Visualizing the Analysis of Customer Dataset

Shruti Walawalkar
Mar 4 · 4 min read

> Walkthrough of a workpiece I happened to be spending my last few hours working on. The idea behind sharing this naive project is to communicate my thought process. This is my first medium story. Also, the workpiece is me randomly brainstorming and practicing python. Feedback appreciated! *

The problem statement is to visualize “sales” insights from the dataset. To begin with, I loaded the required modules. I have used pandas, numpy, seaborn, scipy and matplotlib.

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from scipy.sparse import csr_matrix, lil_matrix
from scipy import spatial
from scipy import stats

%matplotlib inline
```

Later, I loaded the dataset. pandas.Dataframe.info() in python is a method that returns information about the index datatype and column datatypes, non-null values and memory usage of the dataframe.

In [3]:
```
#importing dataset
dataset= pd.read_csv("data.csv", encoding="ISO-8859-1")
```

In [6]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   InvoiceNo    541909 non-null  object 
 1   StockCode    541909 non-null  object 
 2   Description  540455 non-null  object 
 3   Quantity     541909 non-null  int64  
 4   InvoiceDate  541909 non-null  object 
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object 
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
```

Running that command showed how there were lesser non-null CustomerIDs in comparison to InvoiceNo. This hinted that the CustomerID needs to be cleaned. Also, we see the datatype of 'InvoiceDate' is Object. In order to be able to perform any date-time series operations on the dataset, it is considered to be a good practice to convert it to a Datetime format.

```
#removing where customerID is null
dataset=dataset[pd.notnull(dataset['CustomerID'])]
#date format
dataset.InvoiceDate = pd.to_datetime(dataset.InvoiceDate, format="%m/%d/%Y %H:%M")
```

Now, dataframe.info() returns equal rows and the data types of columns is fixed. We proceed with studying the dataset for generating derived column logics. What I wanted to project is "sales" insight. I can see we had "Quantity" and "Unit Price" as two columns. I generated revenue from those two columns. I also made sure we have correct Quantity values and so cleaned that column.

I also tried to eliminate outliers and normalize the dataset by performing z-score normalization. Z-score is a numerical measurement used in statistics of a value's relationship to the mean (average) of a group of values, measured in terms of standard deviations from the mean.

In [11]:
```
dataset = dataset[dataset["InvoiceNo"].astype(str).str[0] != "C"]
dataset = dataset[dataset["InvoiceNo"].astype(str).str[0] != "A"]
dataset = dataset[dataset["Quantity"] > 0 ]
#Z-score Normalization
dataset = dataset[ np.abs((dataset['UnitPrice']-dataset['UnitPrice'].mean())/dataset['UnitPrice'].std())<=1]
dataset = dataset[ np.abs((dataset['Quantity']-dataset['Quantity'].mean())/dataset['Quantity'].std())<=1]
dataset.describe()
```

Out[11]:

	Quantity	UnitPrice	CustomerID
count	394884.000000	394884.000000	394884.000000
mean	10.498914	2.826652	15295.252613
std	17.463665	2.830359	1711.717290
min	1.000000	0.000000	12347.000000
25%	2.000000	1.250000	13969.000000
50%	5.000000	1.950000	15159.000000
75%	12.000000	3.750000	16795.000000
max	192.000000	25.000000	18287.000000

In [12]:
```
#Revenue Column for further visualizations
dataset['Revenue'] = dataset['Quantity']*dataset['UnitPrice']

dataset['Date'] = dataset['InvoiceDate'].dt.date
dataset['Day'] = dataset['InvoiceDate'].dt.day
dataset['Month'] = dataset['InvoiceDate'].dt.month
dataset['Year'] = dataset['InvoiceDate'].dt.year
dataset['Hour'] = dataset['InvoiceDate'].dt.hour
dataset['Week'] = dataset['InvoiceDate'].dt.week
dataset['Minute'] = dataset['InvoiceDate'].dt.minute
```

