**CUSTOMER SEGMENTATION USING DATA SCIENCE**

**PHASE 3: DEVELOPMENT PART -1**

**TEAM MEMBER**

**Priya kumari - 3105211041077**

**Implementing Customer Segmentation Using Machine Learning [Beginners Guide]**

These days, you can personalize everything. There’s no one-size-fits-all approach. But, for business, this is actually a great thing. It creates a lot of space for healthy competition and opportunities for companies to get creative about how they acquire and retain customers.

One of the fundamental steps towards better personalization is customer segmentation. This is where personalization starts, and proper segmentation will help you make decisions regarding new features, new products, pricing, marketing strategies, even things like in-app recommendations.

**What is customer segmentation**

Customer segmentation simply means grouping your customers according to various characteristics (for example grouping customers by age).

It’s a way for organizations to understand their customers. Knowing the differences between customer groups, it’s easier to make strategic decisions regarding product growth and marketing.

**The opportunities to segment are endless and depend mainly on how much customer data you have at your use.** Starting from the basic criteria, like gender, hobby, or age, it goes all the way to things like “time spent of website X” or “time since user opened our app”.

There are different methodologies for customer segmentation, and they depend on four types of parameters:

* **geographic,**
* **demographic,**
* **behavioral,**
* **psychological.**

**Geographic:** customer segmentation is very simple, it’s all about the user’s location. This can be implemented in various ways. You can group by country, state, city, or zip code.

**Demographic :** segmentation is related to the structure, size, and movements of customers over space and time. Many companies use gender differences to create and market products. Parental status is another important feature. You can obtain data like this from customer surveys.

**Behavioral :** customer segmentation is based on past observed behaviors of customers that can be used to predict future actions. For example, brands that customers purchase, or moments when they buy the most. The behavioral aspect of customer segmentation not only tries to understand reasons for purchase but also how those reasons change throughout the year.

**Psychological :**segmentation of customers generally deals with things like personality traits, attitudes, or beliefs. This data is obtained using customer surveys, and it can be used to gauge customer sentiment.

**Advantages of customer segmentation:**

Implementing customer segmentation leads to plenty of new business opportunities. You can do a lot of optimization in:

* **budgeting,**
* **product design,**
* **promotion,**
* **marketing,**
* **customer satisfaction**.

Let’s discuss these benefits in more depth.

* **Budgeting**

Nobody likes to invest in campaigns that don’t generate any new customers. Most companies don’t have huge marketing budgets, so that money has to be spent right. Segmentation enables you to target customers with the highest potential value first, so you get the most out of your marketing budget.

* **Product design**

Customer segmentation helps you understand what your users need. You can identify the most active users/customers, and optimize your application/offer towards their needs.

* **Promotion**

Properly implemented customer segmentation helps you plan special offers and deals. Frequent deals have become a staple of e-commerce and commercial software in the past few years. If you reach a customer with just the right offer, at the right time, there’s a huge chance they’re going to buy. Customer segmentation will help you tailor your special offers perfectly.

* **Marketing**

The marketing strategy can be directly improved with segmentation because you can plan personalized marketing campaigns for different customer segments, using the channels that they use the most.

* **Customer satisfaction**

By studying different customer groups, you learn what they value the most about your company. This information will help you create personalized products and services that perfectly fit your customers’ preferences.In the next section, we’re going to discuss why machine learning for customer segmentation.

**Machine Learning for customer segmentation**

Machine learning methodologies are a great tool for analyzing customer data and finding insights and patterns. Artificially intelligent models are powerful tools for decision-makers. They can precisely identify customer segments, which is much harder to do manually or with conventional analytical methods.There are many machine learning algorithms, each suitable for a specific type of problem. One very common **machine learning algorithm that’s suitable for customer segmentation problems is the k-means clustering algorithm**. There are other clustering algorithms as well such as DBSCAN, Agglomerative Clustering, and BIRCH, etc.

**Why would you implement machine learning for customer segmentation?**

Manual customer segmentation is time-consuming. It takes months, even years to analyze piles of data and find patterns manually.  Also if done heuristically, it may not have the accuracy to be useful as expected.

Customer segmentation used to be done manually and wasn’t too precise. You’d manually create and populating different data tables, and analyze the data like a detective with a looking glass. Now, it’s much better (and relatively easy thanks to rapid progress in ML) to just use machine learning, which can free up your time to focus on more demanding problems that require creativity to solve.

**Ease of retraining :**

Customer Segmentation is not a “develop once and use forever” type of project. Data is ever-changing, trends oscillate, everything keeps changing after your model is deployed. Usually, more labeled data becomes available after development, and it’s a great resource for improving the overall performance of your model.

