Customer Segmentation Using K-Means Clustering Algorithm

Abstract

We live in a world where a large and vast amount of data is collected daily. Analysing such data is an important need. In the modern era of innovation, where there is a large competition to be better than everyone, the business strategy needs to be according to the modern conditions. The business done today runs on the basis of innovative ideas as there are large number of potential customers who are confused about what to buy and what not to buy. The companies doing the business are also not able to diagnose the target potential customers. This is where machine learning comes into picture, the various algorithms are applied to identify the hidden patterns in the data for better decision making. The concept of which customer segment to target is done using the customer segmentation process using the clustering technique. In this paper, the clustering algorithm used is K-means algorithm which is the partitioning algorithm, to segment the customers according to the similar characteristics. To determine the optimal clusters, the elbow method is used.

Keywords: Customer segmentation, K-Means Clustering, Elbow Method

Table of Contents

1. Introduction

Over the years, the competition amongst businesses is increased and the large historical data that is available has resulted in the widespread use of data mining techniques in extracting the meaningful and strategic information from the database of the organisation. Data mining is the process where methods are applied to extract data patterns in order to present it in the human readable format which can be used for the purpose of decision support. According to,[4] Clustering techniques consider data tuples as objects. They partition the data objects into groups or clusters, so that objects within a cluster are similar to one another and dissimilar to objects in other clusters.

Customer Segmentation is the process of division of the customer base into several groups called customer segments such that each customer segment consists of customers who have similar characteristics. The segmentation is based on the similarity in different ways that are relevant to marketing such as gender, age, interests, and miscellaneous spending habits.

The customer segmentation has the importance as it includes, the ability to modify the programs of market so that it is suitable to each of the customer segment, support in business decision; identification of products associated with each customer segment and to manage the demand and supply of that product; identifying and targeting the potential customer base, and predicting customer defection, providing directions in finding the solutions.

2. Literature Review

2.1. Customer Segmentation

Over the years, as there is very strong competition in the business world, the organizations have to enhance their profits and business by satisfying the demands of their customers and attract new customers according to their needs. The identification of customers and satisfying the demands of each customer is a very complex and tedious task. This is because customers may be different according to their demands, tastes, preferences and so on. Instead of “one-size-fits-all” approach, customer segmentation clusters the customers into groups sharing the same properties or behavioural characteristics. According to,[5] customer segmentation is a strategy of dividing the market into homogeneous groups. The data used in customer segmentation technique that divides the customers into groups depends on various factors like, data geographical conditions, economic conditions, demographic conditions as well as behavioural patterns. The customer segmentation technique allows the business to make better use of their marketing budgets, gain a competitive edge over their rival companies, demonstrating the better knowledge of the needs of the customer. It also helps an organization in increasing their marketing efficiency, determining

new market opportunities, making better brand strategy, identifying customers retention.

2.2. Clustering and K-Means Algorithm

Clustering algorithms generates clusters such that within the clusters are similar based on some characteristics. Similarity is defined in terms of how close the objects are in space. K-means algorithm in one of the most popular centroid based algorithm. Suppose data set, D, contains n objects in space. Partitioning methods distribute the objects in D into k clusters, C1,...,Ck , that is, Ci $\subset$ D and Ci $\cap$Cj = $\varnothing$ for ($1 \leq i, j \leq k$). A centroid-based partitioning technique uses the centroid of a cluster, Ci , to represent that cluster. Conceptually, the centroid of a cluster is its center point. The difference between an object p $\in$ Ci and ci , the representative of the cluster, is measured by dist(p,ci), where dist(x,y) is the Euclidean distance between two points x and y.

Algorithm: The k-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster. Input: k: the number of clusters, D: a data set containing n objects. Output: A set of k clusters.

Method: (1) arbitrarily choose k objects from D as the initial cluster centers;

(2) repeat

(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;

(4) update the cluster means, that is, calculate the mean value of the objects for each cluster;

(5) until no change.

