**Customer Segmentation On E-Commerce**

Technical Stacks:

1. Programming Language: Python.
2. Libraries - Pandas, Matplotlib, Seaborn, Plotly, NLTK, Scikit Learn, Word Cloud.
3. IDE: Jupyter Lab, Kaggle.
4. Version Control: Git, GitHub.

Dataset:

This is a transnational data set which contains all the transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retail.The company mainly sells unique all-occasion gifts. Many customers of the company are wholesalers.

The store is UK based and registered. Non-store online retail means the merchandising of goods by means other than retail shops; merchandising by mail order, vending machines, telephone, door-to-door, etc.

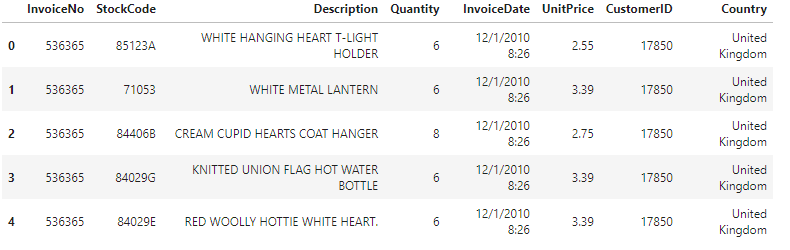
Wholesalers means they are running their own shops, and they are purchasing the gifts from this company in large quantities which will be sold in small quantities in their own store.

NOTE: Per the UCI Machine Learning Repository, this data was made available by Dr Daqing Chen, Director: Public Analytics group. chend '@' lsbu.ac.uk, School of Engineering, London South Bank University, London SE1 0AA, UK.

This dataset contains 8 columns for each entry that correspond to:

* InvoiceNo: Invoice number. Nominal, a 6-digit integral number uniquely assigned to each transaction. If this code starts with letter 'c', it indicates a cancellation.
* StockCode: Product (item) code. Nominal, a 5-digit integral number uniquely assigned to each distinct product.
* Description: Product (item) name. Nominal.
* Quantity: The quantities of each product (item) per transaction. Numeric.
* InvoiceDate: Invoice Date and time. Numeric, the day and time when each transaction was generated.
* UnitPrice: Unit price. Numeric, Product price per unit in sterling.
* CustomerID: Customer number. Nominal, a 5-digit integral number uniquely assigned to each customer.
* Country: Country name. Nominal, the name of the country where each customer resides.

A snapshot of the first 5 rows in the dataset

:The dataset is imported using pandas read\_csv method. In the method, I used the encoding=ISO-8859-1 encoding to properly import the data. At the same time I have converted the data types of ***CustomerID*** and **InvoiceNo** to str data type to be on the safe side.

After loading the dataset, I copied the data to df so that later if needed I can just run this cell instead of loading the whole data again. To make a deep copy, I used copy(deep=True).

The dataset is huge as it contains more than **541K** transaction data.

Furthermore, I checked the column information and found out that there are some missing values as well as the ***InvoiceDate*** column is in string format.

So, I converted the ***InvoiceDate*** to datetime format using the pandas *to\_datetime* function and then dropped the missing values using the CustomerID column with dropna(axis=0, subset=['CustomerID'], inplace=True).

Remove duplicate entries from the dataset. Using the *duplicated* method, I can see which row is a duplicate. To find out how many duplicate rows are present, I am summing the previous result. So, the code is like df.duplicated().sum().

Now, I am deleting the duplicate rows using the drop\_duplicates method and setting inplace=True to permanently modify the original dataframe.

Data Exploration:

First, I explore the **country** column in the dataset.

Here I am trying to know how many transactions happened in each country. As this is a UK-based company, most entries will be from the UK but what about other countries? That is what I am finding.

Many transactions are the same i.e. there are more than or equal to 1 row for each transaction **InvoiceNo** as the data shows different products purchased in each transaction. That is why, I am grouping the unique customers with unique invoice numbers and then taking their countries. I am doing this using the groupby method. The code for this looks like

| df[['CustomerID', 'InvoiceNo', 'Country']].groupby(['CustomerID', 'InvoiceNo', 'Country']).count() |
| --- |

Then, extracting the count of countries in this new dataframe. There are total 37 countries present in the dataset and out of them top 5 countries according to no. of transactions are -

| United Kingdom 19857 Germany 603 France 458 EIRE 319 Belgium 119 |
| --- |

Using the countries data I have plotted a map chart that shows the no. of transactions with colours. Red represents more transactions whereas blue represents less number of transactions in those countries.

From the UK only I found around 20K transactions.

Now, let’s look at the no. of products, customers and total transactions in the dataset.

| *pd.DataFrame([{  'product': df['StockCode'].nunique(),   'customer': df['CustomerID'].nunique(),   'transaction': df['InvoiceNo'].nunique() }], columns=['product', 'customer', 'transaction'], index=['quantity'])* |
| --- |