There are many ways to update customer segmentation models, but here are the two main approaches:

* **Use the old model as the starting point and retrain it.**
* **Keep the existing model and combine its output with a new model.**

**Better scaling**

Machine learning models deployed in production support scalability, thanks to cloud infrastructure. These models are quite flexible for future changes and feedback. For example, consider a company that has 10000 customers today, and they’ve implemented a customer segmentation model. After a year, if the company has 1 million customers, then ideally we don’t need to create a separate project to handle this increased data. Machine Learning models have the inherent capability to handle more data and scale in production.

**Higher accuracy**

The value of an optimal number of clusters for given customer data is easy to find using machine learning methods like the elbow method. Not only the optimal number of clusters but also the performance of the model is far better when we use machine learning.

**Exploring customer dataset and its features**

Let’s analyze a [customer dataset](https://docs.google.com/spreadsheets/d/1KAAjTC6rrcrPvP2HLeUV5OvZnUUqlu-Y6J6a_8IR0So/edit). Our dataset has 24,000 data points and four features. The features are:

* Customer ID – This is the id of a customer for a particular business.
* Products Purchased – This feature represents the number of products purchased by a customer in a year.
* Complaints – This column value indicates the number of complaints made by the customer in the last year
* Money Spent – This column value indicates the amount of money paid by the customer over the last year.

In the upcoming section, we’ll pre-process this dataset.



**Implementing K-means clustering in Python**

K-Means clustering is an efficient machine learning algorithm to solve data clustering problems. It’s an unsupervised algorithm that’s quite suitable for solving customer segmentation problems. Before we move on, let’s quickly explore two key concepts

**Unsupervised Learning**

Unsupervised machine learning is quite different from supervised machine learning. It’s a special kind of machine learning algorithm that discovers patterns in the dataset from unlabelled data. Unsupervised machine learning algorithms can group data points based on similar attributes in the dataset. One of the main types of unsupervised models is clustering models.Note that, supervised learning helps us produce an output from the previous experience.

**Clustering algorithms**

A clustering machine learning algorithm is an unsupervised machine learning algorithm. It’s used for discovering natural groupings or patterns in the dataset. It’s worth noting that clustering algorithms just interpret the input data and find natural clusters in it.

Some of the most popular clustering algorithms are:

* K-Means Clustering
* Agglomerative Hierarchical Clustering
* Expectation-Maximization (EM) Clustering
* Density-Based Spatial Clustering
* Mean-Shift Clustering

In the following section, we’re going to analyze the customer segmentation problem using the k-means clustering algorithm and machine learning. However, before that, let’s quickly discuss why we’re using the k-means clustering algorithm.

**First, we need to implement the required Python libraries as shown in the table below.**

*# Import required libraries*

**import** pandas **as** pd

**import** numpy **as** np

**from** sklearn.cluster **import** KMeans

**import** plotly.express **as** px

**import** plotly.graph\_objects **as** go

**import** matplotlib.pyplot **as** plt

We’ve imported the pandas, NumPy sklearn, plotly and matplotlib libraries. Pandas and NumPy are used for data wrangling and manipulation, sklearn is used for modelling, and plotly along with matplotlib will be used to plot graphs and images.

After importing the library, our next step is to load the data in the pandas data frame. For this, we’re going to use the read\_csv method of pandas.

*#Load customers data*

customersdata = pd.read\_csv("customers-data.csv")



After loading the data, we need to define the K- means model. This is done with the help of the KMeans class that we imported from sklearn, as shown in the code below.

the k-means clustering model with these optimal values of k. This will produce our final model. *# Define K-means model*

kmeans\_model = KMeans(init='k-means++', max\_iter=400, random\_state=42)

After defining the model, we want to train is using a training dataset. This is implemented with the use of the fit method, as shown in the code below.

*# Train the model*

kmeans\_model.fit(customersdata[['products\_purchased','complains',

'money\_spent']])

Though we have trained a K-means model up to these points, we haven’t found the optimal number of clusters required in this case of customer segmentation. Finding the optimal number of clusters, for the given dataset is important for producing a high-performant k-means clustering model. In the upcoming section, we’re going to find the optimal number of clusters of the given dataset and then re-train

**Finding the optimal number of clusters**

Finding the optimal number of clusters is one of the key tasks when implementing a k-means clustering algorithm. It’s worth noting that a k-means clustering model might converge for any value of K, but at the same time, not all values of K will produce the best model.

For some datasets, data visualization can help understand the optimal number of clusters, but this doesn’t apply to all datasets. We have a few methods, such as the elbow method, gap statistic method, and average silhouette method, to assess the optimal number of clusters for a given dataset. We’ll discuss them one by one.