Data Visualizations

#1 — Plotting the monthly revenue over the total time period

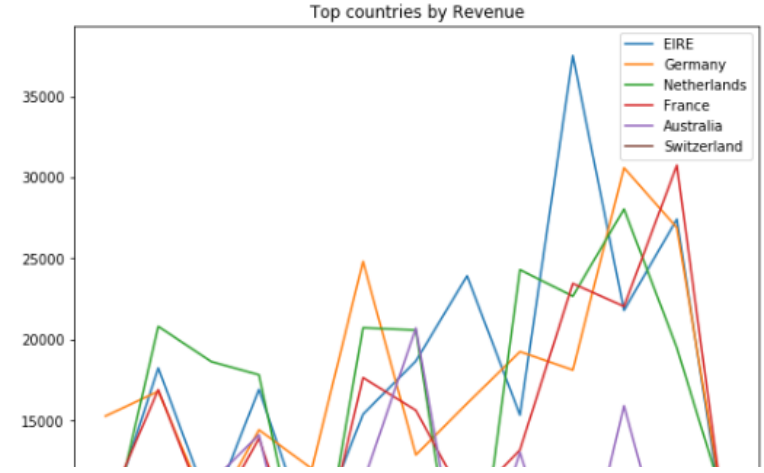
```
In [214]: sales = dataset[['Year', 'Month', 'Revenue']].groupby(['Year', 'Month']).sum().reset_index()

In [216]: sales['Day']=1
          sales['Date'] = pd.to_datetime(sales[['Year', 'Month', 'Day']])

In [218]: sales=sales.drop(["Year","Month","Day"],axis=1)

In [221]: sales.plot(x='Date',y='Revenue', title='Monthly Revenue', color='Red')
Out[221]: <matplotlib.axes._subplots.AxesSubplot at 0x200396df940>
```

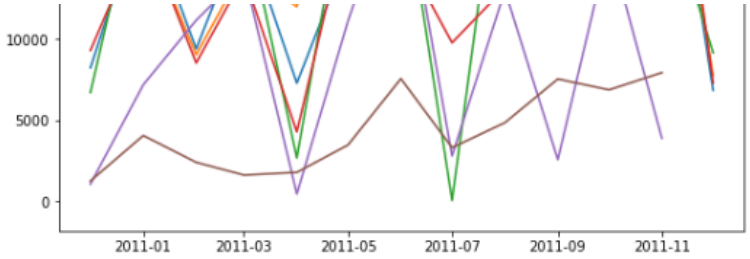
#2 — How much did the top countries added to the revenue?

The plot depicts the total monthly revenue of our top 5 contributors

```
#top countries Contributing to Sales
sales_top = dataset[['Revenue', 'Country']].groupby(['Country']).sum().reset_index().sort_values(by='Revenue', ascending=Fals
```

```
fig, ax = plt.subplots(figsize=(9, 9))
for c in sales_top:
    sales = dataset[dataset['Country'] == c]
    sales = sales[['Year', 'Month', 'Revenue']].groupby(['Year', 'Month']).sum().reset_index()
    sales['Day'] = 1
    sales['Date'] = pd.to_datetime(sales[['Year', 'Month', 'Day']])
    ax.plot(sales['Date'],sales['Revenue'], label=c)
    ax.legend()
    ax.set_title("Top countries by Revenue")
```



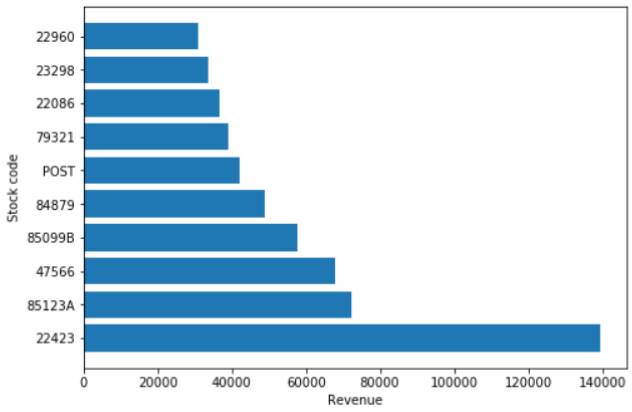


#3 — Top 10 Stock Codes and their revenue Contribution

In [99]:
```
Country_UK=dataset[dataset.Country == "United Kingdom"]
```

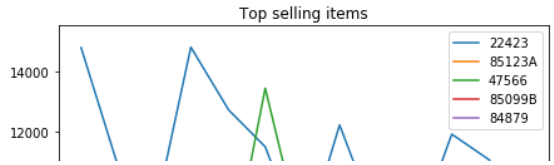
In [100]:
```
Country_UK= dataset[['StockCode','Revenue']].groupby(['StockCode']).sum().sort_values(by='Revenue', ascending=False).reset_ir
Country_UK = Country_UK.head(10)
```

