Capstone Project- Online Retail Store Data Project

Submitted By- Diwakar Acharya

Table of Contents

Problem Statement

An online retail store is trying to understand the various customer purchase patterns for their firm, we are required to give enough evidence based insights to provide the same.

Project Objective

1) With the project we have to come up with the useful insights about the customer purchasing history that can be an added advantage for the online retailer.
2) Segment the customers based on their purchasing history.

Data Description

Given data is Online Retail Store data in CSV format. It is having 541909 rows and 8 columns. These columns represent features name.

Feature Name	Description
Invoice	Invoice number
StockCode	Product ID
Description	Product Description
Quantity	Quantity of the product
InvoiceDate	Date of the invoice
Price	Price of the product per unit
CustomerID	Customer ID
Country	Region of Purchase

Insights from the data:

1) Basic details of data

```
retail_store.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom

2) All statistical figure

```
retail_store.describe()
```

	Quantity	UnitPrice	CustomerID
count	541909.000000	541909.000000	406829.000000
mean	9.552250	4.611114	15287.690570
std	218.081158	96.759853	1713.600303
min	-80995.000000	-11062.060000	12346.000000
25%	1.000000	1.250000	13953.000000
50%	3.000000	2.080000	15152.000000
75%	10.000000	4.130000	16791.000000
max	80995.000000	38970.000000	18287.000000

3) Correlation between features:

```
retail_store.corr()
```

	Quantity	UnitPrice	CustomerID
Quantity	1.000000	-0.001235	-0.00360
UnitPrice	-0.001235	1.000000	-0.00456
CustomerID	-0.003600	-0.004560	1.00000

4) **Maximum**

```
retail_store.max()
```

```
C:\Users\Diwakar\AppData\Local\T
e reductions (with 'numeric_only
umns before calling the reductio
  retail_store.max()
```

```
InvoiceNo            C581569
StockCode                  m
Quantity               80995
InvoiceDate    9/9/2011 9:52
UnitPrice           38970.0
CustomerID          18287.0
Country         Unspecified
dtype: object
```

1) **Statistics insights that can be used by online retailer to improve in various areas:**

I have analysed below task and basis insights is provided.

1) **Popular products:** The most frequently purchased products can be identified by analysing the product description field. These are the products that the stor e should keep in stock and possibly promote more.

popular_products = retail_store.groupby("StockCode")["Quantity"].sum().sort_ values(ascending=False)
print(popular_products)

Most popular products is “84077” and least popular is “84347”.

```
StockCode
84077     53215
22197     48712
85099B    45066
84879     35314
85123A    34204
          ...
21144       -12
CRUK        -16
21645       -24
D         -1194
84347     -1460
```

2) **Sales by country:** The country field can be used to determine which countries generate the most revenue for the store. This information can be used to targe t marketing efforts in those countries.

```
sales_by_country = retail_store.groupby("Country")["Quantity"].sum().sort_val
ues(ascending=False)
print(sales_by_country)
```

Most of the sales is coming from United Kingdom and least sales coming from Saudi Arabia.

```
Country
United Kingdom          4008533
Netherlands              200128
EIRE                     136329
Germany                  117448
France                   109848
Australia                 83653
Sweden                    35637
Switzerland               29778
Spain                     26824
Japan                     25218
Belgium                   23152
Norway                    19247
Portugal                  16044
Finland                   10666
Channel Islands            9479
Denmark                    8188
Italy                      7999
Cyprus                     6317
Singapore                  5234
Austria                    4827
Israel                     3990
Poland                     3653
Canada                     2763
Iceland                    2458
Unspecified                1789
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
```

3) **Customer behaviour:** By analysing the quantity and total cost fields, patterns in customer behaviour can be identified such as how often they make purchas es and how much they spend on average.

```
purchase_frequency = retail_store.groupby("CustomerID")["InvoiceNo"].nunique().sort_values(ascending=False)
average_spend = retail_store.groupby("CustomerID")["Quantity"].sum().sort_values(ascending=False)
print(purchase_frequency)
print(average_spend)
```

Customer “14911” is the most frequent customer and most spent is being done by customer “196719”.

