Class 6 Customer Segmentation using Unsupervised Machine Learning

Dr Wei Miao

UCL School of Management

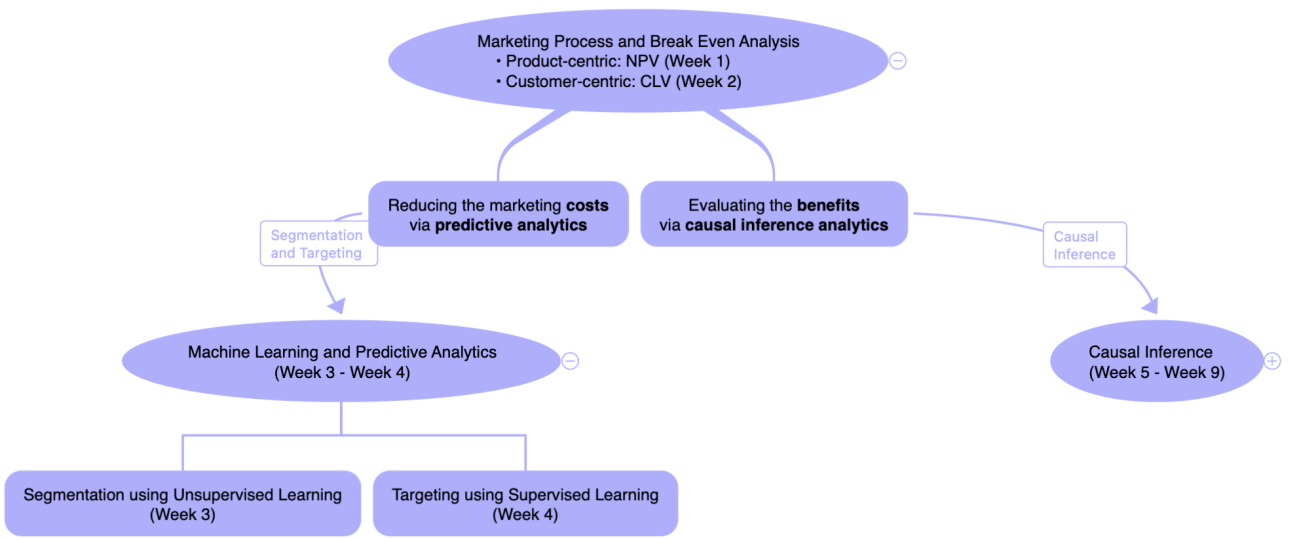
October 18, 2023

Section 1

Overview of Predictive Analytics

Roadmap of Predictive Analytics

- The core of any business decision is **break-even analysis** (BEQ; NPV; CLV. Weeks 1 and 2)
- One effective way to increase firm profitability is to **reduce marketing costs**
- In Weeks 3 and 4, we will learn how to utilize **predictive analytics (i.e., machine learning models)** to reduce marketing costs and improve marketing efficiency

