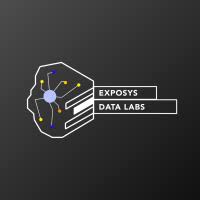
**INTERNSHIP FROM:**

****

**EXPOSYS DATA LABS**

**PROJECT REPORT OF: CUSTOMER SEGMENTATION**

**SUBMITTED BY: ARJYA BASU**

**ABSTRACT**

**Customer Segmentation** is a popular application of unsupervised learning. Using clustering, identifying segments of customers to target the potential user base. They divide customers into groups according to common characteristics like gender, age, interests, and spending habits so they can market to each group effectively.

In business-to-business marketing, a company might segment customers according to a wide range of factors, including:

* Industry
* Number of employees
* Products previously purchased from the company
* Location

In business-to-consumer marketing, companies often segment customers according to demographics that include:

* Age
* Gender
* Marital status
* Location (urban, suburban, rural)
* Life stage (single, married, divorced, empty-nester, retired, etc.)
* **TABLE OF CONTENTS**
* **1. INTRODUCTION**
* **2. EXISTING METHOD**
* **3. PROPOSED METHOD WITH ARCHITECTURE**
* **4. IMPLEMENTATION**
* **5. CONCLUSION**

**INTRODUCTION**

Segmentation allows marketers to better tailor their marketing efforts to various audience subsets. Those efforts can relate to both communications and product development. Specifically, segmentation helps a company:

* Create and communicate targeted marketing messages that will resonate with specific groups of customers, but not with others (who will receive messages tailored to their needs and interests, instead).
* Select the best communication channel for the segment, which might be email, social media posts, radio advertising, or another approach, depending on the segment.
* Identify ways to improve products or new product or service opportunities.
* Establish better customer relationships.
* Test pricing options.
* Focus on the most profitable customers.
* Improve customer service.
* Upsell and cross-sell other products and services.

Customer segmentation requires a company to gather specific information – data – about customers and analyze it to identify patterns that can be used to create segments.

Some of that can be gathered from purchasing information – job title, geography, products purchased, for example. Some of it might be gleaned from how the customer entered your system. An online marketer working from an opt-in email list might segment marketing messages according to the opt-in offer that attracted the customer, for example. Other information, however, including consumer demographics such as age and marital status, will need to be acquired in other ways.

Typical information-gathering methods include:

* Face-to-face or telephone interviews
* Surveys
* General research using published information about market categories
* Focus groups

Using Customer Segments

Common characteristics in customer segments can guide how a company markets to individual segments and what products or services it promotes to them. A small business selling hand-made guitars, for example, might decide to promote lower-priced products to younger guitarists and higher-priced premium guitars to older musicians based on segment knowledge that tells them that younger musicians have less disposable income than their older counterparts. Similarly, a meals-by-mail service might emphasize convenience to millennial customers and “tastes-like-mother-used-to-make” benefits to baby boomers.

Customer segmentation can be practiced by all businesses regardless of size or industry and whether they sell online or in person. It begins with gathering and analyzing data and ends with acting on the information gathered in a way that is appropriate and effective.

**EXISTING METHOD**

### k-means clustering algorithm

|  |
| --- |
| k-means is  one of  the simplest unsupervised  learning  algorithms  that  solve  the well  known clustering problem. The procedure follows a simple and  easy  way  to classify a given data set  through a certain number of  clusters (assume k clusters) fixed apriori. The  main  idea  is to define k centers, one for each cluster. These centers  should  be placed in a cunning  way  because of  different  location  causes different  result. So, the better  choice  is  to place them  as  much as possible  far away from each other. The  next  step is to take each point belonging  to a  given data set and associate it to the nearest center. When no point  is  pending,  the first step is completed and an early group age  is done. At this point we need to re-calculate k new centroids as barycenter of  the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done  between  the same data set points  and  the nearest new center. A loop has been generated. As a result of  this loop we  may  notice that the k centers change their location step by step until no more changes  are done or  in  other words centers do not move any more. Finally, this  algorithm  aims at  minimizing  an objective function know as squared error function given by:  [https://sites.google.com/site/dataclusteringalgorithms/_/rsrc/1273047853039/k-means-clustering-algorithm/kmeans.JPG](https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm/kmeans.JPG?attredirects=0)  where,                            *‘||xi- vj||’* is the Euclidean distance between *xi* and *vj.*  *‘ci’* is the number of data points in *ith* cluster.  *‘c’* is the number of cluster centers. |

**Algorithmic steps for k-means clustering**

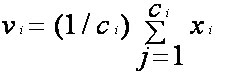
Let  X = {x1,x2,x3,……..,xn} be the set of data points and V = {v1,v2,…….,vc} be the set of centers.