3. Methodology

The data set used to implement clustering and Kmeans algorithm was collected from a store of shopping mall. The data set contains 5 attributes and has 200 tuples, representing the data of 200 customers. The attributes in the data set has CustomerId, gender, age, annual income(k$), spending score on the scale of (1-100).

3.1. Visualize the Gender of Customers

Fig 1: Barplot displaying gender comparison in given dataset.

3.2. Visualize the Age of Customers

```
# count plot for 'Age'
plt.figure(figsize=(20,8))
sns.countplot(record['Age'])
plt.title('Age count plot', fontsize = 15)
plt.xlabel('Age', fontsize = 12)
plt.ylabel('Count', fontsize = 12)
plt.show()
```

Fig 2: Code for Age Count

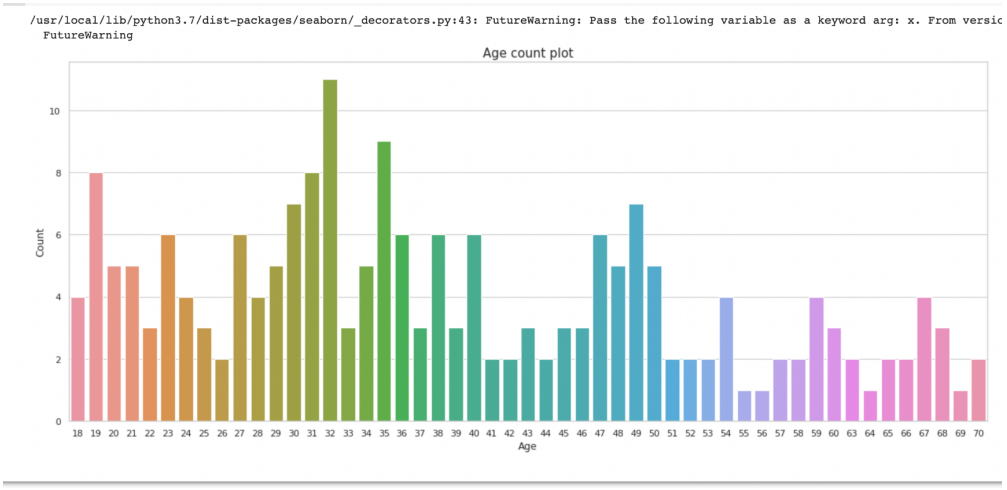


Fig 3: Histogram showing Age count for the Customer dataset.

