Customer Segmentation with K-means Clustering

Unveiling Patterns for Targeted Marketing Strategies

Prepared by: SADOUN kenza

s

Table of contents

Introduction

In the era of big data, customer segmentation has emergerd as a mainspring for startegic decision-making. Leveraging machine learning algorithms, businesses can sift through vast datasets to uncover nuanced patterns, allowing for precise segmentation. This fusion of data and machine learning not only refines marketing strategies but also unveils untapped opportunities for personalized customer engagement.

1. Case Study

My study case revolves around a comprehensive analysis of customer data aquired from Kaggle, with the objective of a customer segmentation.

1.1 Customer Segmentation

Customer Segmentation is the practice of dividing a diverse customer base into distinct groups or segments based on shared characteristics, behaviors, or attributes. The purpose of it, is a better understand and respond to the unique needs and preferences of diffrents customer groups. By classifiying customers into groups, businesses can tailor thier marketing strategies, product offerings and communication approches, this enhances customer satisfaction, engagement and overall business performance.

1.2 Data collection

Data collection is the systematic process of gathering, recording and organizing information from various sources to generate insights, make informed decisions and conduct analyses. My Dataset is on kaggle.

1.3 Objectives

The objetcives of this study are various :

- Refine Marketing staretegies through the fusion of data and machine learning, by understanding distinct customer segments.
- Enhance strayegoc decision-making, businesses aim to discern nuanced patterns within vast datasets, facilitating precise customer segmentation.
- Uncover untapped opportunities for personalized customer engament, businesses aspire to identify areas where personalized interactions can be maximized, fostering deeper connections with customers and potentially discovering new avenues for business growth.

2. Methodology

The methodology employed in this study blends the strengths of data analytics, machine learning, and practical application. Beginning with thorough data exploration and preprocessing to ensure data quality. The core of the methodology revolves around the application of K-means clustering, utilizing the versatility of the R programming language. The process prioritized transparency and reproducibility, allowing for a comprehensive understanding of the decision-making logic.

2.1 R language

R is an open-source programming language and statistical software designed for data analysis and visualization. Known for its versatility with a rich ecosystem of packages contributed by a vibrant community. Its user-friendly syntax and extensive libraries make R a go-to language for diverse applications in academia, industry, and research.

2.2 K-means algorithm

The K-means algorithm is a clustering technique used in unsupervised machine learning to partition a dataset into K distinct clusters. The "K" represents the predefined number of clusters, and the algorithm works iteratively to assign each data point to the nearest cluster centroid based on a defined distance metric, commonly Euclidean distance. The centroids are then recalculated as the mean of the points within each cluster. This process repeats until convergence, optimizing cluster assignments and centroid positions. K-means is offering a computationally efficient method for grouping data points with similar features.

2.3 Elbow method

The Elbow Method is a technique used in K-means clustering to determine the optimal number of clusters for a dataset. The idea behind the method is to run the K-means algorithm for a range of cluster values and calculate the sum of squared distances from each point to its assigned centroid. The Method involves plotting these sum of squared distances against the number of clusters and identifying the "elbow" point, where the rate of decrease sharply changes. The point at which adding more clusters does not significantly reduce the sum of squared distances is considered the optimal number of clusters for the dataset. This visual heuristic aids in selecting an appropriate value for K.

3. Implementation

Moving to the main task, the implementation of the algorithm. This part explains in details, each line of the code.

3.1 EDA

Exploratory Data Analysis, or EDA, is a crucial initial phase in the data analysis process. It involves examining and summarizing key characteristics of a dataset. It employs various statistical and visual techniques to identify trends, anomalies, and patterns within the data.

