STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY

# Customer Segmentation for Online Retail Agency

*Author*
*Nithin Das*

*Instructor*
*M. Daneshmand*

# Content

- Problem Statement
- Software Used
- Data Description
- Data Preprocessing
- Exploratory Data Analysis
- Data Preparation
- Data Normalization
- Finding ‘K’ value
- K means Clustering
- EDA on clusters
- Best Selling Items
- Documentation
- Conclusion

# Problem Statement

- A leading UK based non-store online retail agency wants to segment their customers to improve their marketing strategies

## Why Customer Segmentation?

- A customer segmentation model allows for the effective allocation of marketing resources and the maximization of cross and up-selling opportunities.

## How is it Achieved?

- Customer segments are usually determined on similarities, such as personal characteristics, preferences or behaviors that should correlate with the same behaviors that drive customer profitability.

# Software Used

- Python
- IDE: Jupyter Notebook
- Packages: Pandas, Numpy, matplotlib, Sklearn

# Data Description

- Total Records = 541909
- Variables = `'InvoiceNo', 'StockCode', 'Description', 'Quantity', 'InvoiceDate', 'UnitPrice', 'CustomerID', 'Country'`

```
:   1  retail_data.dtypes

285]: InvoiceNo       object
      StockCode       object
      Description     object
      Quantity         int64
      InvoiceDate     object
      UnitPrice      float64
      CustomerID     float64
      Country         object
      dtype: object
```

# Data Preprocessing

- The *'Description'* variable has 1454 missing *values and 'CustomerID' has 135080 missing values*
- *We will remove these missing values*

```
In [277]:  1  retail_data.isnull().sum()

Out[277]: InvoiceNo          0
          StockCode          0
          Description     1454
          Quantity           0
          InvoiceDate        0
          UnitPrice          0
          CustomerID     135080
          Country            0
          dtype: int64
```

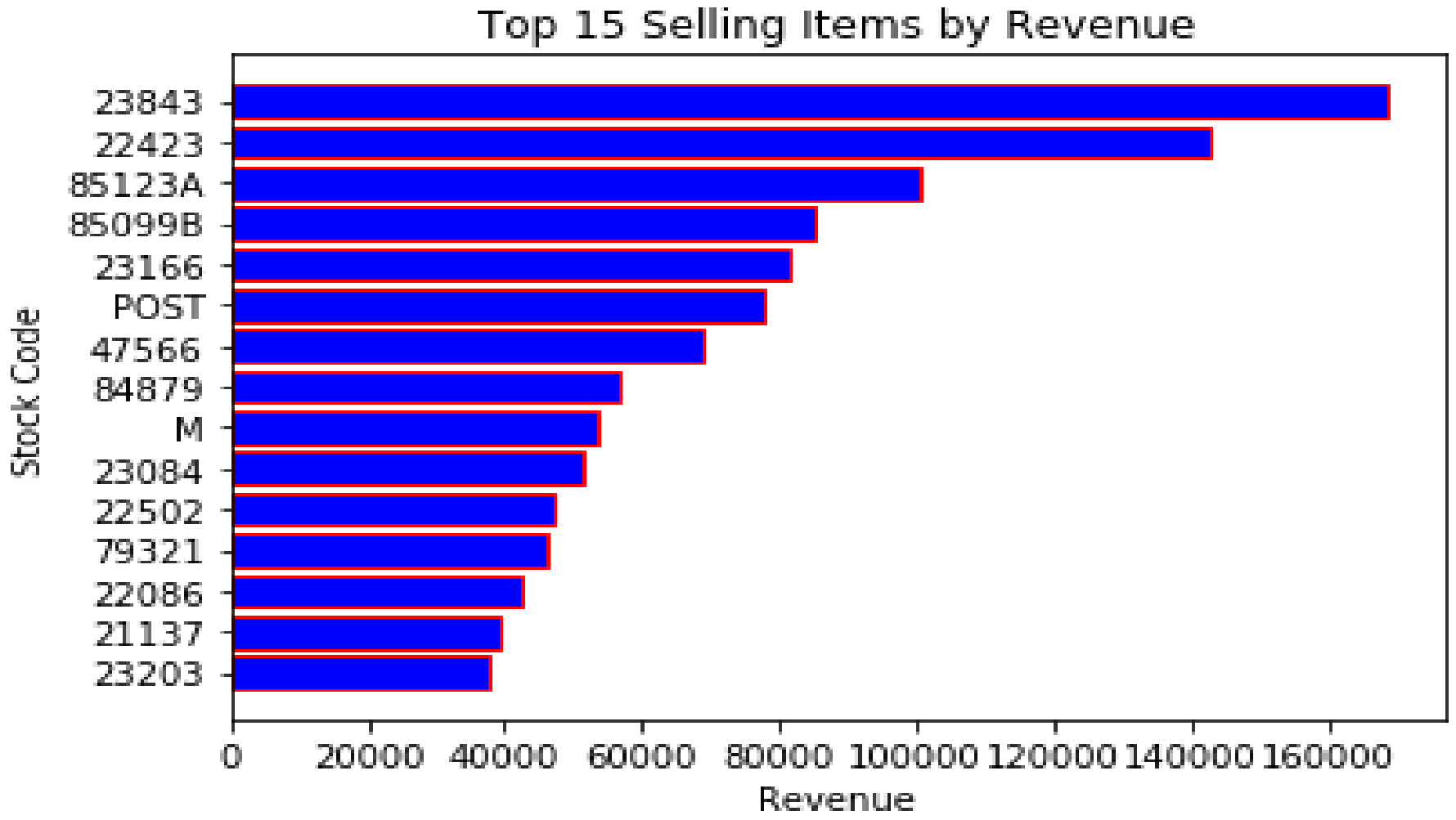
```
1  np.unique(retail_data['InvoiceNo'])

43]: array(['536365', '536366', '536367', ..., 'C581499', 'C581568', 'C581569'],
           dtype=object)
```