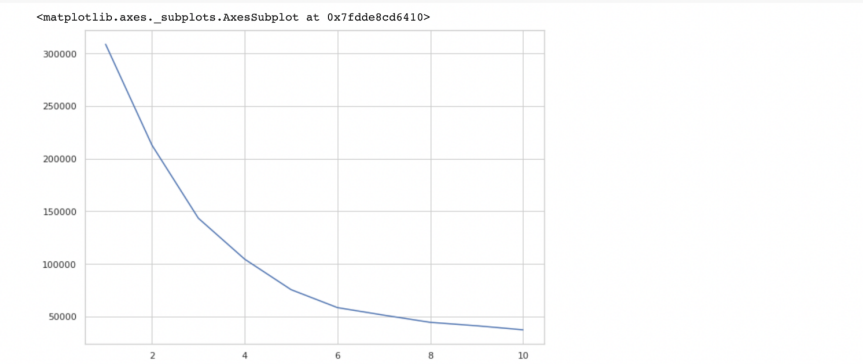
3.3. Elbow Method

The elbow method is based on the observation that increasing the number of clusters can help to reduce the sum of within-cluster variance of each cluster. This is because having more clusters allows one to capture finer groups of data objects that are more similar to each other. To define the optimal clusters, Firstly, we use the clustering algorithm for various values of k. This is done by ranging k from 1 to 10 clusters. Then we calculate the total intra-cluster sum of squares. Then, we proceed to plot the intra-cluster sum of squares based on the number of clusters. The plot denotes the approximate number of clusters required in our model. The optimum clusters can be found from the graph where there is a bend in the graph.

```
[ ] from sklearn.cluster import KMeans

[ ] cluster=list()
    for i in range(1,11):
        kmns=KMeans(n_clusters=i)
        kmns.fit(record)
        cluster.append(kmns.inertia_)
    plt.figure(figsize=(10,7))
    sns.lineplot(x=list(range(1,11)),y=cluster)
```

Fig 4: Code for finding the value of n.



It's notice 2 potential elbow points or "bends" i.e. one at approximately 3 and another at around 5. Thus we run K-means at both those points to form the requires clusters which we'll visualize eventually.

Fig 5: Graph implementing elbow method to show value of n.

Here in the above graph we can see that two values are qualified to be the number of clusters that can be made i.e 3 and 5. We will check for both the values of n to see which gives the best value of `silhouette_score`.

3.4. Visualize the clusters

3.4.1. For n = 3

```
[ ] n=3
    kmeans3=KMeans(n_clusters=n,n_init=10,max_iter=500)
    kmeans3.fit(record)

    KMeans(max_iter=500, n_clusters=3)

[ ] record['clusters']=kmeans3.labels_
    kmeans3.cluster_centers_

    array([[ 0.52631579, 40.39473684, 87.        , 18.63157895],
           [ 0.40650407, 40.32520325, 44.15447154, 49.82926829],
           [ 0.46153846, 32.69230769, 86.53846154, 82.12820513]])

    record.head()
```

	Gender	Age	AnnualIncome	SpendingScore	clusters
0	1	19	15	39	1
1	1	21	15	81	1
2	0	20	16	6	1
3	0	23	16	77	1
4	0	31	17	40	1

The silhouette score is calculated using the mean intra-cluster distance and the mean nearest-cluster distance for each sample.The values of this score range from -1 to 1. Values almost equal to 0 indicate overlapping clusters. Values closer to 1 indicate the best possible clustering while negative values generally indicate that a sample has been assigned to the wrong cluster.

Fig 6: Code for n=3 clusters

```
plt.figure(figsize=(12, 8))
sns.scatterplot(record['AnnualIncome'], record['SpendingScore'], hue=record['clusters'], palette='Set1',style=record['Gender'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the o
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fdde7f6e090>
```

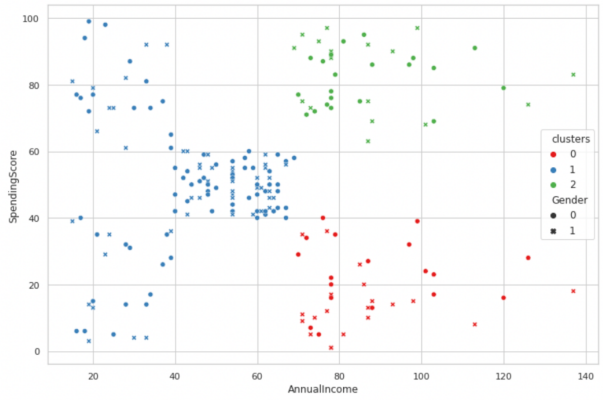


Fig 7: Graph showing 3 clusters.