In [101]:
```
#Plotting Revenue for each individual products
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.barh(Country_UK['StockCode'],Country_UK['Revenue'])
ax.set_ylabel('Stock code')
ax.set_xlabel('Revenue')
plt.show()
```



#4 — Top Selling Items and their revenue Contribution per month

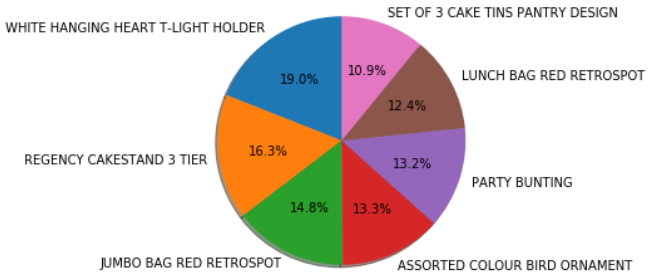
In [105]:
```
#Visualizing top selling items and how they contributed to the revenue every month
fig, ax = plt.subplots(figsize=(7, 7))
Global = dataset[['StockCode','Revenue']].groupby(['StockCode']).sum().sort_values(by='Revenue', ascending=False).reset_index


for c in Global:
    monthly_revenue = dataset[dataset['StockCode'] == c]
    monthly_revenue = monthly_revenue[['Year', 'Month', 'Revenue']].groupby(['Year', 'Month']).sum().reset_index()
    monthly_revenue['Day'] = 1
    monthly_revenue['Date'] = pd.to_datetime(monthly_revenue[['Year', 'Month', 'Day']])
    ax.plot(monthly_revenue['Date'],monthly_revenue['Revenue'], label=c)
    ax.set_ylabel('Revenue')
    ax.set_xlabel('Date')
    ax.legend()
    ax.set_title("Top selling items")
```



#5 — Top 7 Selling Items and their % Contribution to the revenue with each other

In [329]:
```
#Top items with maximum number of sales
Item= dataset[['InvoiceNo','StockCode','Description']].groupby(['StockCode','Description']).count().sort_values(by='InvoiceNo
Item=Item.head(7)
```

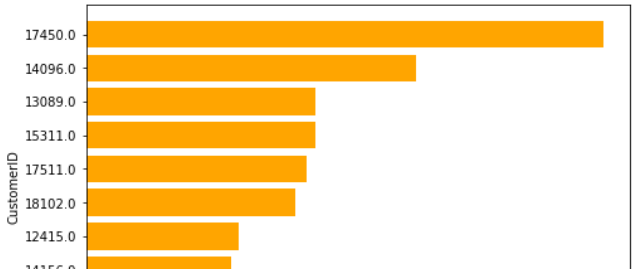
In [330]:
```
fig1, ax1 = plt.subplots()
ax1.pie(Item['InvoiceNo'], labels=Item['Description'], autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')
plt.show()
```

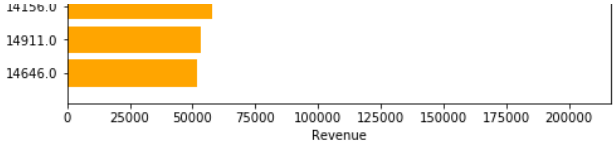


#6 — Top Revenue Contributors wrt Revenue

In [106]:
```
#Top revenue contributors
Customer = dataset[['CustomerID','Revenue']].groupby(['CustomerID']).sum().sort_values(by='Revenue', ascending=False).reset_i
Customer=Customer.head(10)
Customer['CustomerID']=Customer['CustomerID'].astype(str)
```