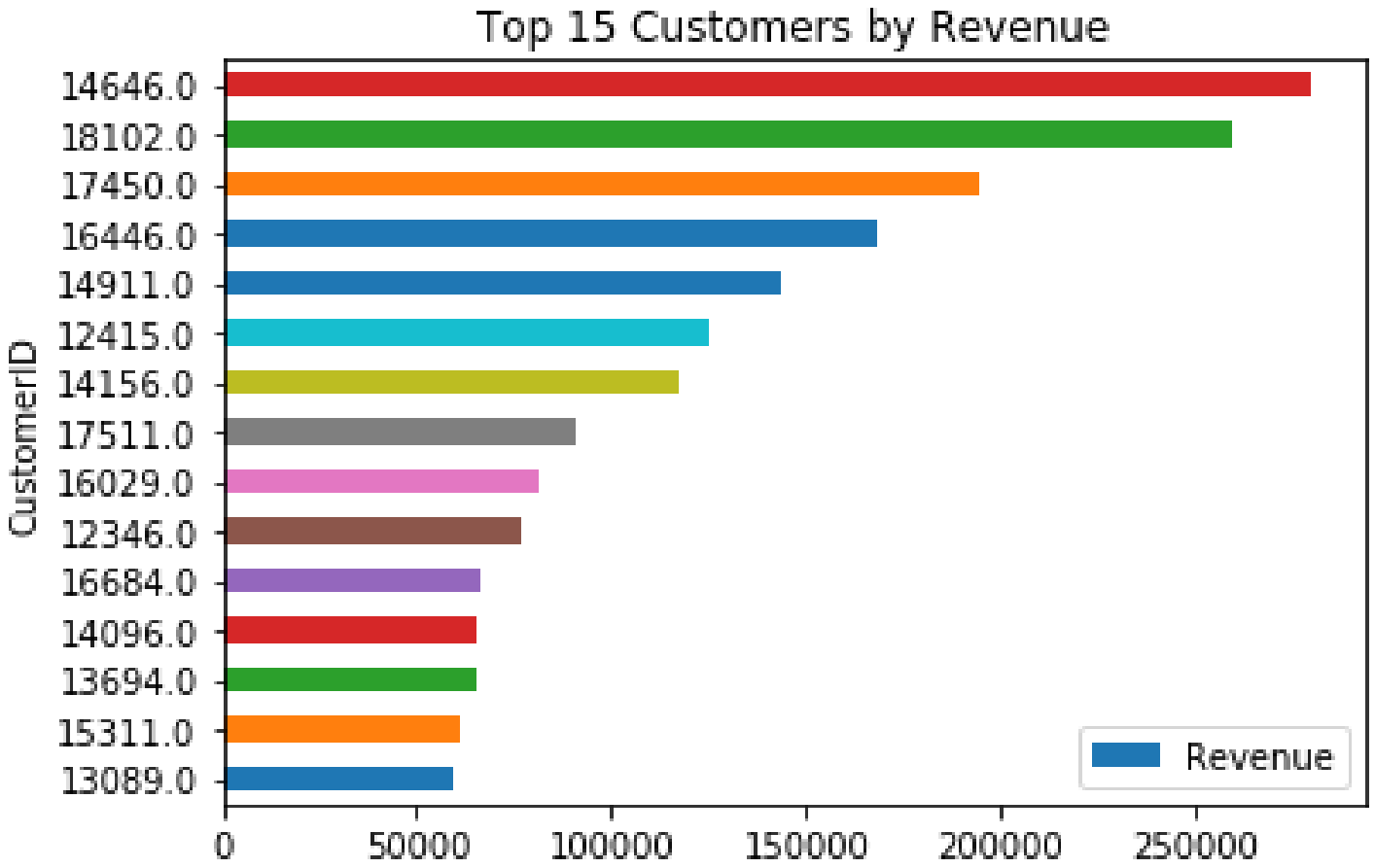
- *The* "Invoice" *variable has values that starts with "C", which Denotes cancelled transactions. We will have to remove such Transactions to derive meaningful insights*
- *After the above steps, we are left with 397924 records*
- *We can also calculate Revenue as "Quantity" * "Unit Price"*

```
1  #Calculate Revenue
2  retail_data['Revenue']= retail_data['Quantity']*retail_data['UnitPrice']
```