3.2 Algorithm

```
library(tidyverse)
library(ggplot2)
library(dplyr)
library(cluster)
library(purrr)
```

- The library Tidyverse is a collection of R packages It provides a consistent and coherent set of functions for working with data, making it easier to handle and analyze datasets in a tidy, organized manner.
- The library dplyr is a library for data manipulation and transformation, dplyr is instrumental in efficiently shaping and preparing data for analysis.
- The cluster library in R is used for cluster analysis, which involves grouping similar data points into clusters. It provides K-means algorithm and more.
- purrr is a functional programming library that enhances and simplifies operations on data structures. It's useful for tasks involving iteration and mapping functions across elements.

```
> cdata <- read.csv ("C:/Users/asus/Desktop/pr/Mall_Customers.csv")
> head(cdata)
  CustomerID Gender Age Annual.Income..k.. Spending.Score..1.100.
1          1   Male  19                 15                     39
2          2   Male  21                 15                     81
3          3 Female  20                 16                      6
4          4 Female  23                 16                     77
5          5 Female  31                 17                     40
6          6 Female  22                 17                     76
```

- The function 'read.csv' CSV file "Mall_Customers.csv" from the specified file path, and the data from the file is assigned to a variable named 'cdata'.
- 'head(cdata)' is used to display the first few rows of the 'cdata' data frame.

```
> dim(cdata)
[1] 200   6
> str(cdata)
'data.frame':   200 obs. of  6 variables:
 $ CustomerID            : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Gender                : chr  "Male" "Male" "Female" "Female" ...
 $ Age                   : int  19 21 20 23 31 22 35 23 64 30 ...
 $ Annual.Income..k..    : int  15 15 16 16 17 17 18 18 19 19 ...
 $ Spending.Score..1.100.: int  39 81 6 77 40 76 6 94 3 72 ...
 $ Cluster               : int  3 2 3 2 3 2 3 2 3 2 ...
> names(cdata)
[1] "CustomerID"             "Gender"                 "Age"                    "Annual.Income..k.."
[5] "Spending.Score..1.100." "Cluster"
>
```

- ‘dim(cdata)’ returns the dimensions (number of rows and columns) of a dataframe.
- ‘str(cdata)’ provides the structure of a dataframe, including column names, data types, and sample values.
- ‘names(cdata)’ Returns the column names of a dataframe.

```
> summary(cdata$Age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  18.00   28.75   36.00   38.85   49.00   70.00
> sd(cdata$Age)
[1] 13.96901
>
> summary(cdata$Annual.Income..k..)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  15.00   41.50   61.50   60.56   78.00  137.00
> sd(cdata$Annual.Income..k..)
[1] 26.26472
>
> summary(cdata$Spending.Score..1.100.)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00   34.75   50.00   50.20   73.00   99.00
> sd(cdata$Spending.Score..1.100.)
[1] 25.82352
>
```

- The function ‘summary(cdata$Age)’ generates summary statistics for numeric variables.
- ‘sd(cdata$Age)’ Computes the standard deviation of numeric variable Age.