3.4.2. For n = 5

```
n=5
kmeans5=KMeans(n_clusters=n,n_init=10,max_iter=500)
kmeans5.fit(record)
```

```
KMeans(max_iter=500, n_clusters=5)
```

```
record['clusters']=kmeans5.labels_
kmeans5.cluster_centers_
```

```
array([[4.17721519e-01, 4.30886076e+01, 5.52911392e+01, 4.95696203e+01,
        9.74683544e-01],
       [4.61538462e-01, 3.26923077e+01, 8.65384615e+01, 8.21282051e+01,
        2.00000000e+00],
       [3.91304348e-01, 2.55217391e+01, 2.63043478e+01, 7.85652174e+01,
        1.00000000e+00],
       [5.27777778e-01, 4.06666667e+01, 8.77500000e+01, 1.75833333e+01,
        4.44089210e-16],
       [3.91304348e-01, 4.52173913e+01, 2.63043478e+01, 2.09130435e+01,
        1.00000000e+00]])
```

```
record.head()
```

	Gender	Age	AnnualIncome	SpendingScore	clusters
0	1	19	15	39	4
1	1	21	15	81	2
2	0	20	16	6	4
3	0	23	16	77	2
4	0	31	17	40	4

Fig 8: Code for n=5 clusters

```
plt.figure(figsize=(12, 8))
sns.scatterplot(record['AnnualIncome'], record['SpendingScore'], hue=record['clusters'], palette='Set1',style=record['Gender'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the o
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fdde7e78c10>
```

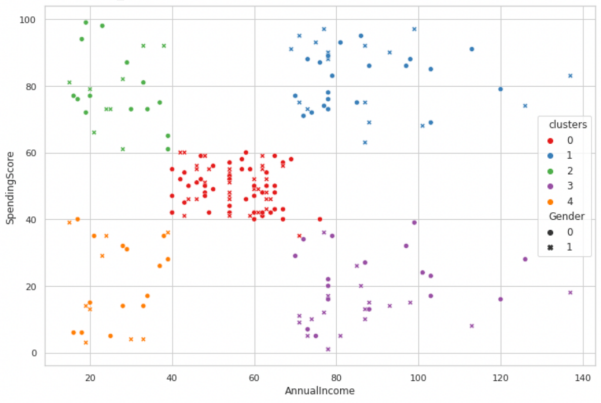


Fig 9: Graph showing 5 clusters

4. Conclusion

With high value of `silhouette_score` for 5 clusters, we conclude that 5 is the optimal value of clusters to segment the given customers.

From the above visualization it can be observed that Cluster 2 denotes the customer who has high annual income as well as high yearly spend. Cluster 4 represents the cluster having high annual income and low annual spend. Cluster 5 represents customer with low annual income and low annual spend. Cluster 3 denotes the low annual income but high yearly spend. Cluster 1 denotes the customer with medium income and medium spending score.

5. References

[1] I. S. Dhillon and D. M. Modha, "Concept decompositions for large sparse text data using clustering," Machine Learning, vol. 42, issue 1, pp. 143-175, 2001.

[2] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient K-means clustering algorithm," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, pp. 881-892, 2002.

[3] MacKay and David, "An Example Inference Task: Clustering," Information Theory, Inference and Learning Algorithms, Cambridge University Press, pp. 284-292, 2003.

[4] Jiawei Han, Micheline Kamber, Jian Pei "Data Mining Concepts and Techniques", Third Edition.

[5] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "The Basis Of Market Segmentation" Euclidean sum-of-squares clustering," Machine Learning, vol. 75, pp. 245-249, 2009.

[6] S. Dasgupta and Y. Freund, "Random Trees for Vector Quantization," IEEE Trans. on Information Theory, vol. 55, pp. 3229-3242, 2009.

[7]Puwanenthiren Premkanth, ―Market Segmentation and Its Impact on Customer Satisfaction with Especial Reference to Commercial Bank of Ceylon PLC.‖ Global Journal of Management and Business Research Publisher: Global Journals Inc. (USA). 2012. Print ISSN: 0975-5853. Volume 12 Issue 1

Step5: Train different classifiers model for mentioned supervised machine learning method based on training data.
Step6: Test the different classifier models for the mentioned supervised machine learning algorithm based on test set.
Step7: Perform a comparative analysis based on performance metrics for each classifier.
Step8: After analyzing based on performance metrics conclude the best performing algorithm.

