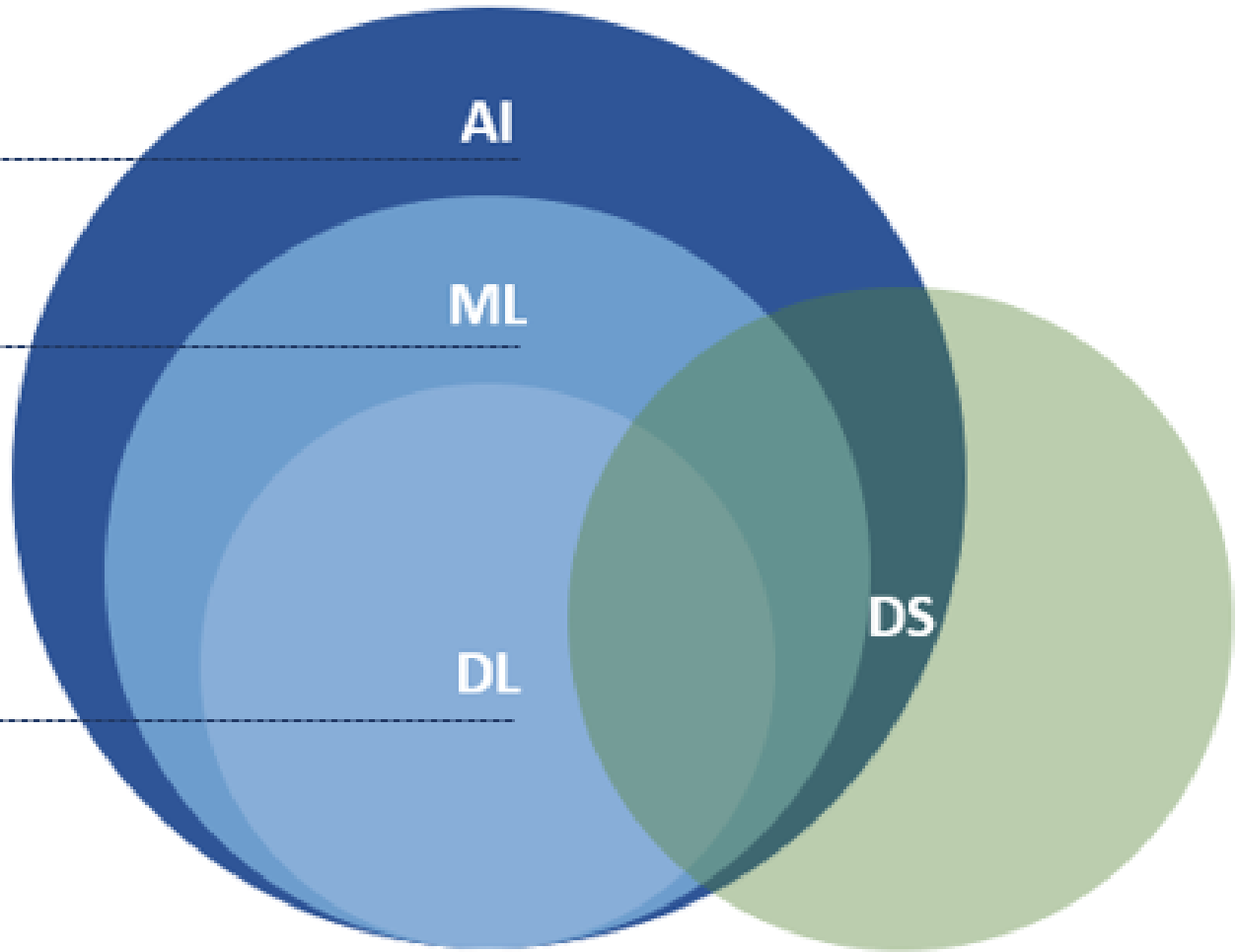
Computational Intelligence and Machine Learning

Anupama Sindgi

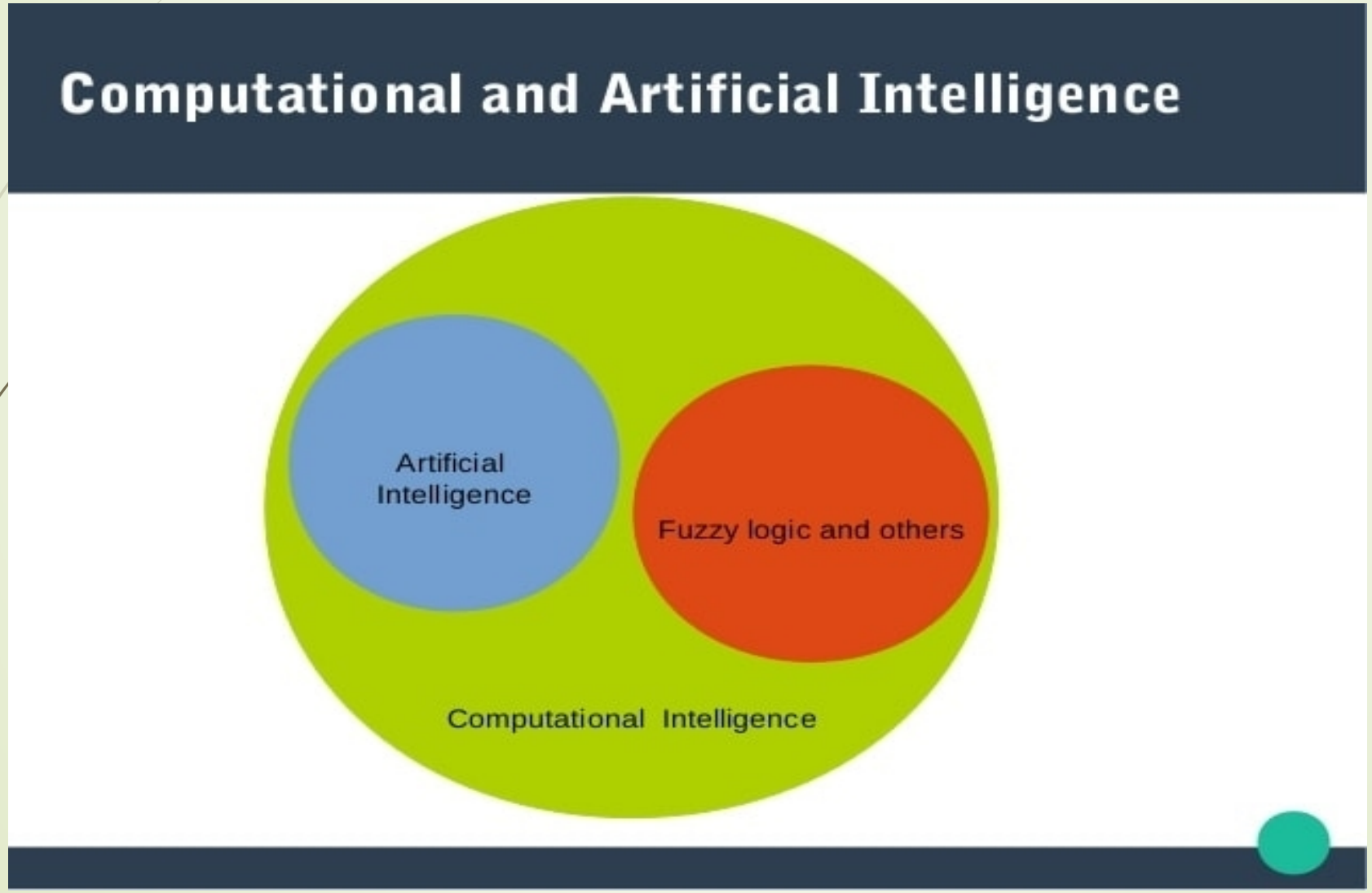
ARTIFICIAL INTELLIGENCE
machines simulate the human thinking process through methods ranging from simple if-then statements to complex models.