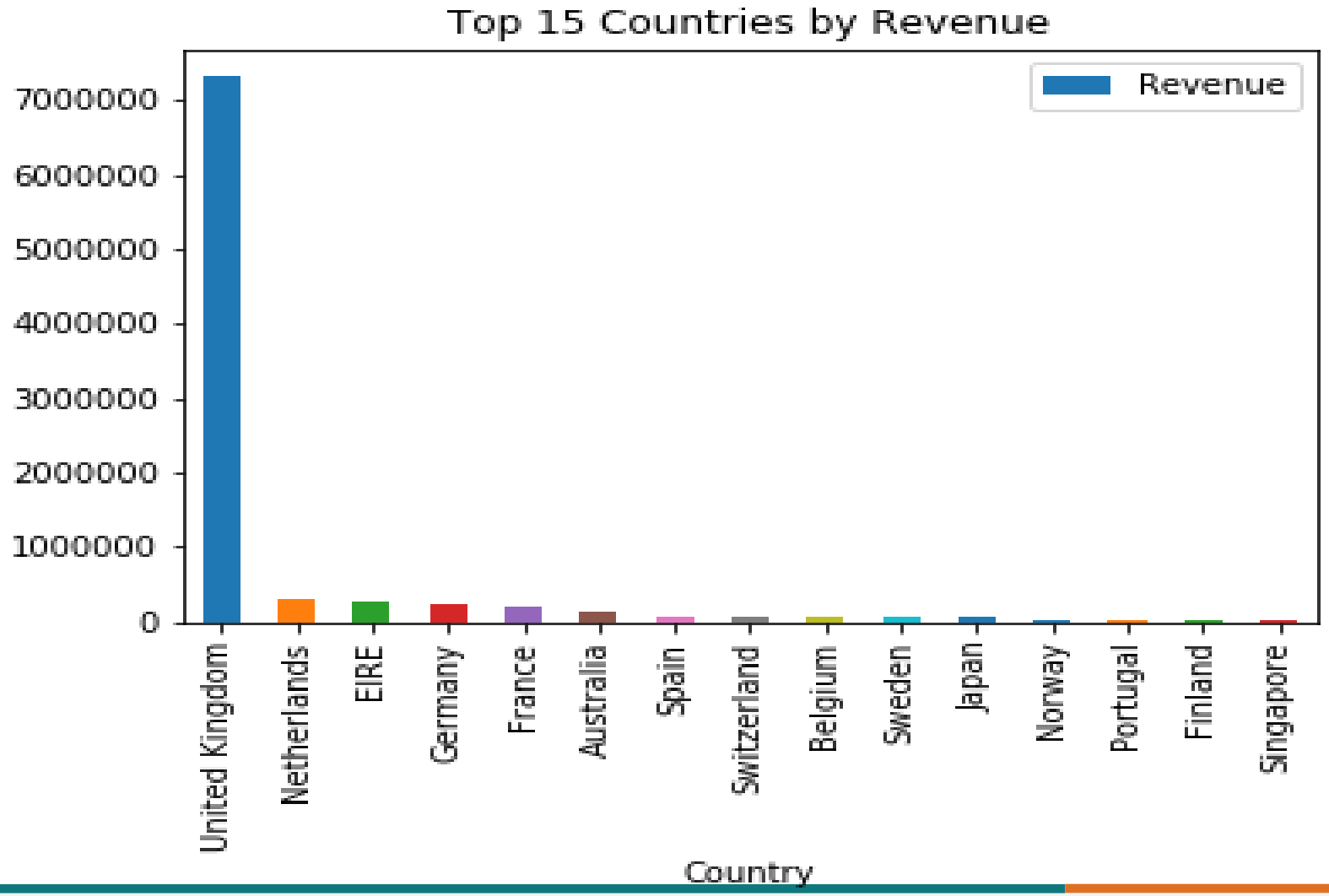
# Exploratory Data Analysis



# EDA (continued)



# EDA (continued)



# Data Preparation

- For customer segmentation, we have to create characteristics for customers
- Therefore we aggregate *"Revenue"* and "Invoice No"

to get Total Revenue and Order Count for each customers

- We can also calculate "Average Order Value"

for each customers as

"Avg. order value"= "Total Revenue"/ "Order Count"

```
1  # use groupby to aggregate sales by CustomerID
2  customer_df = retail_data.groupby('CustomerID').agg({'Revenue': sum,
3                                                        'InvoiceNo': lambda x: x.nunique()})
4
5  # Select the columns we want to use
6  customer_df.columns = ['TotalRevenue', 'OrderCount']
7
8  # create a new column 'AvgOrderValu'
9  customer_df['AvgOrderValue'] = customer_df['TotalRevenue'] / customer_df['OrderCount']
10
11 customer_df.head()
```

61]:

| CustomerID | TotalRevenue | OrderCount | AvgOrderValue |
|---|---|---|---|
| 12346.0 | 77183.60 | 1 | 77183.600000 |
| 12347.0 | 4310.00 | 7 | 615.714286 |
| 12348.0 | 1797.24 | 4 | 449.310000 |
| 12349.0 | 1757.55 | 1 | 1757.550000 |
| 12350.0 | 334.40 | 1 | 334.400000 |

# Data Normalization

- Before clustering, it is important to normalize the data since the customer characteristics may be measure by variables with different units.
- Therefore we will normalize “TotalRevenue”,

“Order Count” and “ Avg Order Value”

```
1  # Data Normalization
2
3  rank_df = customer_df.rank(method='first')
4  normalized_df = (rank_df - rank_df.mean()) / rank_df.std()
5  normalized_df.head(10)
6
```