1) Randomly select *‘c’* cluster centers.

2) Calculate the distance between each data point and cluster centers.

3) Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers..

4) Recalculate the new cluster center using:



where,*‘ci’* represents the number of data points in *ith* cluster.

5) Recalculate the distance between each data point and new obtained cluster centers.

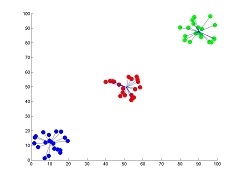
6) If no data point was reassigned then stop, otherwise repeat from step 3).

**Advantages**

1) Fast, robust and easier to understand.

2) Relatively efficient: O(tknd), where n is # objects, k is # clusters, d is # dimension of each object, and t  is # iterations. Normally, k, t, d << n.

3) Gives best result when data set are distinct or well separated from each other.

[](https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm/k-means.jpg?attredirects=0)  
**Fig I**: Showing the result of k-means for *'N'* = 60 and *'c'* = 3

**Note:** For more detailed figure for k-means algorithm please refer to [k-means figure](https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm/k-means-figure) sub page.

**Disadvantages**

1) The learning algorithm requires apriori specification of the number of  cluster centers.

2) The use of  Exclusive Assignment - If  there are two highly overlapping data then k-means will not be able to resolve       that there are two clusters.

3) The learning algorithm is not invariant to non-linear transformationsi.e.with different representation of data we get

    different results (data represented in form of cartesian co-ordinates and polar co-ordinates will give different results).

4) Euclidean distance measures can unequally weight underlying factors.

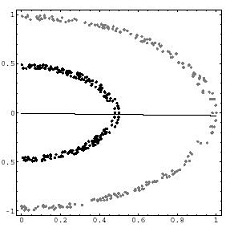
5) The learning algorithm provides the local optima of the squared error function.

6) Randomly choosing of the cluster center cannot lead us to the fruitful result. Pl. refer [Fig](https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm/k-means_initial_cluster_selection).

7) Applicable only when mean is defined i.e. fails for categorical data.

8) Unable to handle noisy data and outliers*.*

9) Algorithm fails for non-linear data set.

[](https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm/k-means_fail.jpg?attredirects=0)

**Fig II**: Showing the non-linear data set where k-means algorithm fails

**PROPOSED METHOD WITH ARCHITECTURE**

### Hierarchical clustering algorithm

|  |
| --- |
| Hierarchical clustering algorithm is of two types:  i) Agglomerative Hierarchical clustering algorithm or AGNES (agglomerative nesting) and  ii) Divisive Hierarchical clustering algorithm or DIANA (divisive analysis).  Both this algorithm are exactly reverse of each other. So we will be covering Agglomerative Hierarchical clustering algorithm in detail.  Agglomerative Hierarchical clustering -This algorithm  works by  grouping  the data one by one on the basis of the  nearest distance measure of all the pairwise distance between the data point. Again distance between the data point is recalculated but which distance to consider when the groups has been formed? For this there are many available methods. Some of them are:  1) single-nearest distance or single linkage.  2) complete-farthest distance or complete linkage.  3) average-average distance or average linkage.  4) centroid distance.  5) ward's method - sum of squared euclidean distance is minimized.  This way we go on grouping the data until one cluster is formed. Now on the basis of dendogram graph we can calculate how many number of clusters should be actually present.  **Algorithmic steps for Agglomerative Hierarchical clustering**  Let  X = {x1, x2, x3, ..., xn} be the set of data points.  1) Begin with the disjoint clustering having level L(0) = 0 and sequence number m = 0.  2) Find the least distance pair of clusters in the current clustering, say pair (r), (s), according to d[(r),(s)] = min d[(i),(j)]   where the minimum is over all pairs of clusters in the current clustering.  3)Increment the sequence number: m = m +1.Merge clusters (r) and (s) into a single cluster to form the next clustering   m. Set the level of this clustering to L(m) = d[(r),(s)].  4) Update the distance matrix, D, by deleting the rows and columns corresponding to clusters (r) and (s) and adding a row and column corresponding to the newly formed cluster. The distance between the new cluster, denoted (r,s) and old cluster(k) is defined in this way: d[(k), (r,s)] = min (d[(k),(r)], d[(k),(s)]).  5)If all the data points are in one cluster then stop, else repeat from step 2).  Divisive Hierarchical clustering - It is just the reverse of Agglomerative Hierarchical approach.  **Advantages**  1) No apriori information about the number of clusters required.  2) Easy to implement and gives best result in some cases.  **Disadvantages**  1) Algorithm can never undo what was done previously.  2) Time complexity of at least O(*n2log n*) is required, where *‘n’* is the number of data points.  3) Based on the type of distance matrix chosen for merging different algorithms can suffer with one or more of the following:      i) Sensitivity to noise and outliers      ii) Breaking large clusters      iii) Difficulty handling different sized clusters and convex shapes  4) No objective function is directly minimized  5) Sometimes it is difficult to identify the correct number of clusters by the dendogram.  [https://sites.google.com/site/dataclusteringalgorithms/_/rsrc/1273874378963/hierarchical-clustering-algorithm/dendogram.png](https://sites.google.com/site/dataclusteringalgorithms/hierarchical-clustering-algorithm/dendogram.png?attredirects=0)  **Fig I**: Showing dendogram formed from the data set of size *'N'* = 60 |