MACHINE LEARNING
machines analyse vast amounts of data searching for patterns to answer a very specific question. ML improves its decisions based on experience.

DEEP LEARNING
is based on deep neural networks similar to the networks of the human brain. These machines work with deep models (i.e. models with several layers).



Introduction



General Intelligence: to perform intellectual task that a human can

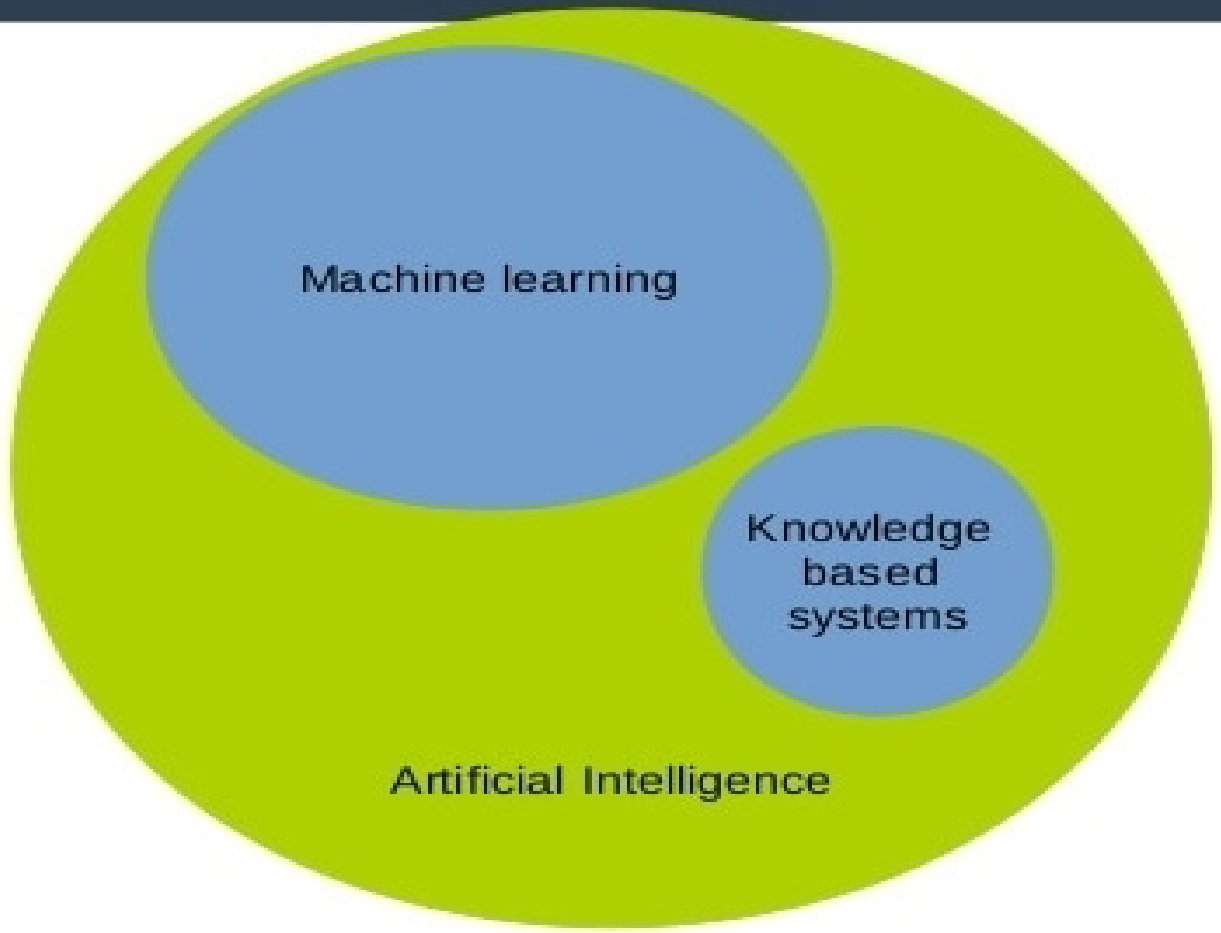
RELATIONSHIP WITH AI

AI	CI
• Make the machine think as we humans do	• Make the machine think as we humans do using biologically inspired methods

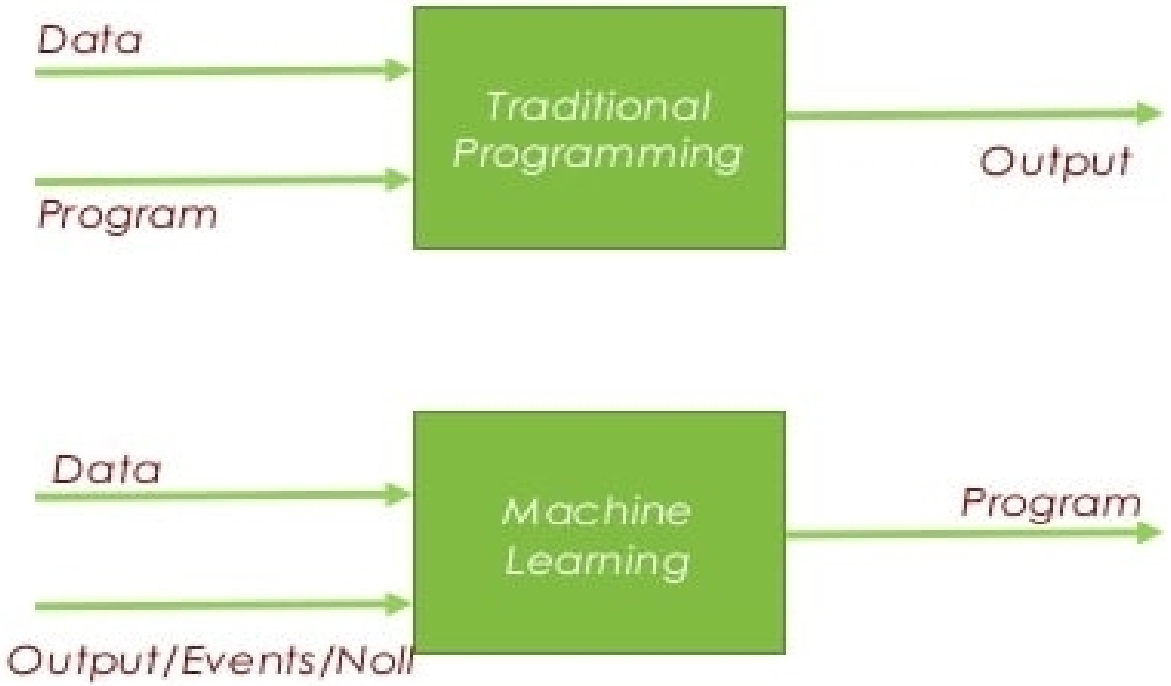
CI vs AI

Computational Intelligence	Artificial Intelligence
Soft Computing techniques	Hard computing techniques
Follows fuzzy logic	Follows binary logic
Nature inspired models	Based on mathematical models
Can work inexact and incomplete data	Not very effective
Probabilistic results	Deterministic results