```
CustomerID
14911.0    248
12748.0    224
17841.0    169
14606.0    128
13089.0    118
           ...
13877.0      1
16400.0      1
13878.0      1
13886.0      1
13670.0      1
Name: InvoiceNo, Length: 4372, dtype: int64
CustomerID
14646.0    196719
12415.0     77242
14911.0     77180
17450.0     69029
18102.0     64122
            ...
16252.0      -158
16742.0      -189
14213.0      -244
15823.0      -283
16546.0      -303
Name: Quantity, Length: 4372, dtype: int64
```

4) **Time-based analysis:** By analysing the date field, patterns in sales can be id entified over time. This information can be used to predict future sales trends a nd plan inventory accordingly.

```
retail_store['InvoiceDate'] = pd.to_datetime(retail_store['InvoiceDate'])
sales_by_month = retail_store.groupby(retail_store['InvoiceDate'].dt.strftime('%B'))['Quantity'].sum().sort_values(ascending=False)
print(sales_by_month)
```

Sales is highest in November and lowest in February.

```
InvoiceDate
November     669915
October      569666
September    537496
December     500198
August       386612
May          367852
July         363418
June         356922
March        344012
April        278585
January      269379
February     262833
Name: Quantity, dtype: i
```

5) **Stock forecasting:** By analyzing the sales data, patterns in stock demand ca n be identified. This information can be used to predict future stock needs and plan inventory accordingly.

```
stock_forecast = retail_store.groupby("StockCode")["Quantity"].sum().sort_values(as
cending=False)
print(stock_forecast)
```

```
StockCode
84077     53215
22197     48712
85099B    45066
84879     35314
85123A    34204
          ...
21144       -12
CRUK        -16
21645       -24
D         -1194
84347     -1460
Name: Quantity, Le
```

2. Customer Segmentation

```
retail_store["segment"] = "low value"

retail_store.loc[retail_store["UnitPrice"] >= 50, "segment"] = "high value"

purchase_counts = retail_store.groupby("CustomerID")["UnitPrice"].count()

purchase_counts = purchase_counts.reset_index()

purchase_counts.columns = ["CustomerID", "purchase_counts"]

retail_store = pd.merge(retail_store, purchase_counts, on="CustomerID")
```

```
retail_store.loc[retail_store["purchase_counts"] >= 10, "segment"] = "frequent" + " " + retail_store["segment"]

retail_store.loc[retail_store["purchase_counts"] < 3, "segment"] = "infrequent" + " " + retail_store["segment"]

print(retail_store[["CustomerID", "segment"]])
```

Output:

```
         CustomerID             segment
0           17850.0  frequent low value
1           17850.0  frequent low value
2           17850.0  frequent low value
3           17850.0  frequent low value
4           17850.0  frequent low value
...             ...                 ...
406824      12713.0  frequent low value
406825      12713.0  frequent low value
406826      12713.0  frequent low value
406827      12713.0  frequent low value
406828      12713.0  frequent low value

[406829 rows x 2 columns]
```

Data pre-processing Steps and Inspiration

1) Data cleaning- The data can have many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling of missing data, noisy data etc
   Here, we have huge percentage approx. (25%) of CustomerID which is blank.

```
retail_store.isnull().sum()
```

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

2) Outliers- Identifying and removing outliers

```
import seaborn as sns
fig, axs = plt.subplots(2,figsize=(6,18))
X = retail_store[['UnitPrice']]
for i,column in enumerate(X):
    sns.boxplot(retail_store[column], ax=axs[i], width=0.8)

# drop the outliers
without_outlier = retail_store[(retail_store['UnitPrice']>5000)]
without_outlier
```

3) Data integration: Combining multiple data sources- Here, there is only one source i.e. retail_store.csv
4) Data transformation: Normalizing or aggregating data- We will do this during model building.
5) Data reduction: Reducing the amount of data through sampling or feature selection- We have train and test set.
6) Data encoding: Encoding categorical data as numerical data to be used in machine learning models.

Choosing the Algorithm for the project

I have chosen 2 algorithm 1) K-means 2) Hierarchical- clustering

1) **K-means-**

Below code first selects the relevant features for clustering, which are "Quantity" and "UnitPrice" in this example. Then it standardizes the data by subtracting the mean and dividing by the standard deviation to ensure that the algorithm is not affected by the scale of the data. Then it runs the K-means algorithm with 3 clusters, which is the number of clusters we want to segment the customers into.

The labels generated by the algorithm are then added to the dataframe as segment column, and the final segmented data containing CustomerID and segment is printed.