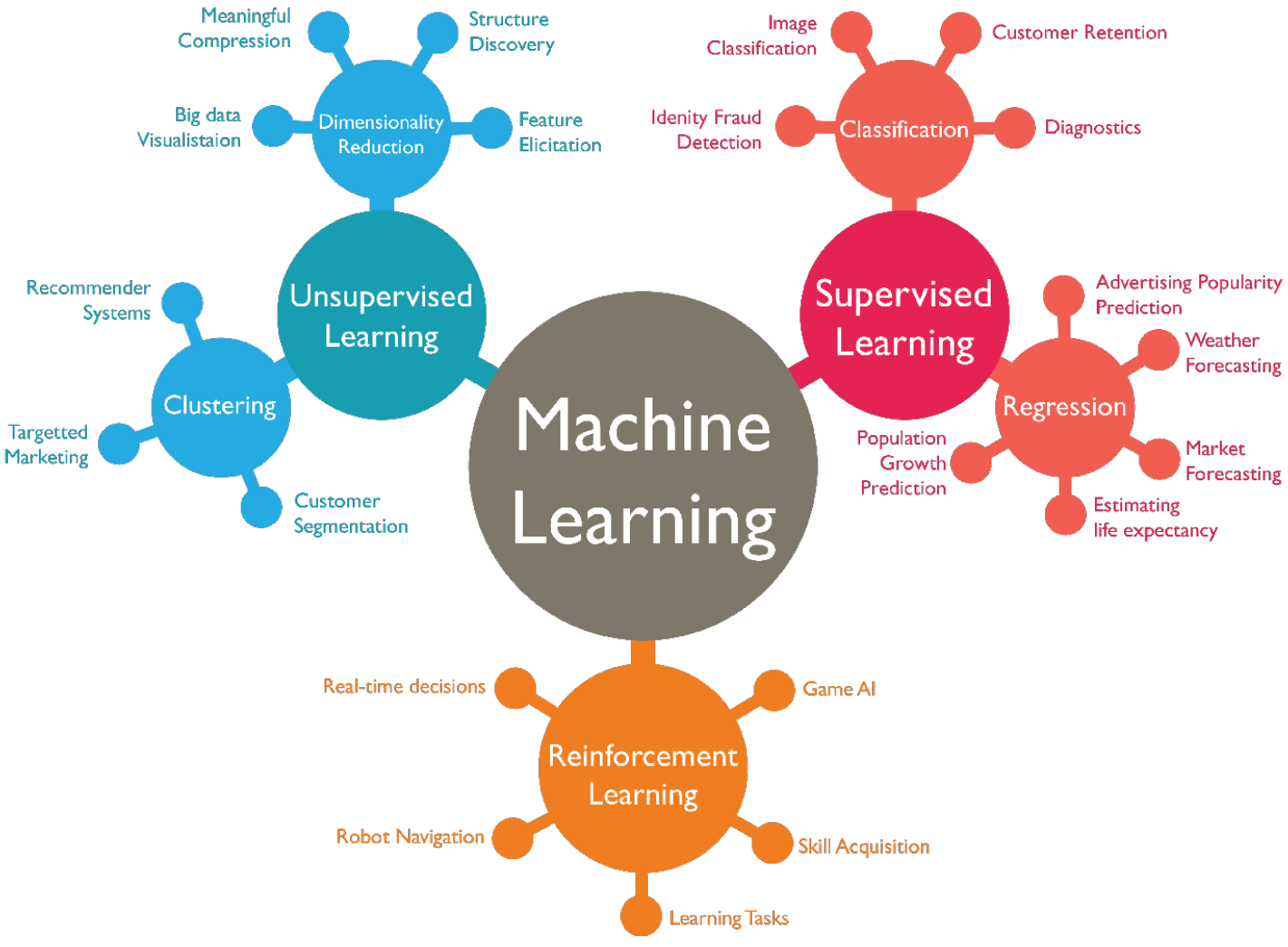


Types of Predictive Analytics

- Unsupervised Learning
  - Only observe X => Want to uncover unknown subgroups
- Supervised Learning
  - Observe both X and Y => Want to predict Y for new data
- Reinforcement Learning
  - Rewards and punishments => Learn the best decision rules
  - Dynamic Coupon Targeting Using Batch Deep Reinforcement Learning: An Application to Livestream Shopping

In Term 2, you will learn predictive analytics models systematically. By then, think about how those techniques can be applied back to these case studies.

Types of Predictive Analytics



Learning Objectives

- Understand the concept of unsupervised learning and how to apply clustering analyses for customer segmentation

Section 2

Segmentation with Unsupervised Learning

Customer Segmentation

Segmentation is the first step in the strategy of marketing (STP), which is the process of dividing customers into meaningful groups based on any characteristics relevant to design and execution of your marketing strategy.

It assumes that different customer groups offer different levels of value to the company and/or require different marketing programs to succeed with (e.g., based on different goals and needs).

Conventional Segmentation

- **Customer value segmentation** is for targeting decisions based on customers' potential long-term financial and strategic value to your company.
- **Benefit segmentation** is for positioning and marketing mix design on the basis of customer and consumer goals or usage, the needs, wants, problems and the trade-offs they are willing to make across benefits (e.g., price vs. quality).
- **Psychographic segmentation** is for positioning and marketing mix design based on the psychology of the customer and consumer, including attitudes, identity, lifestyle, personality, etc.
- **Demographic segmentation** uses variables such as age, gender, income, family life cycle, educational qualification, socio-economic status, religion, company size and income, etc. These serve as proxies for goals, preferences or psychographics, as well as to characterize segments for marketing mix decisions.

Conventional segmentation methods require lots of subjective judgments. A more objective way is to "let the data speak" by using data analytics tools.