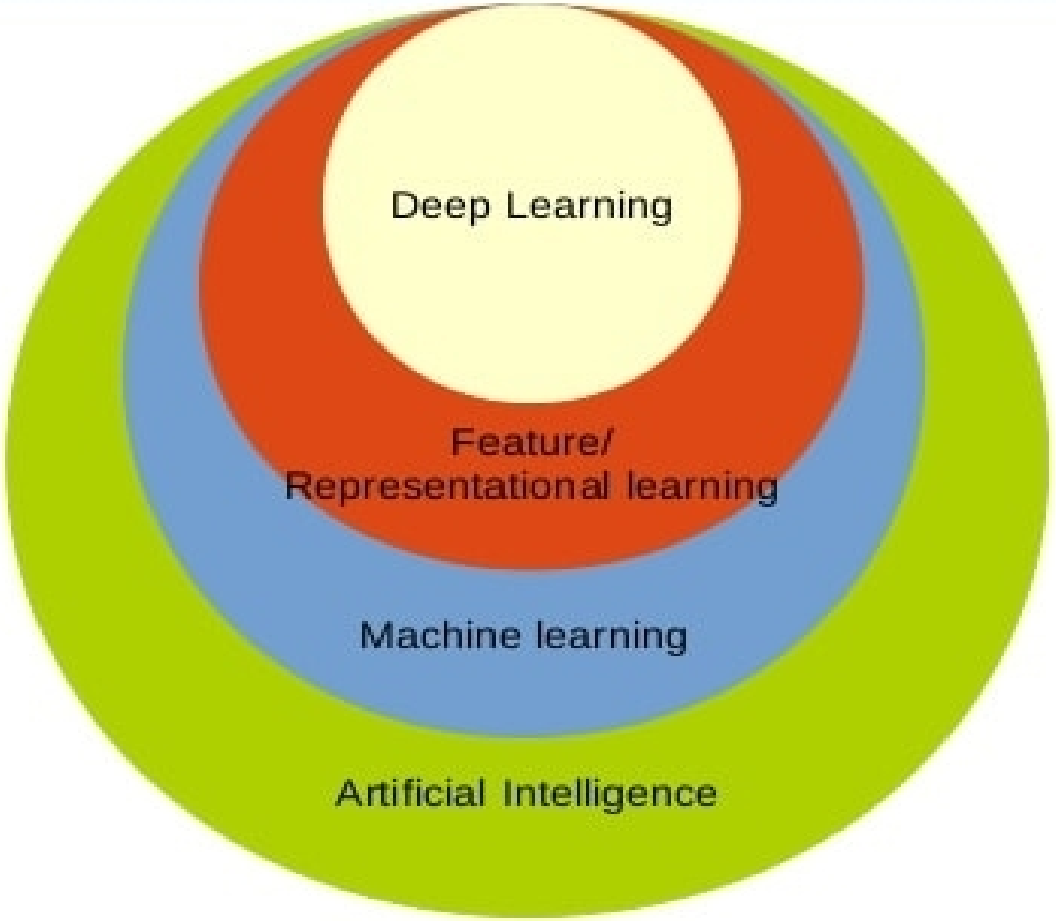
AI and ML


Machine learning
Knowledge based systems
Artificial Intelligence

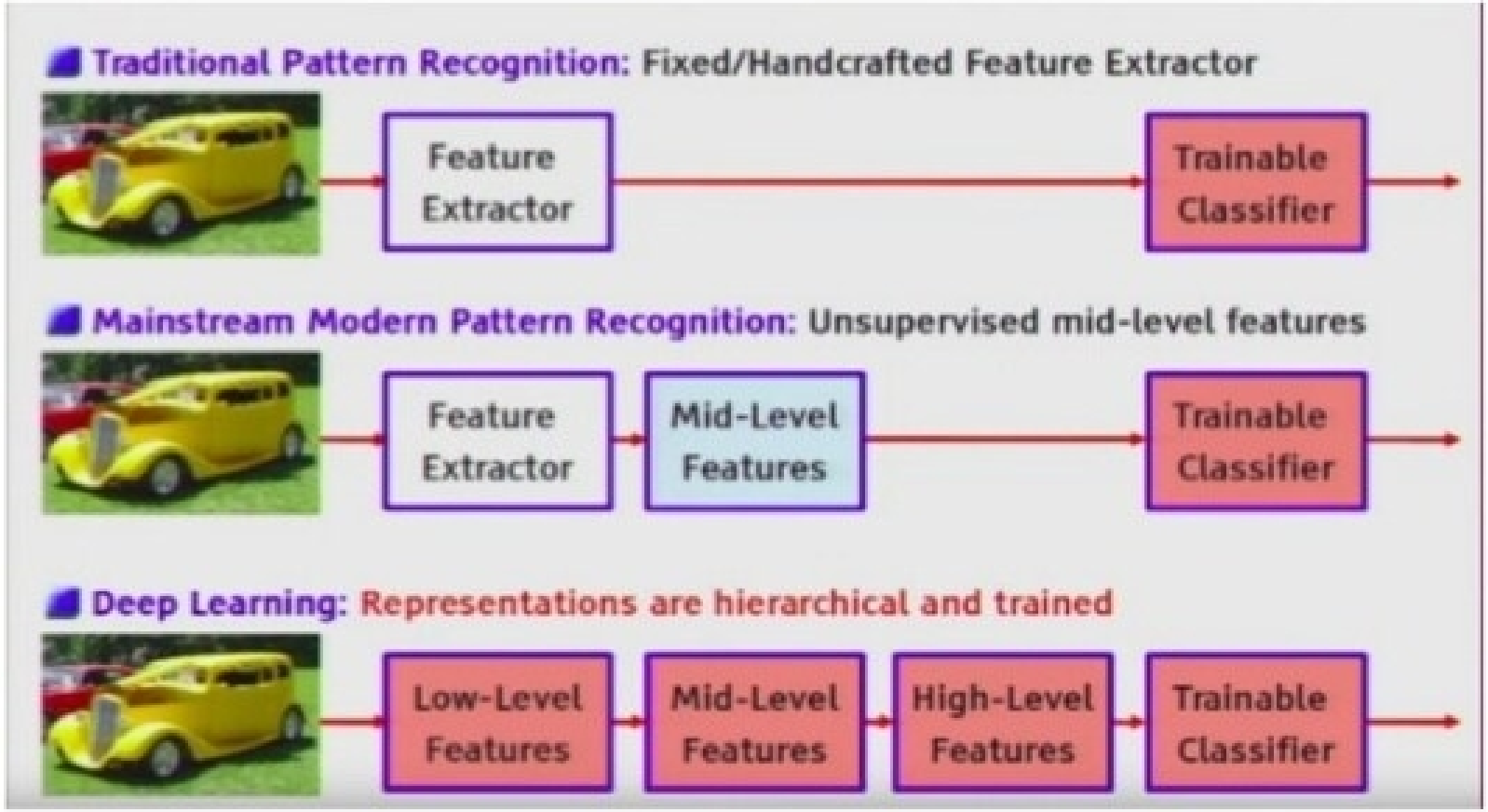
Machine Learning



AI, ML, DL

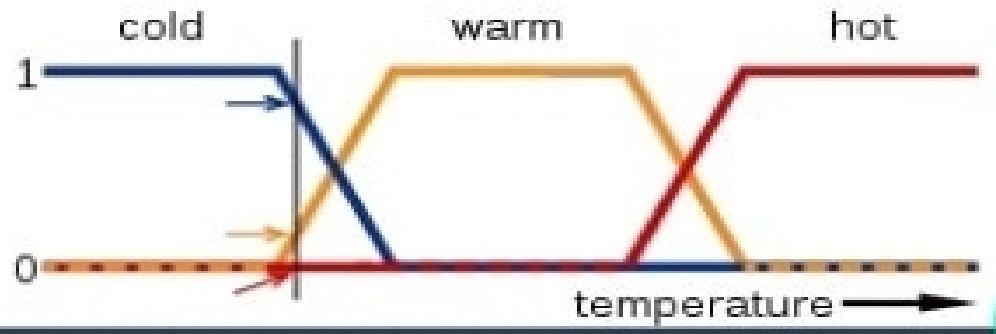