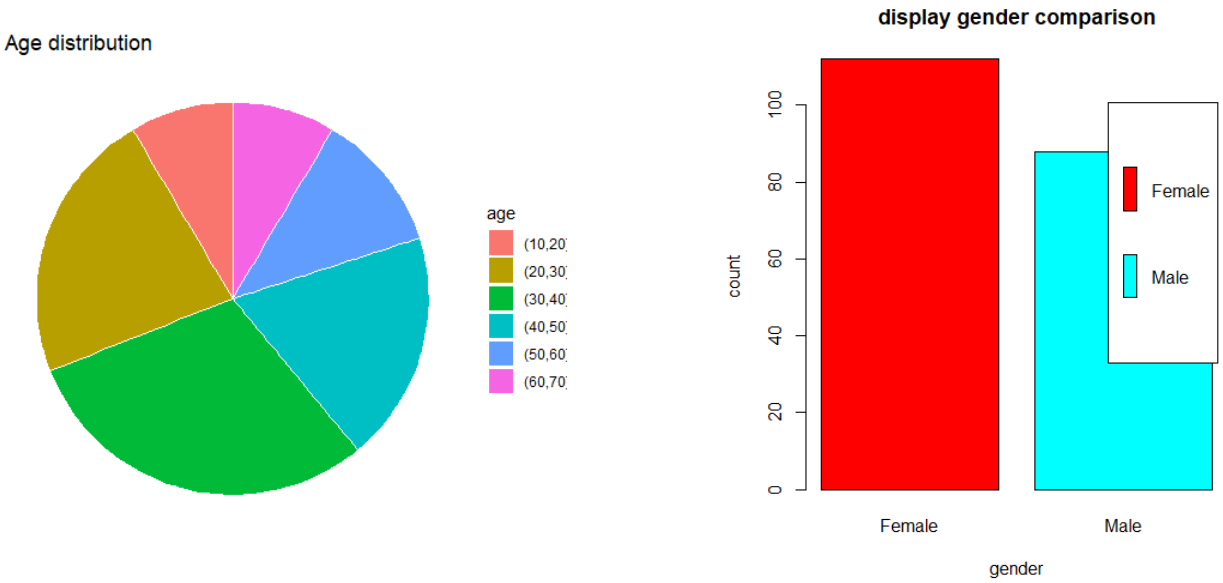
```
a=table(cdata$Gender)
barplot(a, main="display gender comparison", ylab = "count", xlab = "gender",col= rainbow(2), legend= rownames(a))

age_pie <- ggplot(cdata, aes(x= "", fill = cut(Age, breaks = seq(0, 100, by=10)))) +
  geom_bar(width=1, color="white")+
  coord_polar("y", start=0)+
  labs(title = "   Age distribution", fill ="age")+
  theme_void()
  print(age_pie)
```

- table(): Creates a frequency table of categorical variables.
  cdata$Gender: The categorical variable for which the frequency table is generated.

- barplot(): Creates a barplot of categorical data
- ggplot(): Initializes a ggplot object for creating visualizations, geom_bar(): Adds bars to a plot, coord_polar(): Converts Cartesian coordinates to polar coordinates for creating polar plots, abs(): Adds labels to plot axes and titles.
- theme_void(), theme_minimal(): Sets the theme (visual appearance) of the plot.

Result :

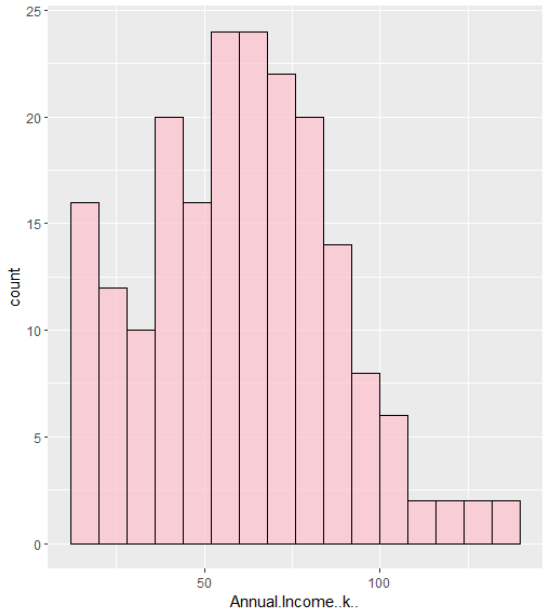
In the pie chart, we observe that the category (30, 40) is the most dominant, and we can note that the maximum age is 70 years old. And the histogram shows that women are the most relevant gender.




```
ggplot (cdata, aes(x = Annual.Income..k..)) +
  geom_histogram(binwidth = 8, fill="pink", color ="black", alpha = 0.7)
  labs(title = " Annual income distibution",
       x ="annual income",
       y ="frequency") +
    theme_minimal()
```

Result :