```
from sklearn.cluster import KMeans

from sklearn.preprocessing import StandardScaler


# select relevant features for clustering

features = retail_store[["Quantity", "UnitPrice"]]


# standardize the data

scaler = StandardScaler()

scaled_data = scaler.fit_transform(features)


# run k-means clustering

kmeans = KMeans(n_clusters=3)

kmeans.fit(scaled_data)


# assign cluster labels to customers

retail_store["segment"] = kmeans.labels_


# print the segmented data

print(retail_store[["CustomerID", "segment"]])
```

```
        CustomerID  segment
0          17850.0        0
1          17850.0        0
2          17850.0        0
3          17850.0        0
4          17850.0        0
...            ...      ...
406824     12713.0        0
406825     12713.0        0
406826     12713.0        0
406827     12713.0        0
406828     12713.0        0

[406829 rows x 2 columns]
```

2) Hierarchical clustering

```
from sklearn.cluster import AgglomerativeClustering
from sklearn.preprocessing import StandardScaler

# select relevant features for clustering
features = retail_store[["Quantity", "UnitPrice"]]

# standardize the data
scaler = StandardScaler()
scaled_data = scaler.fit_transform(features)

# run hierarchical clustering
agg_clustering = AgglomerativeClustering(n_clusters=3)
agg_clustering.fit(scaled_data)

# assign cluster labels to customers
retail_store["segment"] = agg_clustering.labels_

# print the segmented data
print(retail_store[["CustomerID", "segment"]])
```

Motivation and Reasons for choosing the Algorithm

The motivation for choosing k-means algorithm in this case could be because:

1) K-means is a popular algorithm for customer segmentation because it is simple to implement and easy to interpret. It is a form of unsupervised learning, which means it does not require labeled data, so it can be used with datasets that do not have predefined groups.

2) One of the main advantages of K-means is that it is computationally efficient and can handle large datasets. It is also a versatile algorithm that can be used for various types of data such as continuous, categorical, or mixed data.

3) Another advantage of K-means is that it produces clear and well-defined clusters, which makes it easy to interpret the results and assign customers to specific segments. This can be useful for creating targeted marketing campaigns, improving customer service, and optimizing pricing strategies.

4) In addition, K-means is a hard clustering algorithm, which means each data point belongs to only one cluster. This can be useful when we have a clear idea of how many segments we want to create and when we want to assign each customer to a specific segment.

In summary, K-means is a powerful algorithm that can be used to segment customers based on their purchase history, demographics, or other characteristics. It is computationally efficient, easy to interpret, and produces clear and well-defined clusters. This makes it a good choice for customer segmentation when you have a clear idea of how many segments you want to create and when you want to assign each customer to a specific segment.

Assumptions

The k-means algorithm makes the following assumptions:

1) The data points are homogeneous and isotropic, meaning that they have similar variances and are evenly distributed in all directions.

2) The clusters are spherical and have similar variances.

3) The number of clusters (k) is known in advance.

4) The data points are independent and identically distributed.

5) The algorithm converges to the global optimum, although this is not guaranteed in practice.

Model Evaluation and techniques

Model evaluation and techniques are important steps in the process of building and using a machine learning model. They help to ensure that the model is accurate, reliable, and generalizable to new data.

For evaluating a k-means model used for customer segmentation based on purchasing history, some common techniques include:

1) Silhouette Score: It measures how similar an object is to its own cluster compared to other clusters. The score ranges from -1 to 1, with higher values indicating a better fit.
2) Calinski-Harabasz Index: It measures the ratio of the between-cluster variance to the within-cluster variance. A higher value indicates a better fit.

3) Davies-Bouldin Index: It measures the average similarity between each cluster and its most similar cluster. Lower values indicate a better fit.

4) Elbow method: It is used to determine the optimal number of clusters. The method plots the within-cluster sum of squares (WCSS) against the number of clusters and the optimal number of clusters is chosen at the "elbow" point of the plot.