Deep Learning
Feature/
Representational learning
Machine learning
Artificial Intelligence

Deep learing

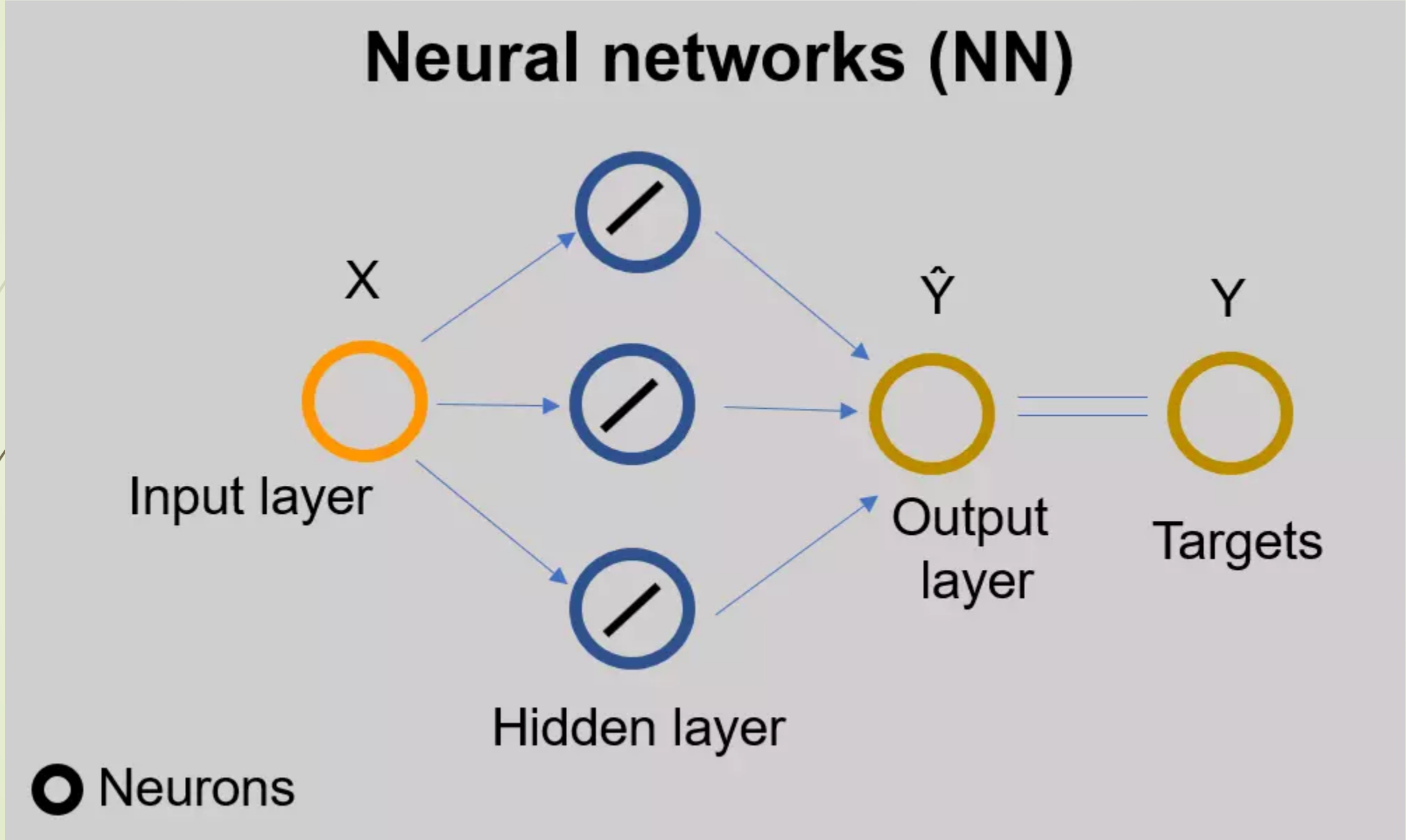


Fuzzy Logic

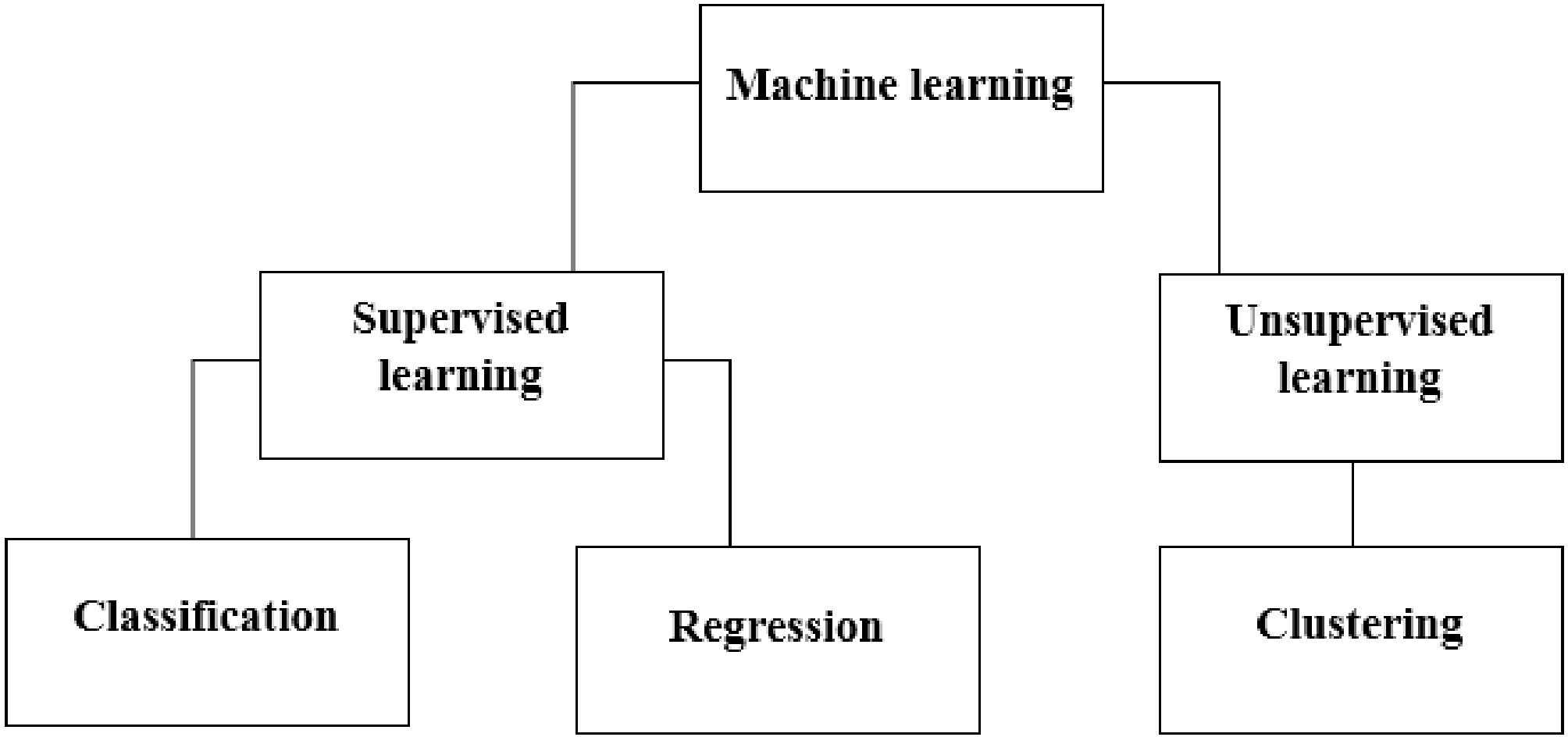
- Multi valued logic
- Many applications