**IMPLEMENTATION**

**Libraries Used** in this projects are:- 1. Matplotlib 3.1.3

2. pandas 1.0.1

3. sklearn 0.22

**CODE:-**

# K-Means Clustering

# Importing the libraries

import matplotlib.pyplot as plt

import pandas as pd

import seaborn as sns

# Importing the dataset

dataset = pd.read\_csv('Mall\_Customers.csv')

X = dataset.iloc[:, [1,2,3, 4]].values

# Exploratory Data Analysis

dataset.isnull().sum()

sns.pairplot(dataset.drop(["CustomerID","Annual Income (k$)","Spending Score (1-100)"],axis=1))

from sklearn.preprocessing import LabelEncoder

labelencoder\_X = LabelEncoder()

X[:, 0] = labelencoder\_X.fit\_transform(X[:, 0])

# Using the elbow method to find the optimal number of clusters

from sklearn.cluster import KMeans

wcss = []

for i in range(1, 11):

kmeans = KMeans(n\_clusters = i, init = 'k-means++', random\_state = 0)

kmeans.fit(X)

wcss.append(kmeans.inertia\_)

plt.plot(range(1, 11), wcss)

plt.title('The Elbow Method')

plt.xlabel('Number of clusters')

plt.ylabel('WCSS')

plt.show()

# Fitting K-Means to the dataset

kmeans = KMeans(n\_clusters = 5, init = 'k-means++', random\_state =0)

y\_kmeans = kmeans.fit\_predict(X)

# Visualising the clusters

plt.scatter(X[y\_kmeans == 0, 2], X[y\_kmeans == 0, 3], s = 50, c = 'red', label = 'Sensible Customers')

plt.scatter(X[y\_kmeans == 1, 2], X[y\_kmeans == 1, 3], s = 50, c = 'blue', label = 'Good Customers')

plt.scatter(X[y\_kmeans == 2, 2], X[y\_kmeans == 2, 3], s = 50, c = 'green', label = 'Target Customers')

plt.scatter(X[y\_kmeans == 3, 2], X[y\_kmeans == 3, 3], s = 50, c = 'cyan', label = 'Cautious Customers')

plt.scatter(X[y\_kmeans == 4, 2], X[y\_kmeans == 4, 3], s = 50, c = 'magenta', label = 'Careless Customers')

plt.scatter(kmeans.cluster\_centers\_[:, 2], kmeans.cluster\_centers\_[:, 3], s = 100, c = 'yellow', label = 'Centroids')

plt.title('Clusters of customers')