K-Means Clustering

- K-means clustering is one of the most commonly used unsupervised machine learning algorithms for partitioning a given data set into a set of k groups (i.e. k clusters), where k represents the number of groups pre-specified by the analyst.
- It can classify customers into multiple segments (i.e., clusters), such that customers within the same cluster are as similar as possible, whereas customers from different clusters are as dissimilar as possible.
- Input: (1) customer characteristics; (2) the number of clusters
- Output: cluster membership of each customer

K-Means Clustering: Step 1

(a)

- Raw data points; each dot is a customer
- X and Y axis are customer characteristics
- Obviously there should be 2 segments
- Let's see how K-means uses a data-driven way to classify customers into 2 segments

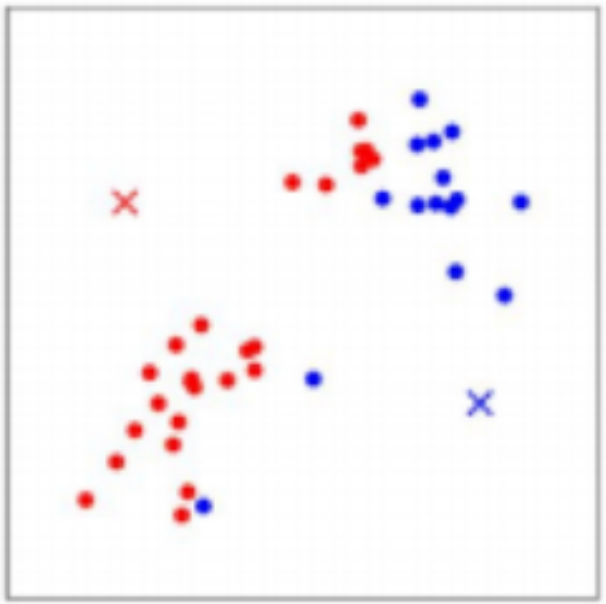
K-Means Clustering: Step 2

(b)

- We specify 2 segments
- K-means initializes the process by **randomly** selecting 2 centroids[a]

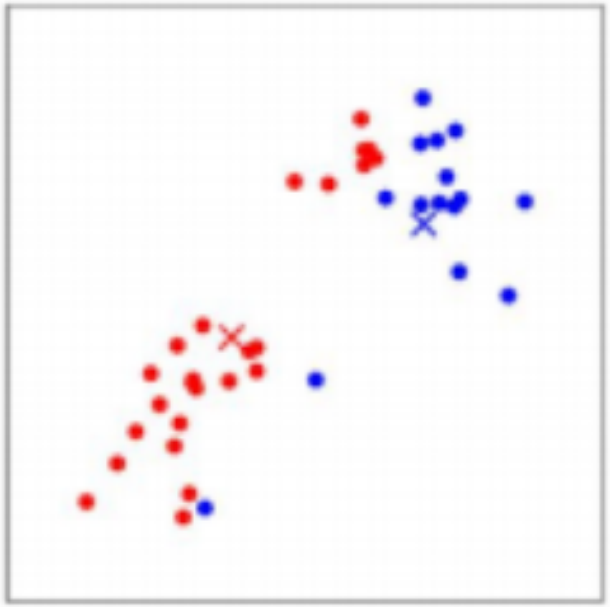
[a] Due to this randomness, each different starting points might give different results. We need to reinitialize the process repeatedly to ensure **robustness** of results.

K-Means Clustering: Step 3



- K-means computes the distance of each customer to the red and blue centroids
- K-means assigns each customer to red segment or blue segment based on which centroid is closer

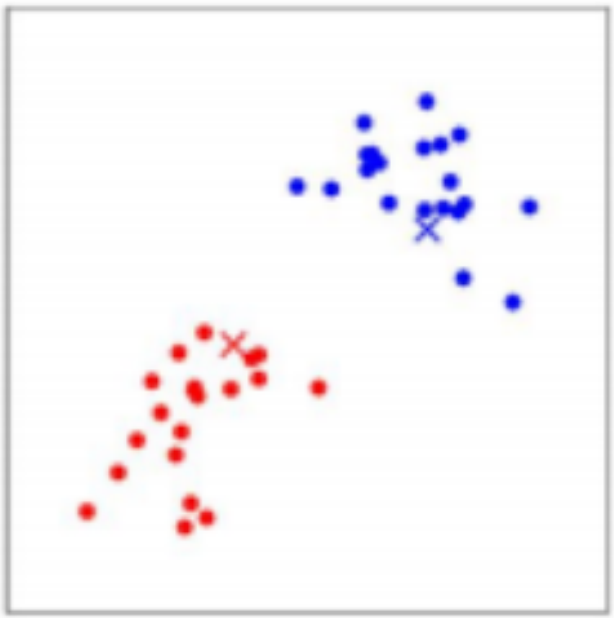
K-Means Clustering: Step 4



(d)