52]:

| CustomerID | TotalRevenue | OrderCount | AvgOrderValue |
|---|---|---|---|
| 12346.0 | 1.724268 | -1.731452 | 1.730654 |
| 12347.0 | 1.464031 | 1.173460 | 1.319544 |
| 12348.0 | 0.929189 | 0.533246 | 0.938768 |
| 12349.0 | 0.906837 | -1.730654 | 1.681161 |
| 12350.0 | -0.750376 | -1.729856 | 0.336073 |
| 12352.0 | 1.171863 | 1.287613 | 0.171628 |
| 12353.0 | -1.638853 | -1.729057 | -1.571798 |
| 12354.0 | 0.486946 | -1.728259 | 1.610115 |
| 12355.0 | -0.407119 | -1.727461 | 0.977085 |
| 12356.0 | 1.252489 | 0.127724 | 1.558227 |

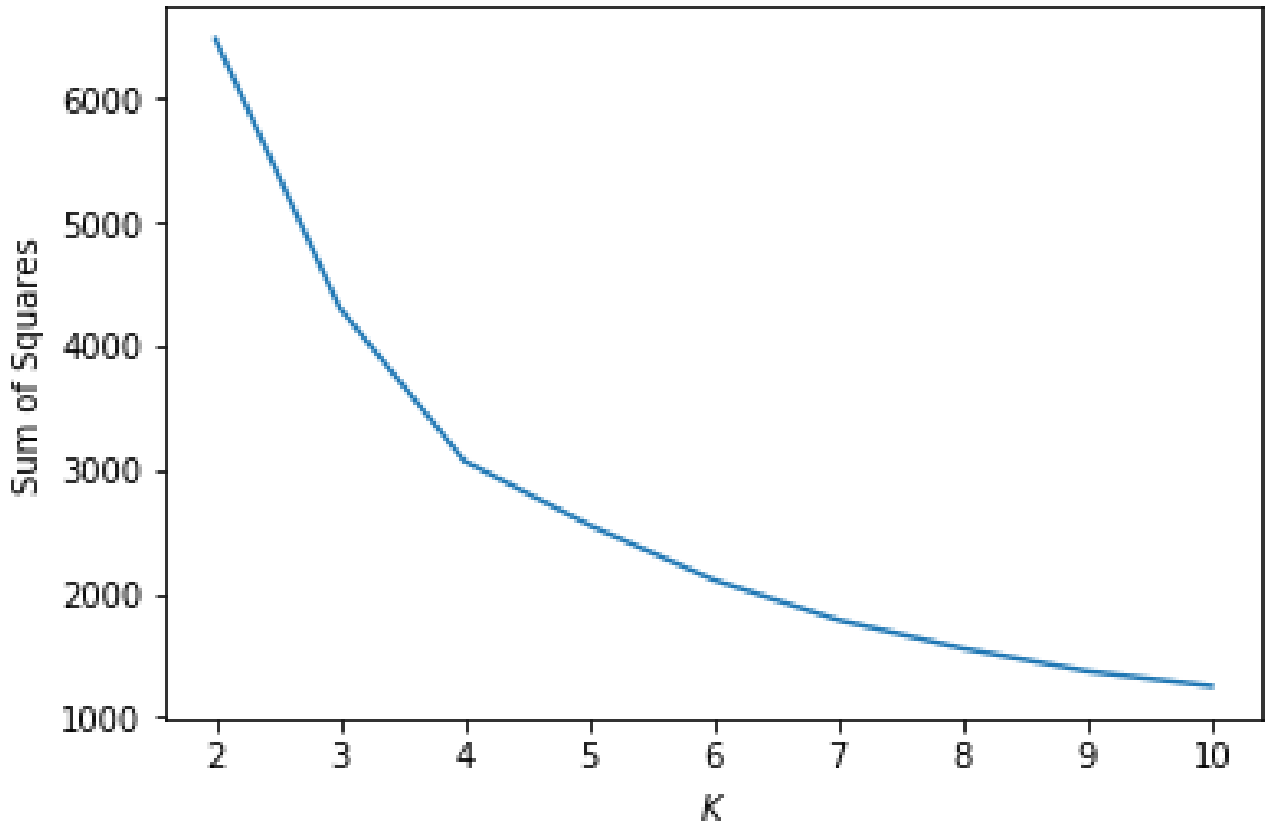
# Finding ‘K’ value for Kmeans

- Elbow Method:

- After k=4, the curve tends

to get flatter

- Therefore, e will set total clusters equals 4



# K means Clustering

- Each customers are assigned in to one of the 4 clusters

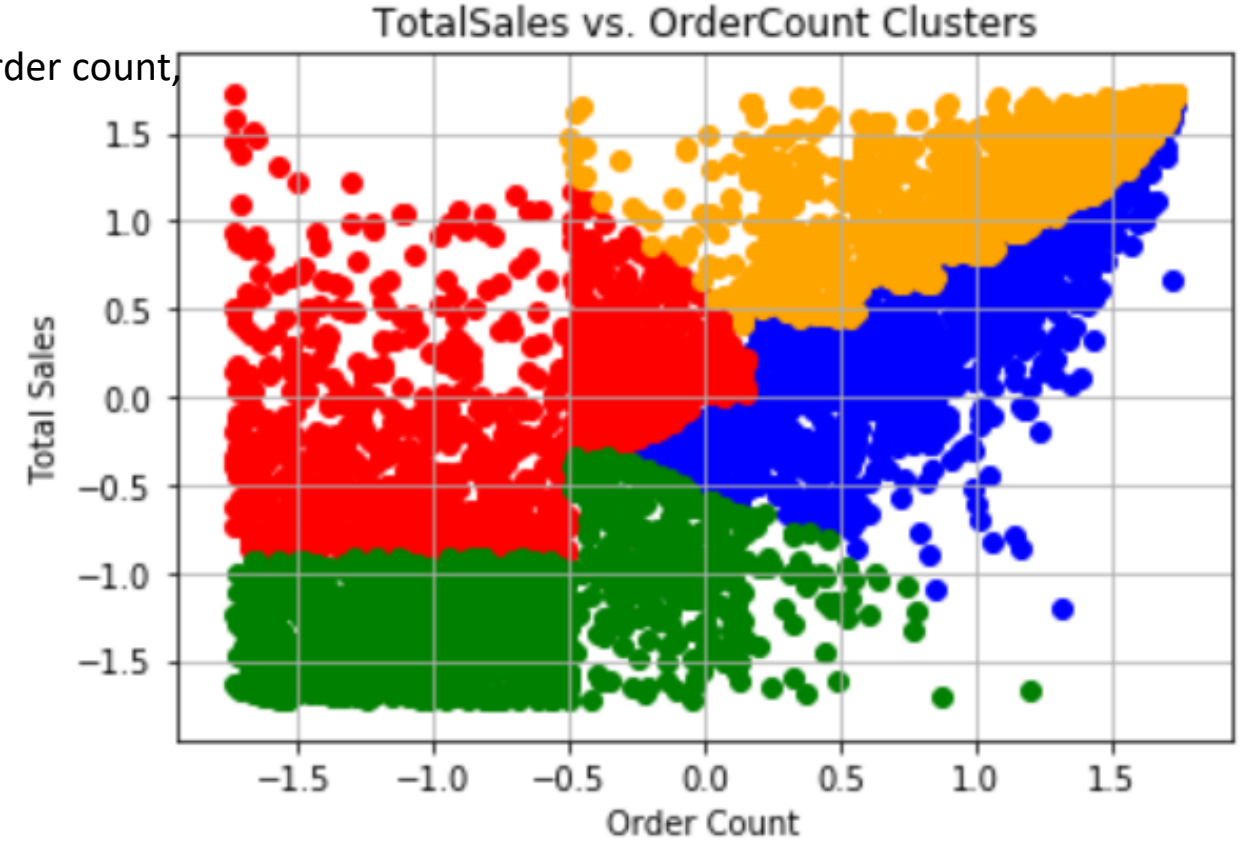
```
1  kmeans = KMeans(n_clusters=4).fit(normalized_df[['TotalRevenue', 'OrderCount', 'AvgOrderValue']])
2
3  four_cluster_df = normalized_df[['TotalRevenue', 'OrderCount', 'AvgOrderValue']].copy(deep=True)
4  four_cluster_df['Cluster'] = kmeans.labels_
5
6  four_cluster_df.head(10)
```