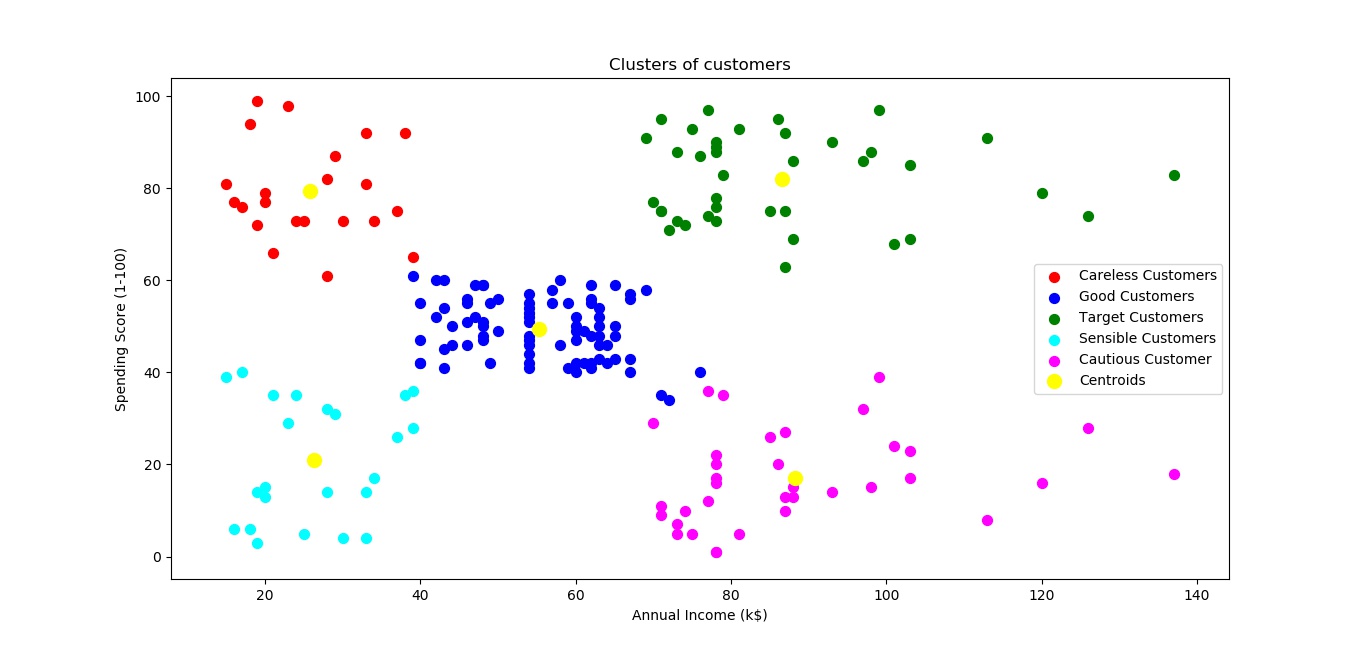
plt.xlabel('Annual Income (k$)')

plt.ylabel('Spending Score (1-100)')

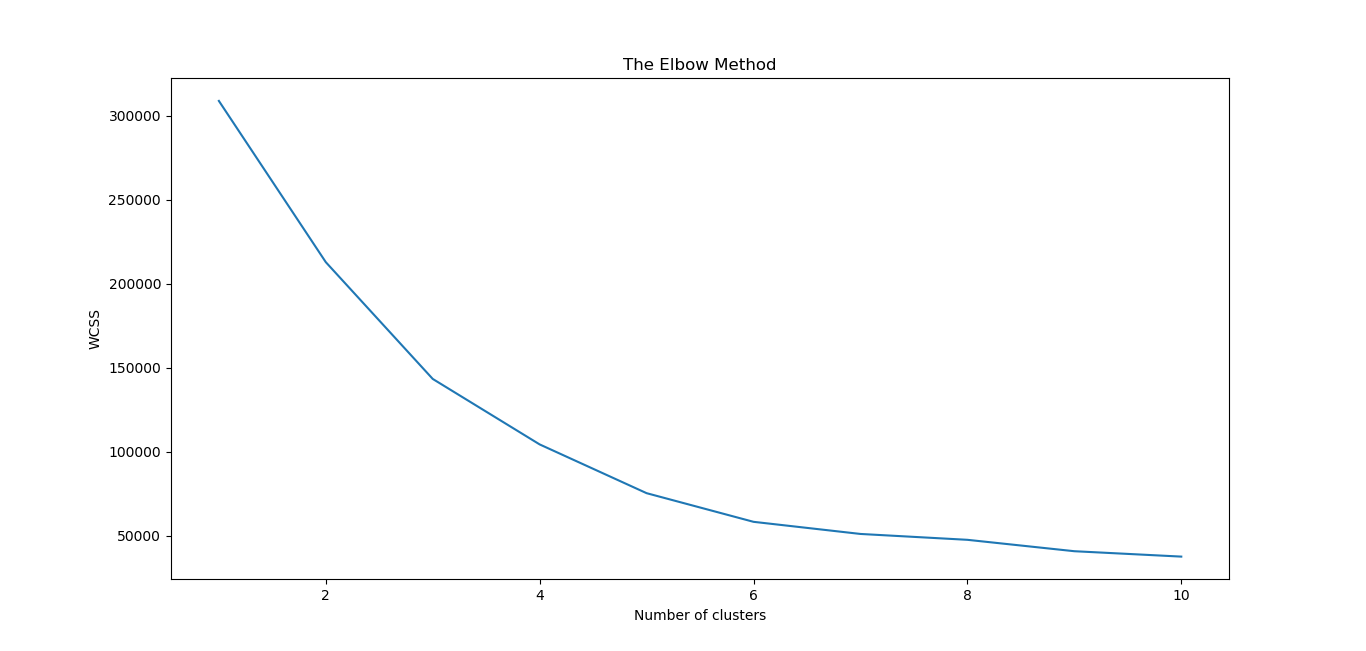
plt.legend()

