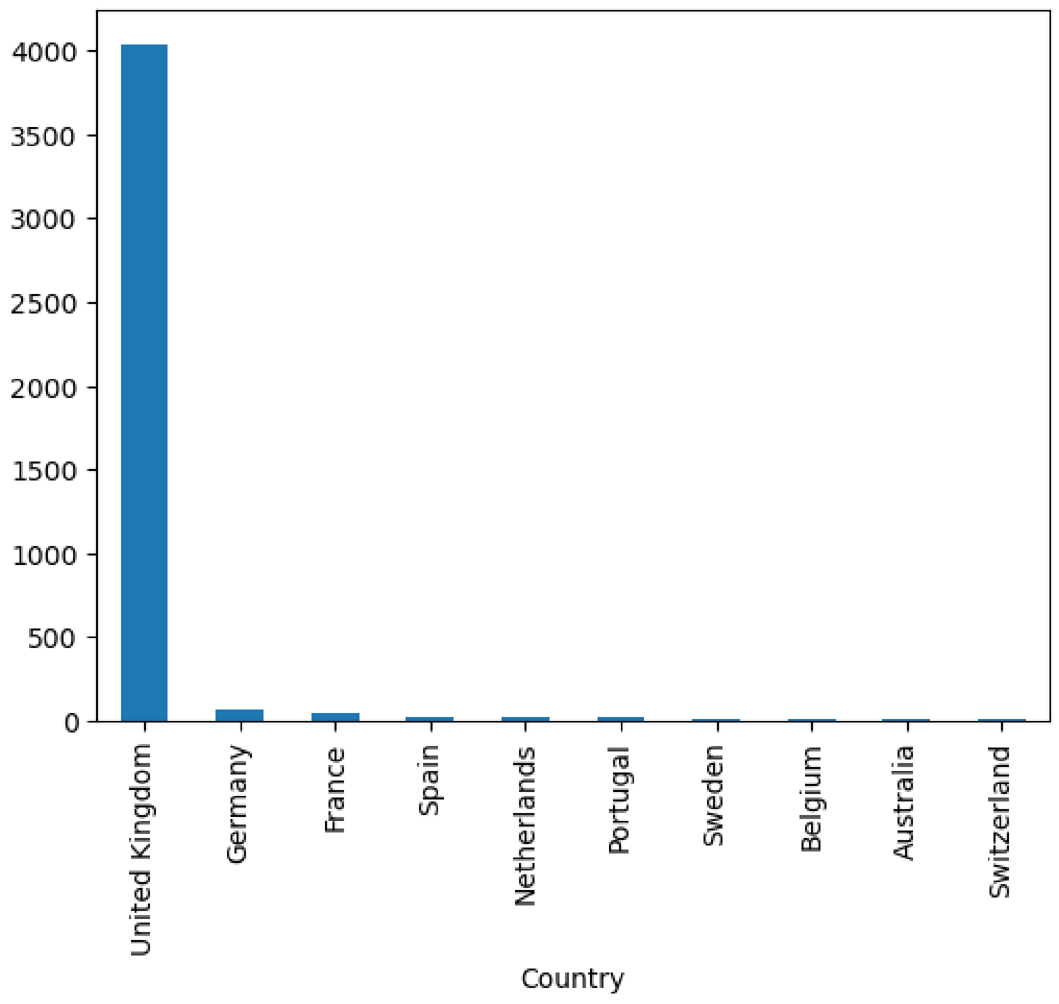
```
  Invoice StockCode                          Description  Quantity  \
0  489434     85048  15CM CHRISTMAS GLASS BALL 20 LIGHTS        12
1  489434    79323P                   PINK CHERRY LIGHTS        12
2  489434    79323W                  WHITE CHERRY LIGHTS        12
3  489434     22041         RECORD FRAME 7" SINGLE SIZE         48
4  489434     21232       STRAWBERRY CERAMIC TRINKET BOX        24

          InvoiceDate  Price  Customer ID         Country
0 2009-12-01 07:45:00   6.95      13085.0  United Kingdom
1 2009-12-01 07:45:00   6.75      13085.0  United Kingdom
2 2009-12-01 07:45:00   6.75      13085.0  United Kingdom
3 2009-12-01 07:45:00   2.10      13085.0  United Kingdom
4 2009-12-01 07:45:00   1.25      13085.0  United Kingdom
```

[24]:
```
filtered_data = data[['Country', 'Customer ID']].drop_duplicates()
```

[26]:
```
filtered_data.Country.value_counts()[:10].plot(kind='bar')
```

[26]:
```
<Axes: xlabel='Country'>
```



[28]:
```
uk_data=data[data.Country=='United Kingdom']
```

[30]:
```
uk_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 485852 entries, 0 to 525460
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   Invoice      485852 non-null  object
 1   StockCode    485852 non-null  object
 2   Description  482924 non-null  object
 3   Quantity     485852 non-null  int64
 4   InvoiceDate  485852 non-null  datetime64[ns]
 5   Price        485852 non-null  float64
 6   Customer ID  379423 non-null  float64
 7   Country      485852 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(1), object(4)
memory usage: 33.4+ MB
```

[32]:
```
print(data.nunique())
```

```
Invoice        28816
StockCode       4632
Description     4681
Quantity         825
InvoiceDate    25296
Price           1606
Customer ID     4383
Country           40
dtype: int64
```