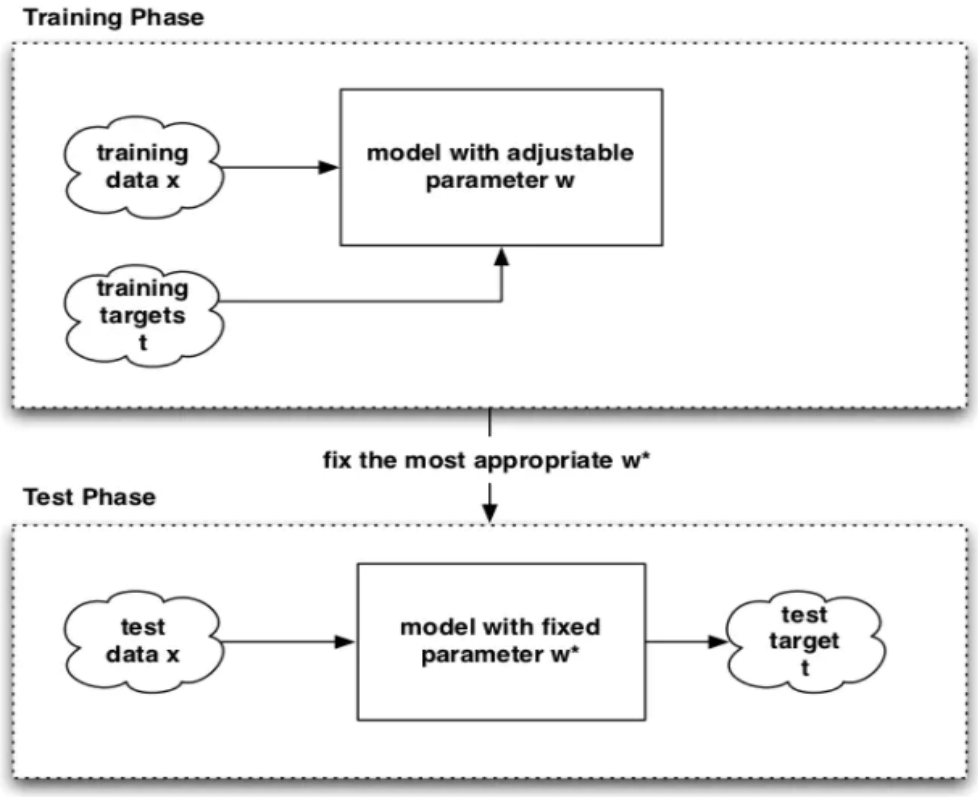
Basics of Neural Networks



Microsoft
werPoint Presentat


Machine learning
Supervised learning
Unsupervised learning
Classification
Regression
Clustering

Supervised Learning



©2011 Christfried Webers
NICTA
The Australian National University

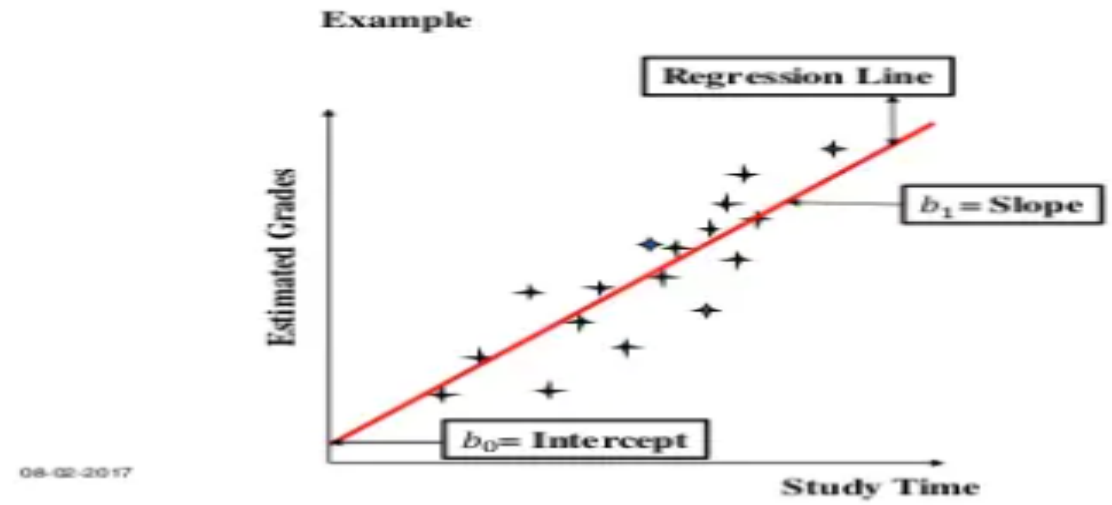
ISML 2011

What is linear regression in machine learning?

What is Regression Analysis?

- ✓ Regression analysis is a form of predictive modelling technique which investigates the relationship between a dependent (target) and independent variable(s) (predictor).
- ✓ This technique is used for forecasting, time series modelling and finding the causal effect relationship between the variables.
- ✓ For example, relationship between rash driving and number of road accidents by a driver is best studied through regression.