]:

| CustomerID | TotalRevenue | OrderCount | AvgOrderValue | Cluster |
|---|---|---|---|---|
| **12346.0** | 1.724268 | -1.731452 | 1.730654 | 2 |
| **12347.0** | 1.464031 | 1.173460 | 1.319544 | 1 |
| **12348.0** | 0.929189 | 0.533246 | 0.938768 | 1 |
| **12349.0** | 0.906837 | -1.730654 | 1.681161 | 2 |
| **12350.0** | -0.750376 | -1.729856 | 0.336073 | 2 |
| **12352.0** | 1.171863 | 1.287613 | 0.171628 | 1 |
| **12353.0** | -1.638853 | -1.729057 | -1.571798 | 0 |
| **12354.0** | 0.486946 | -1.728259 | 1.610115 | 2 |
| **12355.0** | -0.407119 | -1.727461 | 0.977085 | 2 |
| **12356.0** | 1.252489 | 0.127724 | 1.558227 | 1 |

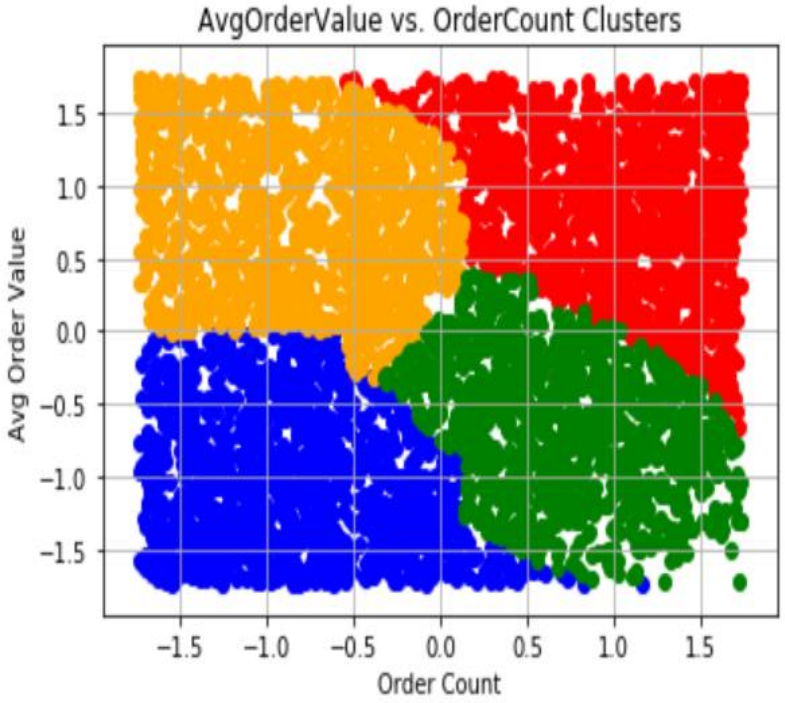
# EDA on obtained clusters

- The customers in green have low total sales/revenue and low order count, meaning they are all-around low-value customers. On the other hand, the customers in orange have high total sales and high order counts, indicating they are the highest value customers.