- K-means updates the new centroids of each segment
- The red cross and blue cross in the picture are the new centroids
- We still see some "outliers", so need to continue the algorithm

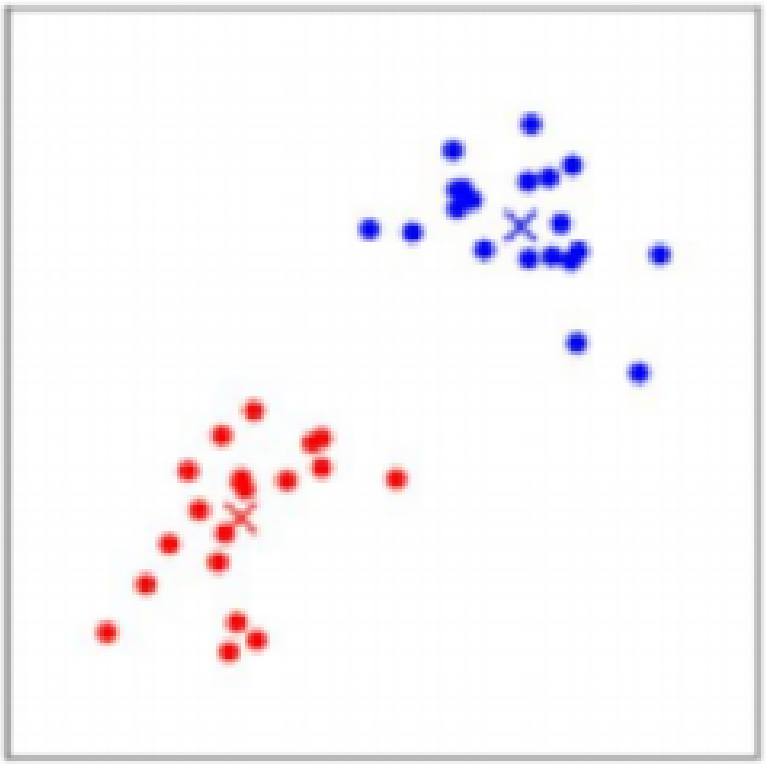
K-Means Clustering: Step 5



(e)

- K-means computes the distance of each customer to the red and blue centroids
- K-means updates each customer to red segment or blue segment based on which centroid is closer
- Now the outliers are correctly assigned each segment

K-Means Clustering: Step 6



(f)

- K-means updates the new centroid from the previous segmentation
- K-means computes the distance of each customer to the new centroids
- K-means finds that all customers are correctly segmented to nearest centroids, so no need to continue
- As the algorithm **converges**, the algorithm stops

Section 3

Customer Segmentation for Tesco

Syntax of `kmeans()`

1. Decide to do customer segmentation based on *total spending* and *income*

```
kmeans(x, centers, iter.max = 10, nstart = 1,
       algorithm = c("Hartigan-Wong", "Lloyd", "Forgy",
                     "MacQueen"), trace=FALSE)
```

- `x`: data with selected variables to apply K-means
- `centers`: number of clusters
- `iter.max`: the maximum number of iterations allowed
- `nstart`: how many random sets should be chosen
- `algorithm`: which algorithm to choose; default often works
- `trace`: do you want to trace intermediate steps?

Data collection and cleaning

- Need to re-scale the two variables using `scale()`, because the two variables are of very different scales
  - **This is extremely important!**
  - `set.seed()` is to allow replication of results.
  - Refer to this data camp tutorial for more details.

```
data_kmeans <- data_full%>%
  select(Income,total_spending)%>%
  mutate(Income = scale(Income),
         total_spending = scale(total_spending))
```

Conduct K-means clustering

```
set.seed(888)
result_kmeans <- kmeans(data_kmeans,
                        centers = 2,
                        nstart = 10)
```