Population Regression Line

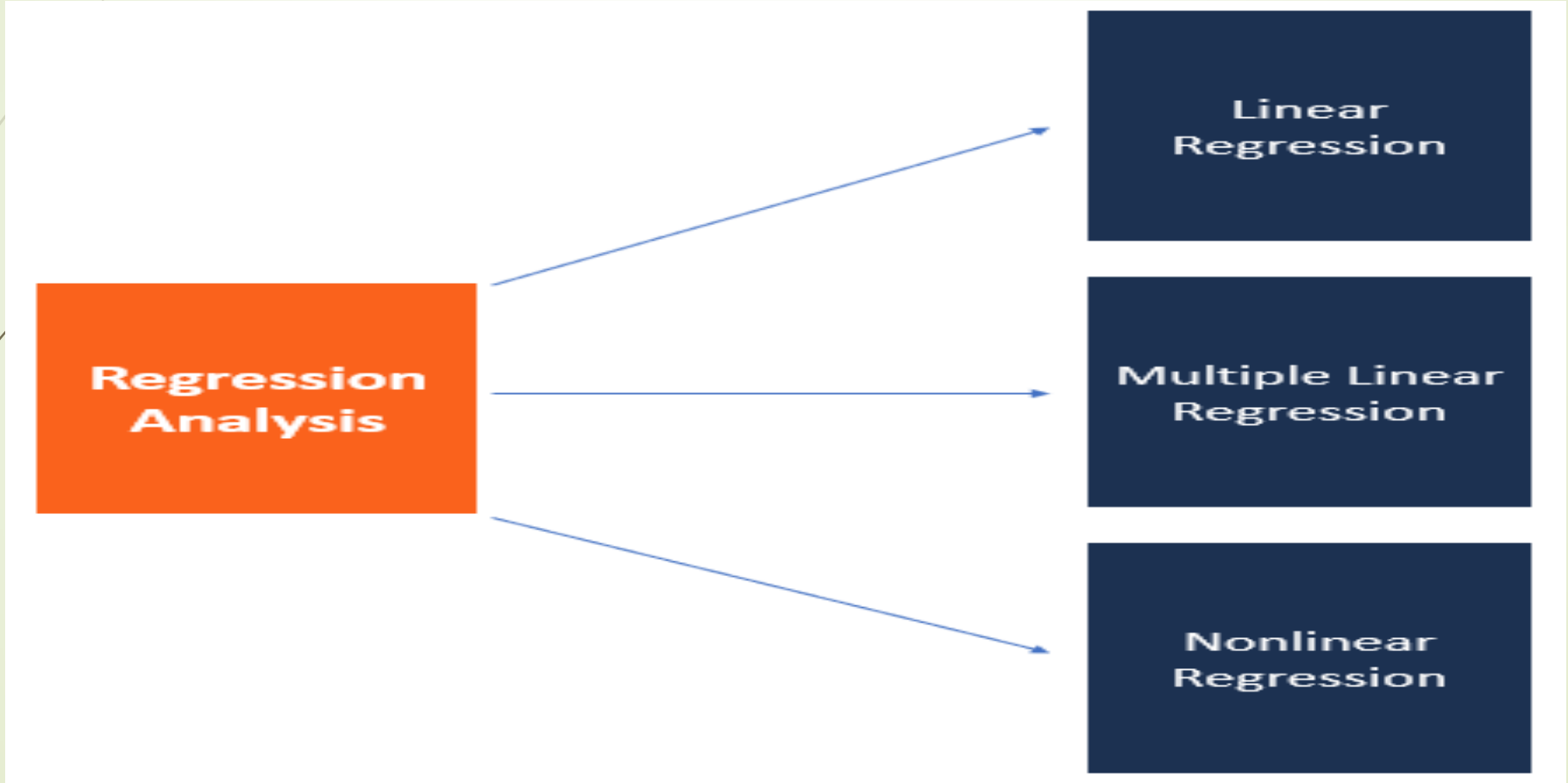


Population regression function =

$$\hat{y} = b_0 + b_1 x$$

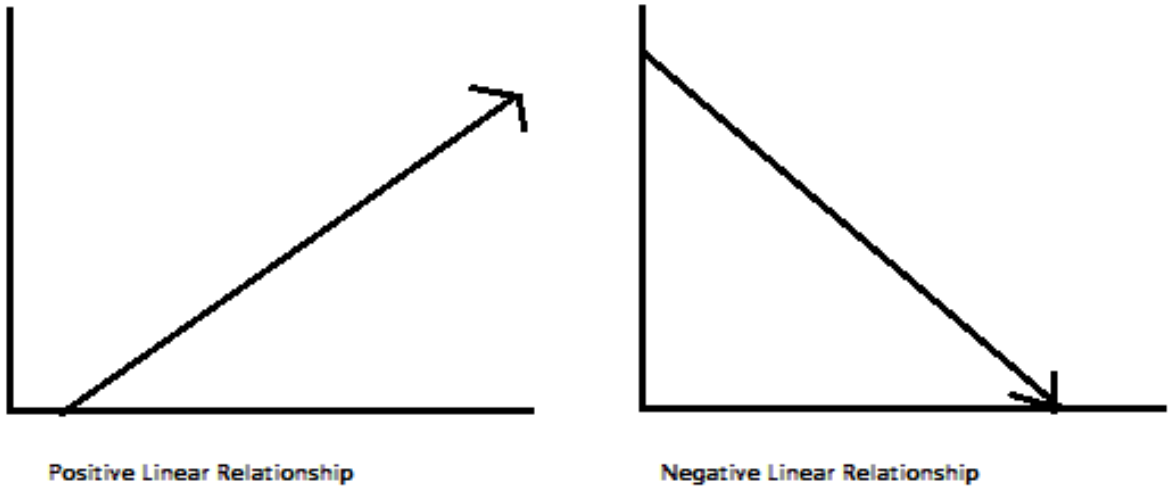
$\hat{y}$ = Estimated Grades
x = Study Time
b_0 = Intercept
b_1 = Slope

- Regression analysis is a statistical method that shows the relationship between two or more variables. Method tests the relationship between a dependent variable against independent variables.



Linear regression

Linear regression is a statistical model that examines the linear relationship between two (Simple Linear Regression) or more (Multiple Linear Regression) variables — a dependent variable and independent variable(s). Linear relationship basically means that when one (or more) independent variables increases (or decreases), the dependent variable increases (or decreases) too



Positive Linear Relationship

Negative Linear Relationship

basic equation for a straight line is:

$$Y=mx+b$$

where

m=the slope

and b=the y-intercept.

Linear Basis Function Models

- Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labeled datasets. But the difference between both is how they are used for different machine learning problems.

- The main difference between Regression and Classification algorithms that Regression algorithms are used to **predict the continuous** values such as price, salary, age, etc. and Classification algorithms are used to **predict/Classify the discrete values** such as Male or Female, True or False, Spam or Not Spam, etc.

Linear basic function Model

The simplest linear model for regression is one that uses linear combination of input variables

Linear basis function models

Linear regression

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \cdots + w_D x_D,$$

where $\mathbf{x} = (x_1, \ldots, x_D)$

In our example,

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + w_2 x_2$$

Note: Linear regression (LR) models the linear relationship between the independent (X) variable with that of the dependent variable (y).