[34]:
```
print(uk_data.describe())
```

```
            Quantity                    InvoiceDate          Price  \
count  485852.000000                         485852  485852.000000
mean        9.116039  2010-06-27 21:41:53.628553472       4.543470
min     -9600.000000            2009-12-01 07:45:00  -53594.360000
25%         1.000000            2010-03-19 13:01:00       1.250000
50%         3.000000            2010-07-05 12:09:00       2.100000
75%        10.000000            2010-10-15 14:52:00       4.210000
max     10200.000000            2010-12-09 20:01:00   25111.090000
std        85.883463                            NaN     149.623198

         Customer ID
count  379423.000000
mean    15559.935694
min     12346.000000
25%     14210.000000
50%     15581.000000
75%     16938.000000
max     18287.000000
std      1593.744626
```

[36]:
```
uk_data = uk_data[(uk_data['Quantity']>0)]
```

[38]:
```
uk_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 474938 entries, 0 to 525460
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   Invoice      474938 non-null  object
 1   StockCode    474938 non-null  object
 2   Description  473837 non-null  object
 3   Quantity     474938 non-null  int64
 4   InvoiceDate  474938 non-null  datetime64[ns]
 5   Price        474938 non-null  float64
 6   Customer ID  370951 non-null  float64
 7   Country      474938 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(1), object(4)
memory usage: 32.6+ MB
```

[51]:
```
uk_data = uk_data[['Customer ID', 'Invoice', 'InvoiceDate', 'Quantity',␣
 ↪'Price']]
```

[53]:
```
uk_data['TotalPrice'] = uk_data['Quantity'] * uk_data['Price']
```

[57]:
```
uk_data['InvoiceDate'].min(),uk_data['InvoiceDate'].max()
```

[57]:
```
(Timestamp('2009-12-01 07:45:00'), Timestamp('2010-12-09 20:01:00'))
```

[59]:
```
PRESENT = dt.datetime(2011,12,10)
```

[61]:
```
uk_data['InvoiceDate'] = pd.to_datetime(uk_data['InvoiceDate'])
```

[63]:
```
print(uk_data.head())
```

```
   Customer ID Invoice         InvoiceDate  Quantity  Price  TotalPrice
0      13085.0  489434 2009-12-01 07:45:00        12   6.95        83.4
1      13085.0  489434 2009-12-01 07:45:00        12   6.75        81.0
2      13085.0  489434 2009-12-01 07:45:00        12   6.75        81.0
3      13085.0  489434 2009-12-01 07:45:00        48   2.10       100.8
4      13085.0  489434 2009-12-01 07:45:00        24   1.25        30.0
```

[105]:
```
rfm = uk_data.groupby('Customer ID').agg({'InvoiceDate': lambda date:
                                        (PRESENT - date.max()).days, 'Invoice':␣
 ↪lambda num: len(num),
                                        'TotalPrice': lambda price: price.
 ↪sum()})
```

[97]:
```
print(rfm.head())
```

```
             InvoiceDate  Invoice
Customer ID
12346.0              529       33
12608.0              404       16
12745.0              486       22
12746.0              540       17
12747.0              369      154
```

[107]:
```
rfm.columns = ['recency', 'frequency', 'monetary']
```

[109]:
```
rfm['r_quartile'] = pd.qcut(rfm['recency'], 4, ['1','2','3','4'])
```

[111]:
```
rfm['f_quartile'] = pd.qcut(rfm['frequency'], 4, ['4','3','2','1'])
```

[113]:
```
rfm['m_quartile'] = pd.qcut(rfm['monetary'], 4, ['4','3','2','1'])
```

[115]:
```
print(rfm.head())
```

```
             recency  frequency  monetary r_quartile f_quartile m_quartile
Customer ID
12346.0          529         33    372.86          4          3          3
12608.0          404         16    415.79          2          4          3
12745.0          486         22    723.85          3          3          2
12746.0          540         17    254.55          4          4          4
12747.0          369        154   5080.53          1          1          1
```

[117]:
```
rfm['RFM_Score'] = rfm.r_quartile.astype(str)+ rfm.f_quartile.astype(str) + rfm.
 ↪m_quartile.astype(str)
print(rfm.head())
```

```
             recency  frequency  monetary r_quartile f_quartile m_quartile  \
Customer ID
12346.0          529         33    372.86          4          3          3
12608.0          404         16    415.79          2          4          3
12745.0          486         22    723.85          3          3          2
12746.0          540         17    254.55          4          4          4
12747.0          369        154   5080.53          1          1          1

            RFM_Score
Customer ID
12346.0           433
12608.0           243
12745.0           332
12746.0           444
12747.0           111
```

[119]:
```
print(rfm[rfm['RFM_Score']=='111'].sort_values('monetary',ascending=False).
 ↪head())
```

```
             recency  frequency   monetary r_quartile f_quartile m_quartile  \
Customer ID
18102.0          365        627  349164.35          1          1          1
13694.0          373        957  131443.19          1          1          1
17511.0          367        948   84541.17          1          1          1
15061.0          367        584   83284.38          1          1          1
16684.0          379        441   80489.21          1          1          1
```

```
             RFM_Score
Customer ID
18102.0            111
13694.0            111
17511.0            111
15061.0            111
16684.0            111
```

```
[121]: print(rfm.columns)
```

```
Index(['recency', 'frequency', 'monetary', 'r_quartile', 'f_quartile',
       'm_quartile', 'RFM_Score'],
      dtype='object')
```

```
[ ]:
```