Examine the returned object, `result_kmeans`

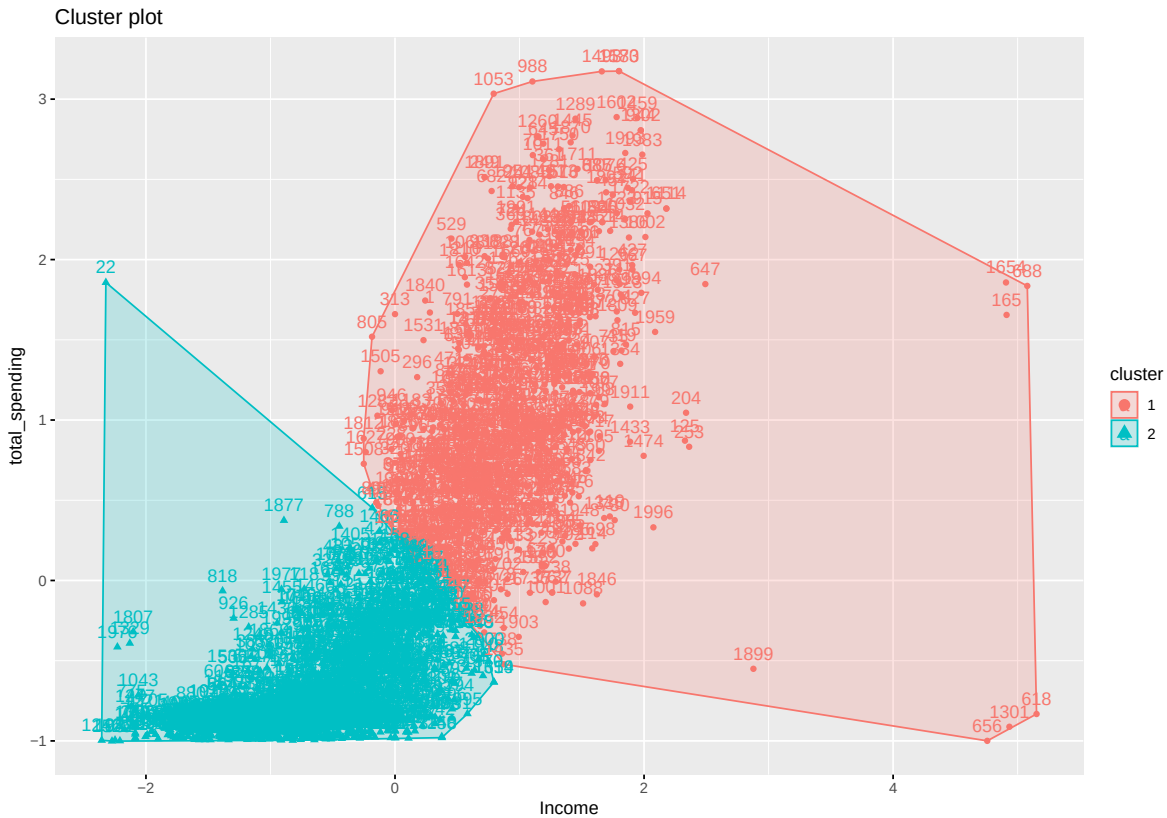
```
1 str(result_kmeans)
```

- `cluster`: **A vector of integers (from 1:k) indicating the cluster to which each point is allocated.**
- `centers`: A matrix of cluster centers.
- `totss`: The total sum of squares.
- `withinss`: Vector of within-cluster sum of squares, one component per cluster.
- `tot.withinss`: Total within-cluster sum of squares, i.e. sum(withinss).
- `betweenss`: The between-cluster sum of squares, i.e. $totss-tot.withinss$.
- `size`: The number of points in each cluster.

Visualize the clusters

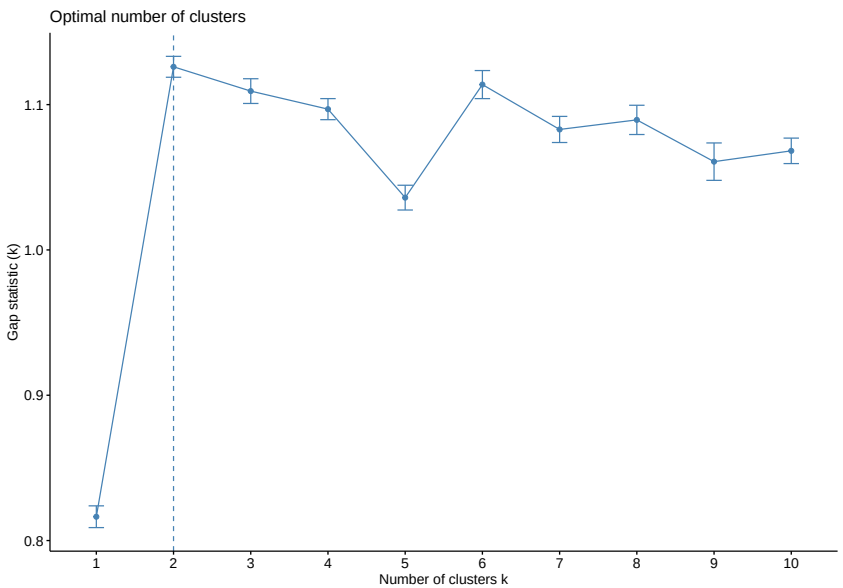
- We need 2 packages `cluster` and `factoextra`
- Use function `fviz_cluster()` to generate visualizations

```
pacman::p_load(cluster,factoextra)
set.seed(888)
fviz_cluster(result_kmeans,
             data = data_kmeans)
```