plt.show()

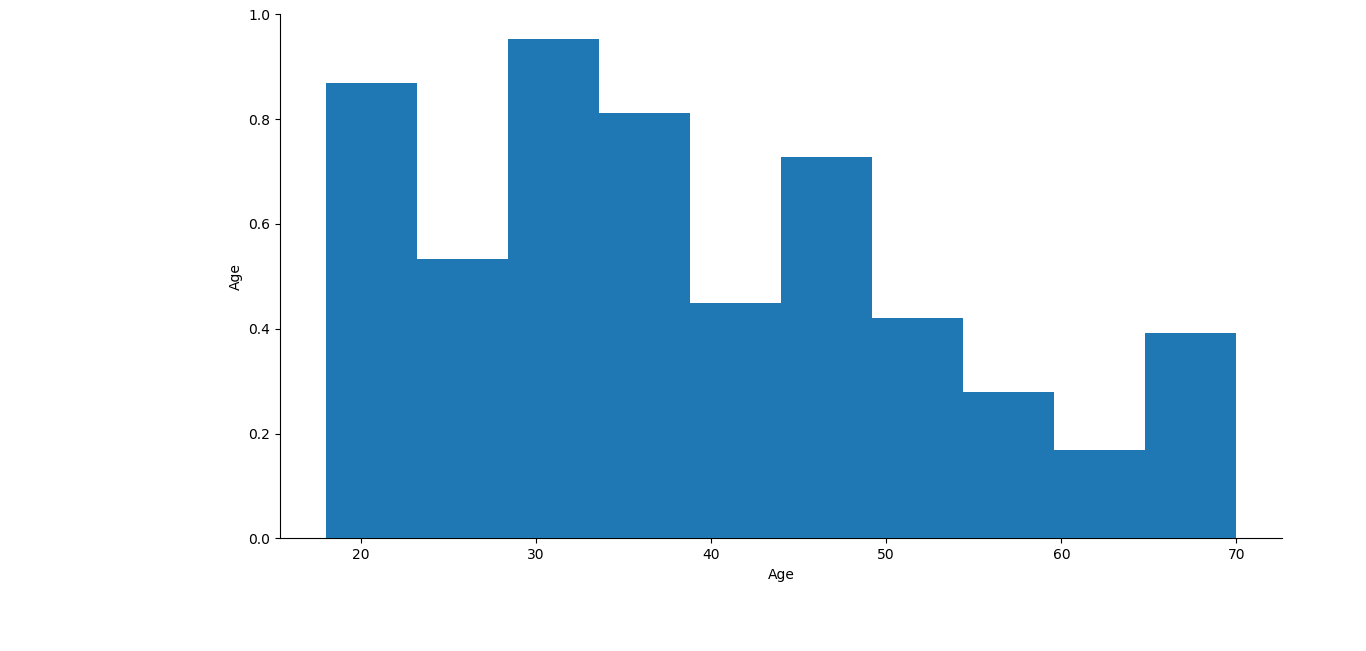
**OUTPUT GRAPHS:-**



**Customer Segmentation Graph**



**Elbow Plot Graph**



**Age-Gender Distribution Graph**

**CONCLUSION**

In this project, we clustered the customers into 5 groups with each group having it’s own specialty. The names of the groups are Sensible customers, Good customers, Target customers, Cautious customers and Careless customers respectively. Now the marketing team can easily show the right advertisements to their customers and hence increasing their revenues. This will also become a great recommendation system of products for the customers thus becoming a loyal buyer of products from the company.