3. Proposed Method with Architecture

The aim of this project is to apply the parallel Random Forest technique to predict diabetes with less computational time. We experimented with the conventional Random Forest and parallel Random forest algorithms to predict diabetes. In the following, we briefly discuss the phase.

A. Dataset Analysis-

The dataset used in this project is available inside the UCI machine learning repository named as Pima Indian Diabetes Dataset. The dataset contains 9 feature sets and 786 instances.

The target variable is the 9th attribute named Outcome which indicates negative and positive for Diabetes in the binary value.

B. Data Preprocessing:

Prediction from raw data is not a suggested approach. Data preprocessing is the most important technique to produce the accurate result and successful predictions. For Pima Indian diabetes dataset we need to perform preprocessing in two phases.

- Removal of missing or null values and irrelevant feature sets help to reduce the dimensionality of data and helps to work faster.
- After cleaning the dataset, a normalization technique is applied to bring all the attributes of the training and testing dataset under the same scale.

C. Apply Machine Learning:

When data has been prepared we apply Machine Learning Technique. We use conventional and parallel Random Forest techniques to predict diabetes. The algorithm applied on Pima Indians diabetes dataset. The main focus is to apply both approaches to analyze the computational time of these methods and find better accuracy of them.

Figure 1: Overview of the Process

4. Methodology

The proposed Method experimented with conventional and parallel Random Forest techniques available in supervised machine learning algorithms. In the proposed

method, **Parallel Random Forest** achieved almost same accuracy with less computational time compared to conventional Random Forest technique.

The procedure of Proposed Methodology-

Step1: Import required libraries, Import diabetes dataset.

Step2: Preprocessing of data to remove null and missing data from the dataset.

Step3: Perform splitting of the dataset into training and test dataset in 80:20 ratio.

Step4: Select the supervised machine learning algorithm i.e conventional
Random Forest and Parallel Random Forest algorithm.

Step5: Train different classifiers model for mentioned supervised machine learning method based on training data.

Step6: Test the different classifier models for the mentioned supervised machine learning algorithm based on test set.

Step7: Perform a comparative analysis based on performance metrics for each classifier and computational time of both approached.

Step8: After analyzing based on performance metrics and computational time conclude the best performing algorithm.

5. Experimental Results

In this project, the different procedure has been taken. The proposed approach uses a Parallel Random forest method to reduce computational time and is implemented using R. In this project we see that parallel random forest classifier has less computational time with almost same acurracy. Overall we have used conventional and parallel random forest methods for prediction. The Table shows the result of these Machine Learning methods.

Conventional Random Forest	

Dataset(80:20)

Computational Time		16.556	17.856
		Training Dataset	
Accuracy	0.8058	0.8119	**Training Dataset**
Computational Time		0.8228	0.8175
		Testing Dataset	
Accuracy	0.139	0.161	**Testing Dataset**

This result shows that parallel Random Forest performs best at the testing phase with high accuracy and low computational time.

Result of Random Forest on testing phase:

Result of Parallel Random Forest on testing phase:

6. Conclusion

The main goal of this project was to implement Diabetes Prediction at early stage using Parallel Machine Learning Technique and Performance Analysis of that method and it has been achieved successfully. The proposed approach uses Parallel Random Forest method to reduce the computational time of training and testing phases. And 82%(AUC) classification accuracy has been achieved in testing dataset for Parallel Random Forest.

7. References

[1]Mitushi Soni, Dr. Sunita Varma, 2020, Diabetes Prediction using Machine Learning Techniques, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 09, Issue 09 (September 2020).

[2]Rajeswari, M & Prabhu, P.. (2019). A Review of Diabetic Prediction Using Machine Learning Techniques.

[3]Aishwarya Mujumdar, V Vaidehi,Diabetes Prediction using Machine Learning Algorithms, Procedia Computer Science, Volume 165, 2019,Pages 292-299,ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.01.047.