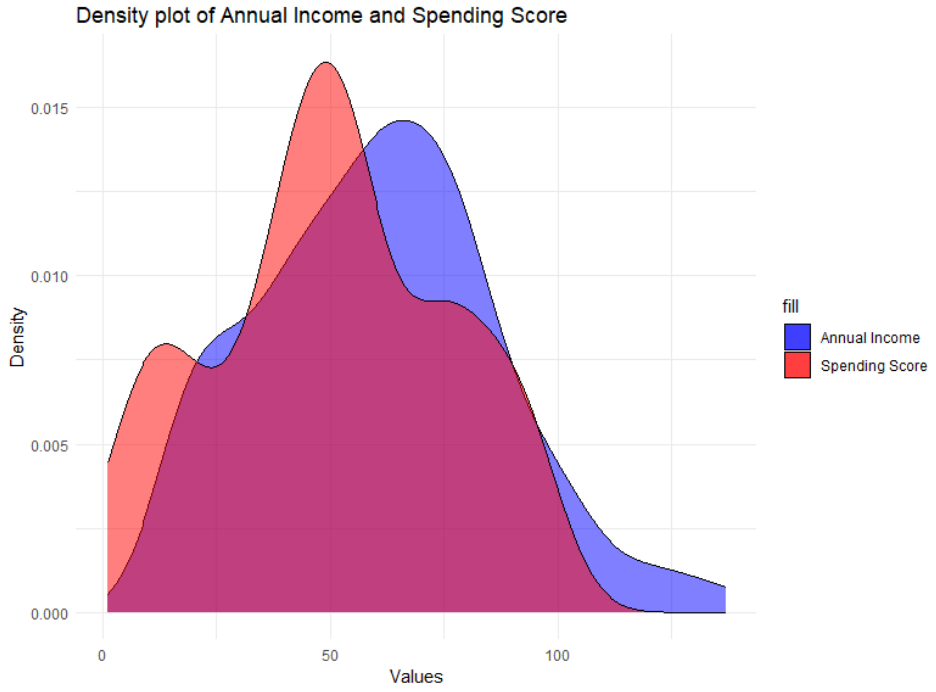
The customers who earn an annual salary of over 100k are fewer than 10, whereas those who earn a salary between 50k and less than 100k are the most numerous.



```r
ggplot(cdata, aes(x = Annual.Income..k.., fill = "Annual Income")) +
  geom_density(alpha = 0.5) +
  geom_density(aes(x = Spending.Score..1.100., fill = "Spending Score"), alpha = 0.5) +
  labs(title = "Density plot of Annual Income and Spending Score",
       x = "Values",
       y = "Density") +
  scale_fill_manual(values = c("Annual Income" = "blue", "Spending Score" = "red")) +
  theme_minimal()
```

Result :

In the Density Plot that we displayed, it shows us a real correlation between the annual spending and earning. We observe that the annual income has a normal distribution.



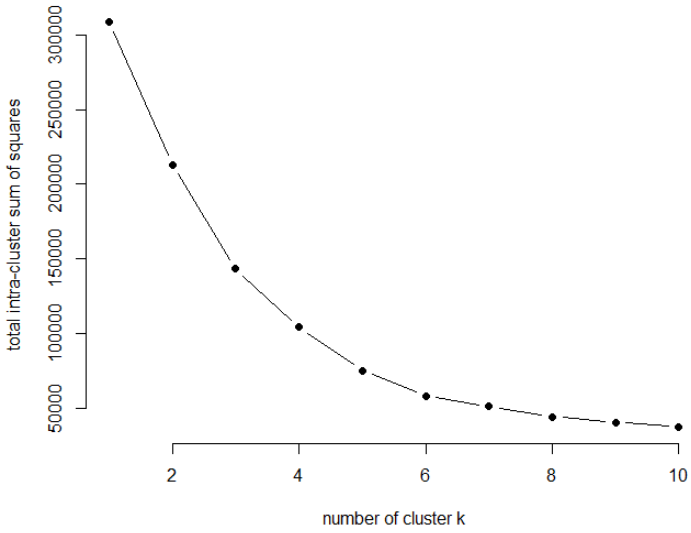
The exploratory data analysis reveals valuable insights into the underlying patterns within the dataset. From the distribution of variables to the prevalence of certain demographics, we've gained a deeper understanding of our data. The visualization techniques employed have shed light on key trends that will enable us to develop targeted strategies and solutions.