- **To extend the "Class of Model", we have to consider the linear combination of fixed non-linear function of the input variables**

Generally

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x})$$

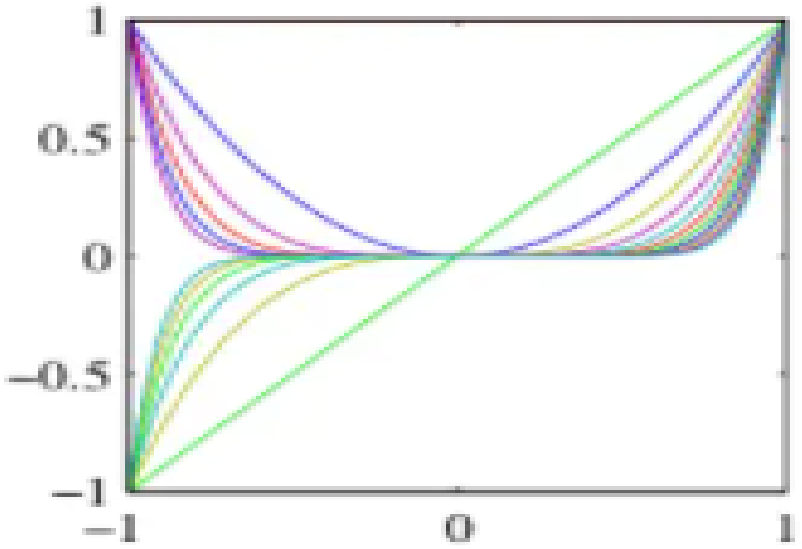
where $\phi_j(\mathbf{x})$ are known as **basis functions**.
Typically, $\phi_0(\mathbf{x}) = 1$, so that w_0 acts as a bias.
In the simplest case, we use linear basis functions:
$\phi_d(\mathbf{x}) = x_d$.

Polynomial Basis Functions

- Scalar input variable x

$$\phi_j(x) = x^j$$

- Limitation : Polynomials are global functions of the input variable x.
- Extension: Split the input space into regions and fit a different polynomial to each region (spline functions).

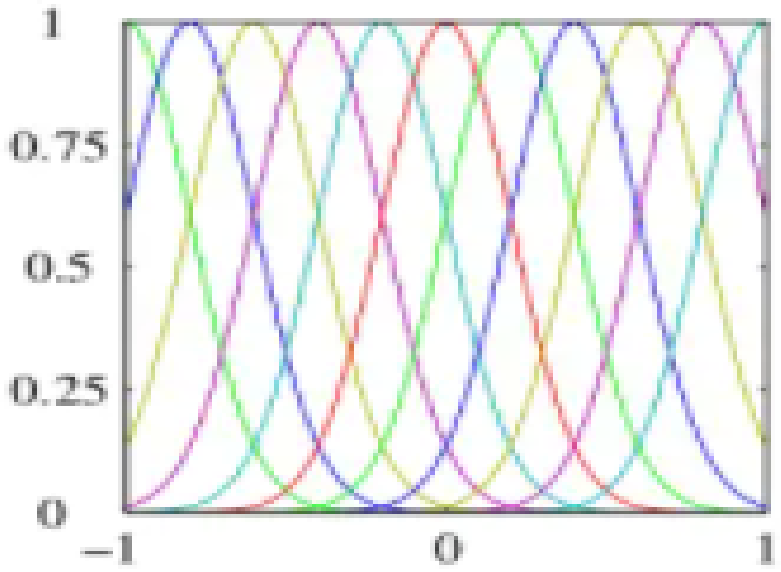


'Gaussian' Basis Functions

- Scalar input variable x

$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$

- Not a probability distribution.
- No normalisation required, taken care of by the model parameters w.



Sigmoidal Basis Functions

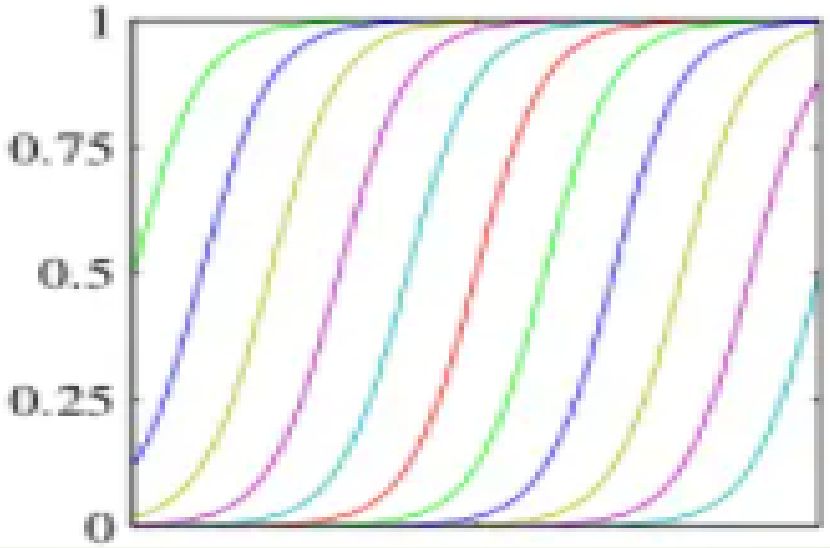
- Scalar input variable x

$$\phi_j(x) = \sigma\left(\frac{x - \mu_j}{s}\right)$$

where $\sigma(a)$ is the logistic sigmoid function defined by

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

- $\sigma(a)$ is related to the hyperbolic tangent $\tanh(a)$ by $\tanh(a) = 2\sigma(a) - 1$.



Other Basis Functions

- Fourier Basis : each basis function represents a specific frequency and has infinite spatial extent.
- Wavelets : localised in both space and frequency (also mutually orthogonal to simplify appliciation).
- Splines (polynomials restricted to regions of the input space).

Maximum likelihood

Maximum Likelihood

- There are three major paradigms of estimating linear models
  - Method of Moments
    - Oldest estimation method
    - Population moments are best estimated by sample moments
    - Not too useful for complex estimation
  - Least Squares
    - Minimize the sum of the squared errors
  - Maximum Likelihood Estimation
    - Find the model which has the highest probability of producing the observed data (or the maximum likelihood)

- **Maximum likelihood estimation (MLE)** is a technique used for estimating the parameters of a given distribution, using some observed data. For example, if a population is known to follow a normal distribution but the mean and variance are unknown, MLE can be used to estimate them using a limited sample of the population, by finding particular values of the mean and variance so that the observation is the most likely result to have occurred.

- MLE is usually recommended for large samples because it is versatile, applicable to most models and different types of data, and produces the most precise estimates.

Maximum likelihood and least squares:

Maximum Likelihood and Least Squares

- No special assumption about the basis functions $\phi_j(\mathbf{x})$. In the simplest case, one can think of $\phi_j(\mathbf{x}) = \mathbf{x}$.
- Assume target t is given by

$$t = \underbrace{y(\mathbf{x}, \mathbf{w})}_{\text{deterministic}} + \underbrace{\epsilon}_{\text{noise}}$$

where ϵ is a zero-mean Gaussian random variable with precision (inverse variance) β.

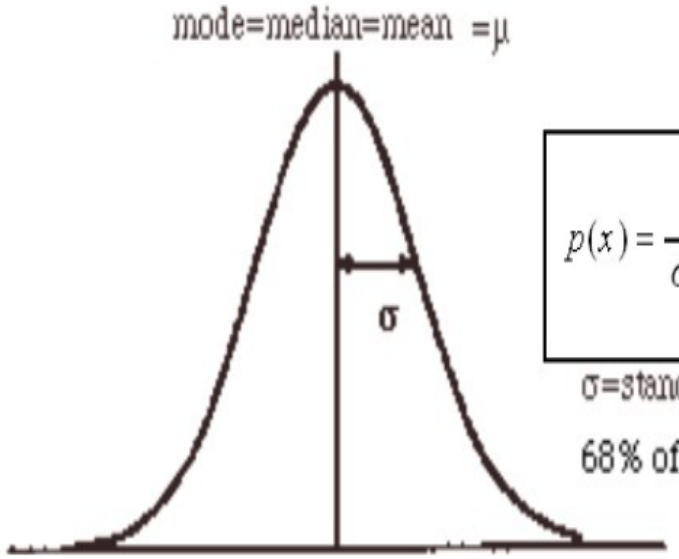
- Thus

$$p(t \mid \mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t \mid y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

Note: The normal (or Gaussian) distribution is a continuous probability distribution that has a bell-shaped probability density function, known as the Gaussian function or informally as the bell curve.