* **The elbow method**finds the value of the optimal number of clusters using the total within-cluster sum of square values. This represents how spread-apart the generated clusters are from one another. In this case, the K-means algorithm is evaluated for several values of k, and the within-cluster sum of square values is calculated for each value of k. After this, we plot the K versus the sum of square values. After analyzing this graph, the number of clusters is selected, so that adding a new cluster doesn’t change the values of the sum of square values significantly.
* **Average silhouette method** is a measure of how well each data point fits its corresponding cluster. This method evaluates the quality of clustering. As a general rule, a high average silhouette width denotes better clustering output.
* **Gap statistic method** is a measure of the value of gap statistics. Gap statistics is the difference between the total intracluster changes for various values of k compared to their expected values. This calculation is done using the null reference distribution of data points. The optimal number of clusters is the value that maximizes the value of gap statistics.

*# Create the K means model for different values of K*

**def** **try\_different\_clusters**(K, data):

cluster\_values = list(range(1, K+1))

inertias=[]

**for** c **in** cluster\_values:

model = KMeans(n\_clusters = c,init='k-means++',max\_iter=400,random\_state=42)

model.fit(data)

inertias.append(model.inertia\_)

**return** inertias

The method try\_different\_clusters is called using the below code, where we pass the values of K from 1 to 12 and calculate the inertia for each value of k.

*# Find output for k values between 1 to 12*

outputs = try\_different\_clusters(12, customersdata[['products\_purchased','complains','money\_spent']])

distances = pd.DataFrame({"clusters": list(range(1, 13)),"sum of squared distances": outputs})

Using the below code, we plot the value of K (on the x-axis) against corresponding values of inertia on the Y-axis.

*# Finding optimal number of clusters k*

figure = go.Figure()

figure.add\_trace(go.Scatter(x=distances["clusters"], y=distances["sum of squared distances"]))

figure.update\_layout(xaxis = dict(tick0 = 1,dtick = 1,tickmode = 'linear'),

xaxis\_title="Number of clusters",

yaxis\_title="Sum of squared distances",

title\_text="Finding optimal number of clusters using elbow method")

figure.show()

We can generate the below plot using the above code. The elbow of the code is at K=5. We have chosen 5 as if we increase the number of clusters to more than 5, there is very small change in the inertia or sum of the squared distance.

**Optimal value of K = 5**

The stage at which the number of clusters is optimal is called the elbow of the clustering model. For example, in the below image, the elbow is at five clusters (K =5). Adding more than 5 clusters will cause the creation of an inefficient or less performant clustering model.

As discussed before, we need to train the k-means clustering model again with the optimal number of clusters found.

*# Re-Train K means model with k=5*

kmeans\_model\_new = KMeans(n\_clusters = 5,init='k-means++',max\_iter=400,random\_state=42)

kmeans\_model\_new.fit\_predict(customersdata[['products\_purchased','complains','money\_spent']])

Note that we’re using the fit\_predict method to train the model.

In the next section, we’re going to discuss how to visualize customer segmentation clusters in three dimensions.

**Visualizing customer segments**

In this section, we’ll be implementing some code using plotly express. This way we’ll visualize the clusters in three dimensions, formed by our k-means algorithm. Plotly express is a library based on plotly that works on several types of datasets and generates highly-styled plots.

First, let’s add a new column named ‘clusters’ to the existing customer data dataset. This column will be able to tell which customer belongs to what cluster.

*# Create data arrays*

cluster\_centers = kmeans\_model\_new.cluster\_centers\_

data = np.expm1(cluster\_centers)

points = np.append(data, cluster\_centers, axis=1)

points

*# Add "clusters" to customers data*

points = np.append(points, [[0], [1], [2], [3], [4]], axis=1)

customersdata["clusters"] = kmeans\_model\_new.labels\_

After adding the new column, named clusters, the customer data dataset will look as below.

Finally, we’re going to use the below code to visualize the five clusters created. This is done using plotly with the express library.Plotly is a Python library used for graphing, statistics, plotting, and analytics. This can be used along with Python, R, Julia, and other programming languages. Plotly is a free and open-source library.

As the [2D scatter plot](https://plotly.com/python/line-and-scatter/), px.scatter plots individual data in a two-dimensional space, and the 3D method px.scatter\_3d plots individual data in a three-dimensional space.

*# visualize clusters*

figure = px.scatter\_3d(customersdata,

color='clusters',

x="products\_purchased",

y="complains",

z="money\_spent",

category\_orders = {"clusters": ["0", "1", "2", "3", "4"]}

)

figure.update\_layout()

figure.show()

**Conclusion**

It’s not wise to serve all customers with the same product model, email, text message campaign, or ad. Customers have different needs. A one-size-for-all approach to business will generally result in less engagement, lower-click through rates, and ultimately fewer sales. Customer segmentation is the cure for this problem.

Finding an optimal number of unique customer groups will help you understand how your customers differ, and help you give them exactly what they want. Customer segmentation improves customer experience and boosts company revenue. That’s why segmentation is a must if you want to surpass your competitors and get more customers. Doing it with machine learning is definitely the right way to go.