5) Visualization: Visualizing the data points and the clusters can also provide insights into the performance of the model and help identify any issues or areas for improvement.

Inferences from the same

From the evaluation of a k-means model can provide valuable insights into the performance of the model and the characteristics of the segments. Here are some possible inferences that can be drawn from the evaluation of a k-means model:

1) High silhouette score or Calinski-Harabasz index and low Davies-Bouldin index indicate that the model has found well-defined and distinct clusters.

2) Low silhouette score or Calinski-Harabasz index and high Davies-Bouldin index indicate that the model has found overlapping or poorly defined clusters.

3) A high value of WCSS (Within-cluster sum of squares) suggests that the clusters are far away from the centroid and data points are scattered around the centroid.

4) A low value of WCSS suggests that the clusters are tightly packed around the centroid and data points are concentrated around the centroid.

5) Visualization of the clusters can help identify any outliers or anomalies in the data, as well as any patterns or trends in the data that may not be apparent from the numerical evaluation metrics.

6) Cluster completeness and purity can be used to evaluate the quality of the clusters. High completeness and purity imply that the cluster is composed of similar data points and different clusters are composed of different data points.

7) Adjusted Rand index can be used to compare the similarity of two different partitions of the same data set. A high index value indicates that the two partitions are similar, while a low value indicates that they are dissimilar.

8) The number of clusters chosen by the Elbow method can be used to determine the optimal number of clusters, which can then be used as a guide for future analyses.

Future possibilities of the project

There are several future possibilities for an online retail store using k-means:

1) Personalized Recommendations: By segmenting customers based on their purchasing history, an online retail store can offer personalized recommendations to each group of customers, increasing the likelihood of repeat purchases.

2) Targeted Marketing: By understanding the characteristics of different customer segments, an online retail store can target specific groups of customers with relevant marketing messages and promotions, increasing the efficiency and effectiveness of their marketing efforts.

3) Inventory Management: By identifying patterns in customer purchasing behavior, an online retail store can anticipate the demand for certain products and optimize their inventory management to avoid stockouts or overstocking.

4) Up-selling and cross-selling: By understanding the purchase patterns of different segments, an online retail store can recommend related or complimentary products to customers, increasing the average order value.

5) Fraud Detection: By identifying patterns of abnormal behavior, an online retail store can detect and prevent fraudulent activities.

6) Churn Prediction: By identifying patterns of behavior that are associated with customers who are likely to stop shopping, an online retail store can take proactive steps to retain these customers before they leave.

7) Customer Lifetime Value prediction: By understanding the behavior patterns of different customer segments, an online retail store can predict the lifetime value of each customer and accordingly design retention and acquisition strategies.

8) Customer Segmentation can also be used to identify the most profitable customer segments and focus on retaining and acquiring more customers in those segments.

Conclusion:

In this project, using k-means for customer segmentation in an online retail store can provide valuable insights into customer behaviour and enable the store to personalize its offerings, improve its marketing efforts, optimize its inventory management, detect fraud, and predict churn.

By understanding the characteristics of different customer segments, an online retail store can make data-driven decisions that lead to increased sales and customer satisfaction.

Additionally, k-means can also be used to predict customer lifetime value and identify profitable customer segments, allowing the store to focus on retaining and acquiring more customers in those segments.

However, it's important to keep in mind that k-means is an unsupervised learning algorithm, so prior knowledge of the dataset is needed to make sure the assumptions are met and the results are meaningful.

References

- Intellipaat live session recording & notes
- Wikipedia
- Google.com
- https://www.analyticsvidhya.com/blog/2021/05/k-means-clustering
- Geeksforgeeks.com
- https://towardsdatascience.com/customer-segmentation-using-k-means-clustering