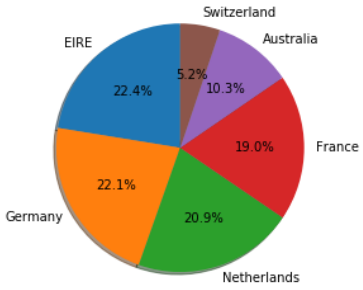
In [108]:
```
#Top customers
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.barh(Customer['CustomerID'], sorted(Customer['Revenue']), color='orange')
ax.set_ylabel('CustomerID')
ax.set_xlabel('Revenue')
plt.show()
```





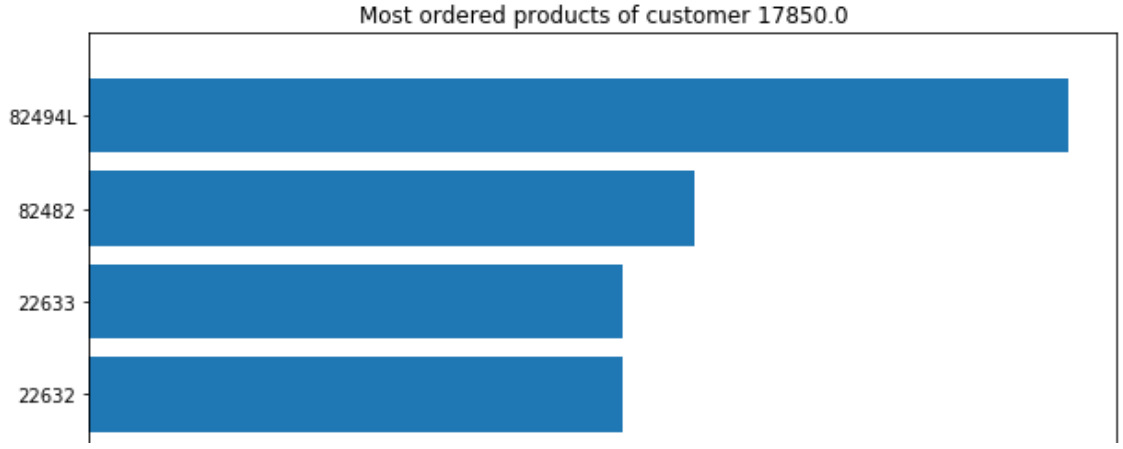
#7— Top 7 Countries and comparison of their % contribution in the total revenue with each other

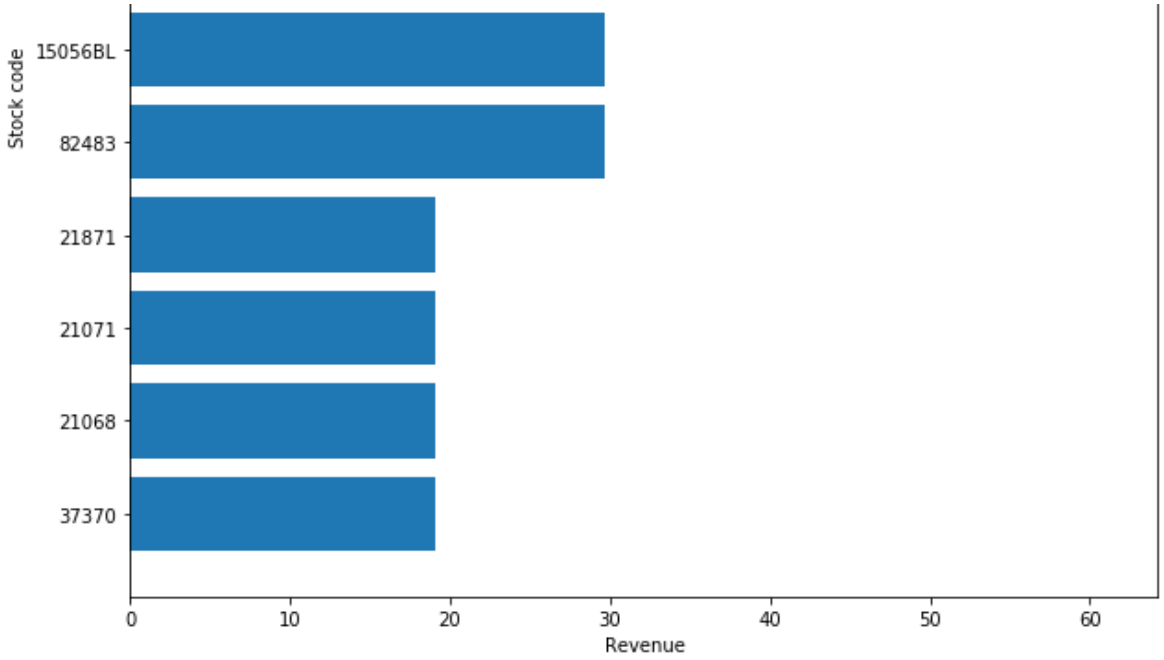
In [96]:
```
#Top 7 countries and how they contribute
Country= dataset[['Revenue','Country']].groupby(['Country']).sum().sort_values(by='Revenue', ascending=False).reset_index()
Country= Country[1:7]
fig1, ax1 = plt.subplots()
ax1.pie(Country['Revenue'], labels=Country['Country'], autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')
plt.show()
```