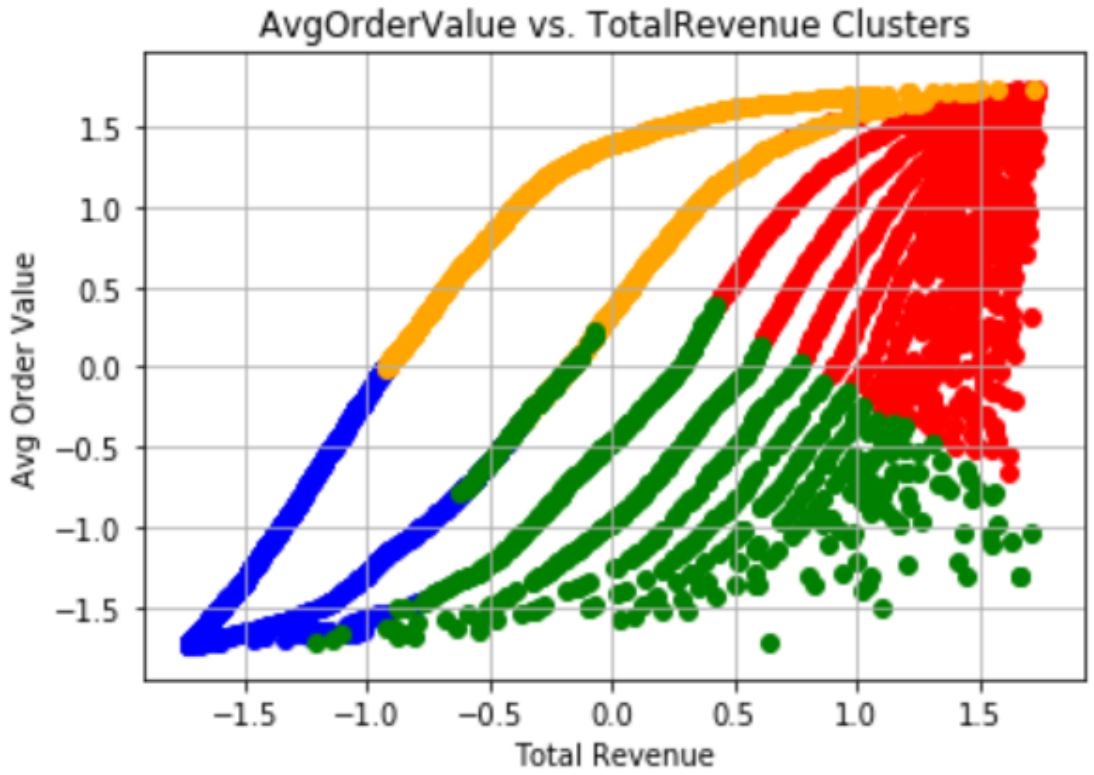
# EDA on  the clusters

- The customers in blue are the lowest value customers and the customers in red are the highest value customers.
- We could look at the customers in the orange cluster and attempt to find ways to increase their order count with email reminders or SMS push notifications targeted based on some other identifying factors.
- Likewise, with customers in the green segment, you might want to try some cross-selling and up-selling techniques at the cart.



# EDA on obtained clusters

- In this plot, we have the average order value versus total revenue clusters. This plot further substantiates the previous 2 plots in identifying the red cluster as the highest value customers, blue as the lowest value customers, and the green and orange as high opportunity customers.



# Best selling Item

- Based on this information, we now know that the *Jumbo Bag Red Retrospot* is the best-selling item for our highest-value cluster.
- We can also make recommendations of ***Other Items You Might Like*** to customers within this segment.

```
high_value_cluster = four_cluster_df.loc[four_cluster_df['Cluster'] == 1]

pd.DataFrame(retail_data.loc[retail_data['CustomerID'].isin(high_value_cluster.index)].groupby(
    'Description').count()['StockCode'].sort_values(ascending=False).head())
```

| Description | StockCode |
|---|---|
| JUMBO BAG RED RETROSPOT | 1191 |
| WHITE HANGING HEART T-LIGHT HOLDER | 1126 |
| REGENCY CAKESTAND 3 TIER | 1111 |
| LUNCH BAG RED RETROSPOT | 976 |
| PARTY BUNTING | 886 |

# Documentation

- https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a
- https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/

# Conclusion

- Application of K-means clustering based on the purchasing behaviors of historical customers.
- These analysis are useful to recommend more items to the customers under categories like ‘Items You might Like’, ‘Recommendations based on your Usage’ etc.

# Thanks!