Moving to K-means clustering algorithm, the first step is to determine the number of clusters (k) that we wish to produce in the final output, also known as centroids. The algorithm starts by selecting k objects from dataset randomly that will serve as the initial centers for our clusters. This centroid is defined by the Euclidean Distance present between the object and the cluster mean.When the assignment is complete, the algorithm proceeds to calculate new mean value of each cluster present in the data.

To determine the optimal number of cluster, there are many methods such as: **the elbow method.**

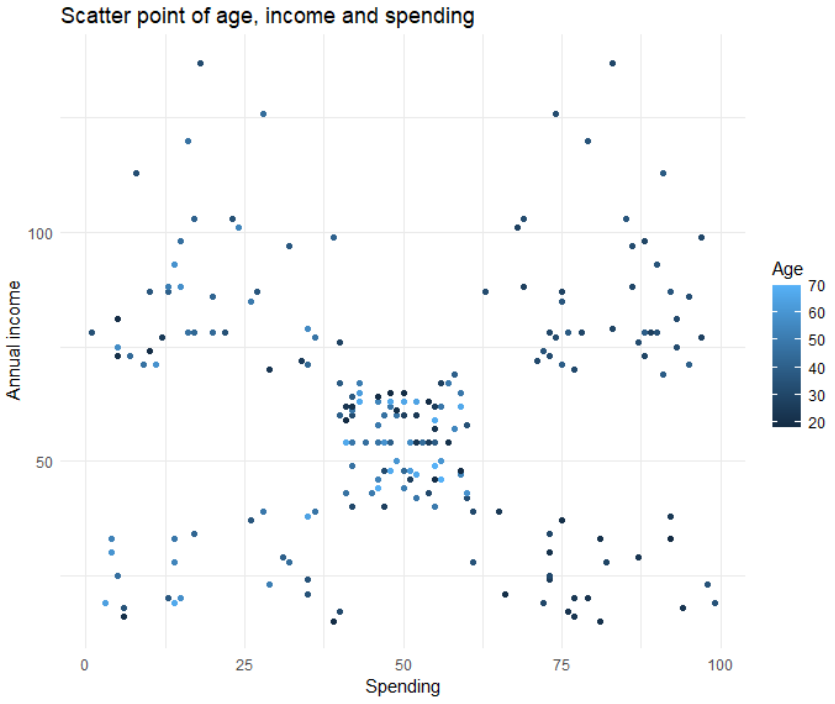
The goal of methods like k-means is to minimize intra-cluster variation. To find the optimal number of clusters, we calculate the total intra-cluster sum of squares (iss) for different values of k. By plotting iss against k, we identify the point where there's a bend or knee, indicating the optimal number of clusters. This approach helps create compact and well-separated clusters.

```
set.seed(123)
iss <- function(k) {
  kmeans(cdata[, 3:5],k,iter.max = 1000, nstart = 1000, algorithm = "Lloyd")$tot.withinss
}
 k.values <- 1:10

 iss_values <- map_dbl(k.values, iss)

plot (k.values, iss_values,
      type = "b" , pch = 19, frame =FALSE,
      xlab = "number of cluster k",
      ylab = "total intra-cluster sum of squares")
```



From the graph, we conclude that 5 is the appropriate number of clusters since it seems to be appearing at the bend in the elbow plot.

We can observe the scatter plot before the implementation of our k-means algorithm.



Finally, we conclude with the implementation of our algorithm :

```
variables <- cdata [, c("Spending.Score..1.100.", "Annual.Income..k..")]
scaled_var <- scale (variables)

k <- 5
kmeans_result <- kmeans(scaled_var, centers = k)

cat("cluster centers (age, spending score, annual income):\n")
print(kmeans_result$centers)

cat("\nAffectation of the cluster for each observation:\n")
print(kmeans_result$cluster)
```

This block of code performs k-means clustering on the selected variables from the dataset (`Spending.Score..1.100.` and `Annual.Income..k..`). Firstly, it standardizes the variables by centering and scaling them using the `scale()` function. Then, it specifies the number of clusters to be created (`k <- 5`). Next, the k-means algorithm is applied to the standardized variables with the chosen number of clusters using the `kmeans()` function. The resulting cluster centers are printed out, representing the mean values of each variable within each cluster. Additionally, the cluster assignments for each observation are printed, indicating which cluster each data point belongs to. This process provides insight into how the data is segmented into different groups based on spending score and annual income, facilitating further analysis and decision-making.