#8 — Most ordered Products for Customer ID '17850'

In [109]:
```
#Most Ordered Products for Customer ID 17850
data=dataset[0:300]
data['StockCode']=data['StockCode'].astype(str)
data['CustomerID']=data['CustomerID'].astype(str)
def customer_record(customer_id):
    c_data = data[data['CustomerID'] == customer_id ]
    most_ordered = c_data[['StockCode', 'Revenue']].groupby(['StockCode']).sum().sort_values(by='Revenue', ascending=True).re
    most_ordered = most_ordered[0:10]
    fig, ax = plt.subplots(1,1, figsize=(10, 10))
    ax.barh(most_ordered['StockCode'], most_ordered['Revenue'])
    ax.set_ylabel('Stock code')
    ax.set_xlabel('Revenue')
    ax.set_title('Most ordered products of customer ' + str(customer_id))

customer_record(customer_id = '17850.0')
```

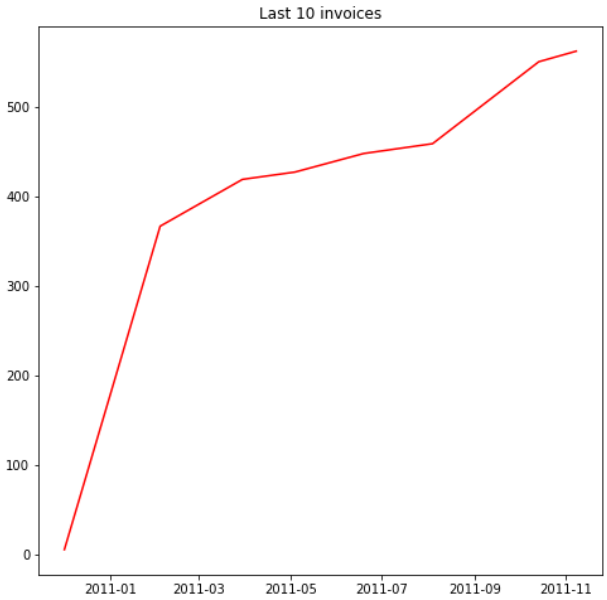




#9 — Revenue Contribution of Last 10 orders of Customer ID '13047' over total time period

In [111]:

```
#Last 10 orders of Customer ID 13047
data1=dataset
data1['CustomerID']=data1['CustomerID'].astype(str)

def customer_record1(customer_id):
    c_data = data1[data1['CustomerID'] == customer_id ]

    last_inv = c_data[['InvoiceNo', 'Revenue', 'Date']].groupby(['Date']).sum().sort_values(by='Revenue').reset_index()
    #print(last_inv)

    fig, ax = plt.subplots(1,1, figsize=(8, 8))
    ax.plot(sorted(last_inv['Date']) , last_inv['Revenue'], color='red')
    ax.set_title('Last 10 invoices')

customer_record1(customer_id = '13047.0')
```



I hope my try at communicating my process of analysis and gathering insights was worth a read. Thank you!

Data Analysis Data Visualization Python Data Science Data