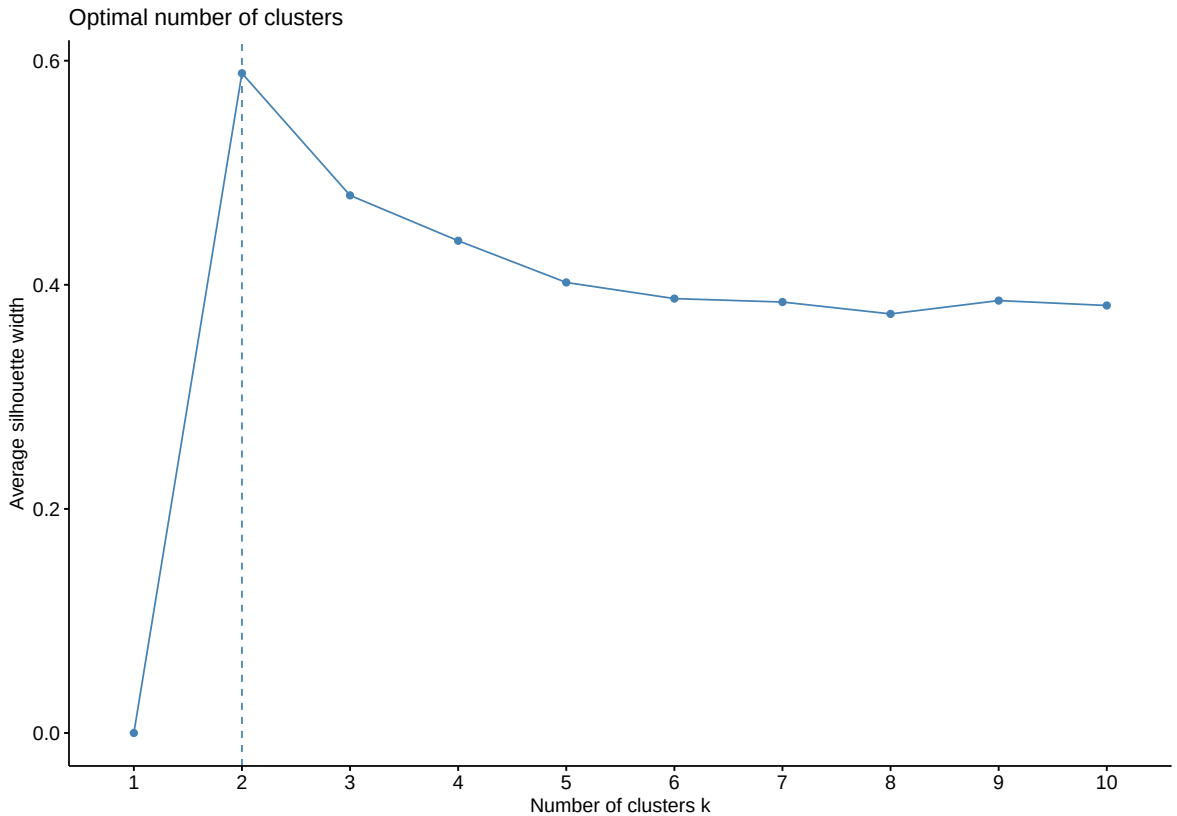
Determine the optimal number of clusters: GAP Method

```
set.seed(888)
gap_stat <- clusGap(data_kmeans,
                    FUN = kmeans,
                    K.max = 10,
                    B = 50)
fviz_gap_stat(gap_stat)
```



Determine the optimal number of clusters: Silhouette Method

```
set.seed(888)
fviz_nbclust(data_kmeans, kmeans, method = "silhouette")
```



Business Implications

- Compare the CLV in the two segments, and decide which segment to serve.
  - This is a general idea of segmentation and targeting using unsupervised learning
  - Finish this exercise after class

After-Class Readings

- Useful source: K-means Cluster Analysis