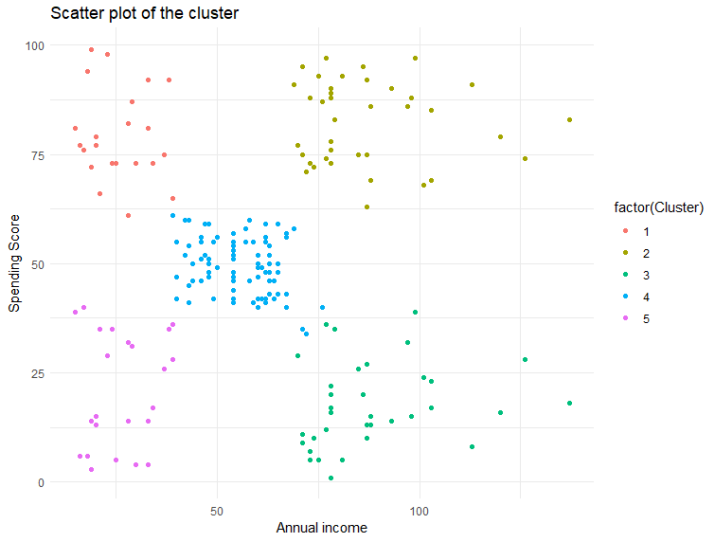
Result :

```
R 4.3.0 · C:/Users/asus/Desktop/pr/
> variables <- cdata [, c("Spending.Score..1.100.", "Annual.Income..k..")]
> scaled_var <- scale (variables)
>
> k <- 5
> kmeans_result <- kmeans(scaled_var, centers = k)
>
> cat("cluster centers (age, spending score, annual income):\n")
cluster centers (age, spending score, annual income):
> print(kmeans_result$centers)
  Spending.Score..1.100. Annual.Income..k..
1            -1.28122394          1.0523622
2            -1.13411939         -1.3042458
3             1.23640011          0.9891010
4            -0.02638995         -0.2004097
5             1.12934389         -1.3262173
>
> cat("\nAffectation of the cluster for each observation:\n")

Affectation of the cluster for each observation:
> print(kmeans_result$cluster)
  [1] 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 5 2 4 2 5 4 4 4 4
 [51] 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
[101] 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 1 3 4 3 1 3 1 3 4 3 1 3 1 3 1 3 1 3 4 3 1 3 1 3 1 3
[151] 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3
>
```

Now, the scatter plot after the implementation of our k-means algorithm shows us 5 clusters:

```
cdata$Cluster <- kmeans_result$cluster

ggplot( cdata, aes(x = Annual.Income..k.. , y = Spending.Score..1.100., color = factor(Cluster))) +
  geom_point() +
  labs (title = "Scatter plot of the cluster",
        x = "Annual income",
        y = "Spending Score") +

  theme_minimal()
```

Result :



- Cluster 4 – These clusters represent the customer data with the medium income salary as well as the medium annual spend of salary.
- Cluster 2 – This cluster represents the customer data having a high annual income as well as a high annual spend.
- Cluster 5 – This cluster denotes the customer data with low annual income as well as low yearly spend of income.
- Cluster 3 – This cluster denotes a high annual income and low yearly spend.
- Cluster 1 – This cluster represents a low annual income but its high yearly expenditure.

Conclusion

In conclusion, this study highlights the pivotal role of customer segmentation in modern business strategies, enabled by the fusion of data analytics and machine learning techniques. Through rigorous data exploration and the application of the K-means clustering algorithm, distinct customer segments have been identified based on income levels and spending behaviors. These insights offer valuable guidance for targeted marketing initiatives and personalized customer engagement, ultimately driving enhanced business performance and customer satisfaction in today's data-driven landscape.