The Gaussian (Normal) Distribution

The **Gaussian Distribution** is one of the most used distributions in all of science. It is also called the "bell curve" or the ***Normal Distribution*** .

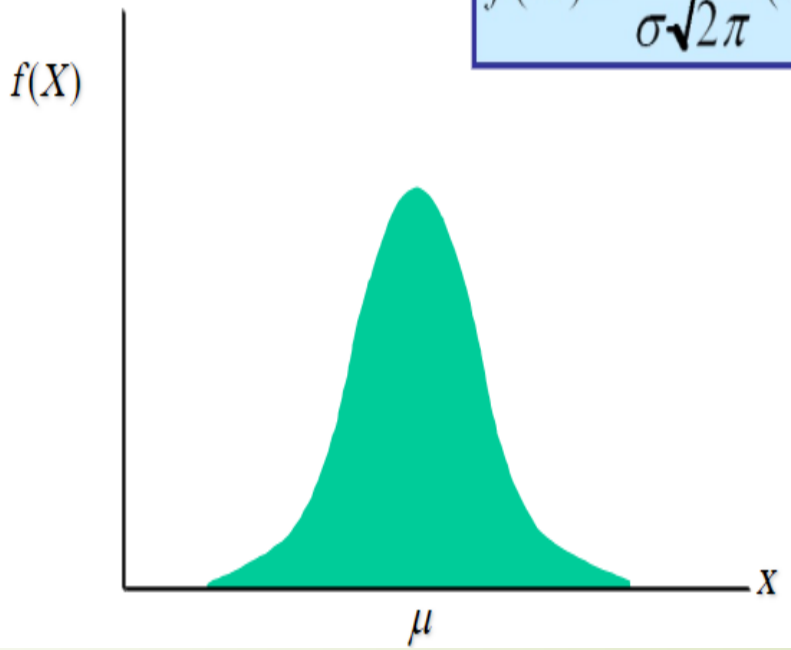


$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad gaussian$$

σ=standard deviation

68% of area within ±1σ

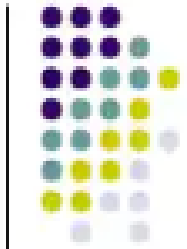
$$f(X) = \frac{1}{\sigma\sqrt{2\pi}} (e)^{-\frac{(X-\mu)^2}{2\sigma^2}}$$



There are only 2 variables that determine the curve, the mean μ & the variance σ. The rest are constants.

MODULE 3

INTRODUCTION-
What is clustering?



- **Clustering** is the classification of objects into different groups, or more precisely, the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait - often according to some defined distance measure.

Types of clustering:

1. **Hierarchical algorithms**: these find successive clusters using previously established clusters.

    1. Agglomerative ("bottom-up"): Agglomerative algorithms begin with each element as a separate cluster and merge them into successively larger clusters.
    2. Divisive ("top-down"): Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters.

2. **Partitional clustering:** Partitional algorithms determine all clusters at once. They include:

    - ***K*-means and derivatives**
    - Fuzzy *c*-means clustering
    - QT clustering algorithm

Common Distance measures:

- *Distance measure* will determine how the *similarity* of two elements is calculated and it will influence the shape of the clusters.

They include:

1. The Euclidean distance (also called 2-norm distance) is given by:

$$d(x, y) = \sum_{i=1}^{p} |x_i - y_i|$$

2. The Manhattan distance (also called taxicab norm or 1-norm) is given by:

$$d(x, y) = \sqrt[2]{\sum_{i=1}^{p} |x_i - y_i|^2}$$

3. The maximum norm is given by:

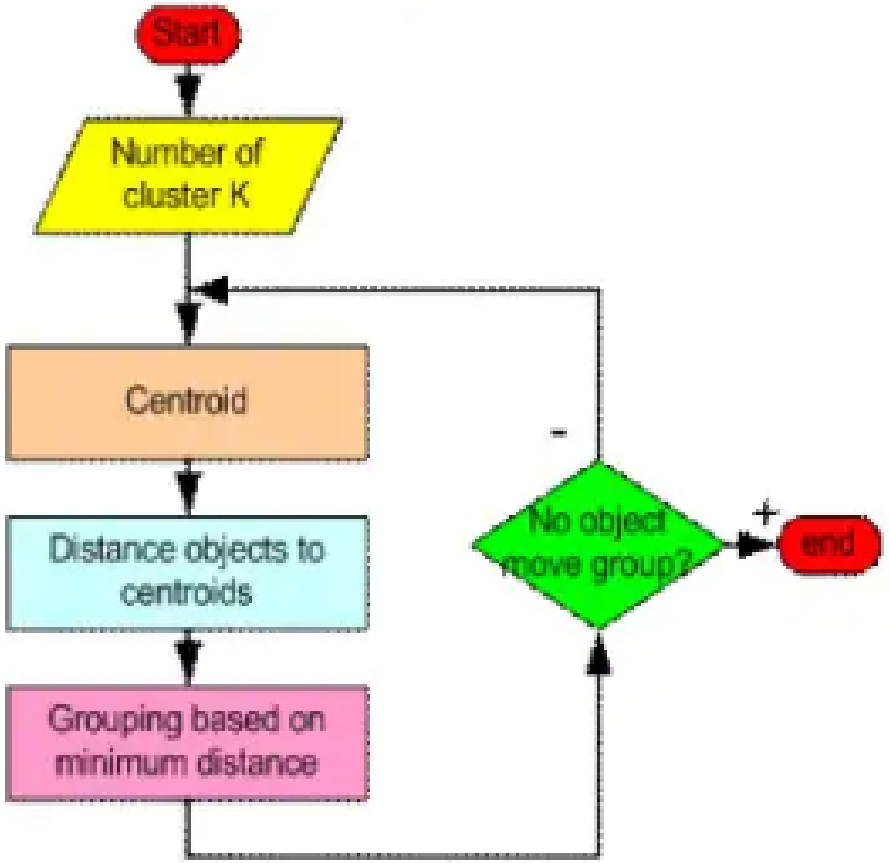
$$d(x, y) = \max_{1 \le i \le p} |x_i - y_i|$$

4. The Mahalanobis distance corrects data for different scales and correlations in the variables.
5. Inner product space: The angle between two vectors can be used as a distance measure when clustering high dimensional data
6. Hamming distance (sometimes edit distance) measures the minimum number of substitutions required to change one member into another.

K-MEANS CLUSTERING

- The **k-means algorithm** is an algorithm to cluster *n* objects based on attributes into *k* partitions, where $k < n$.
- It is similar to the expectation-maximization algorithm for mixtures of Gaussians in that they both attempt to find the centers of natural clusters in the data.
- It assumes that the object attributes form a vector space.

How the K-Mean Clustering algorithm works?

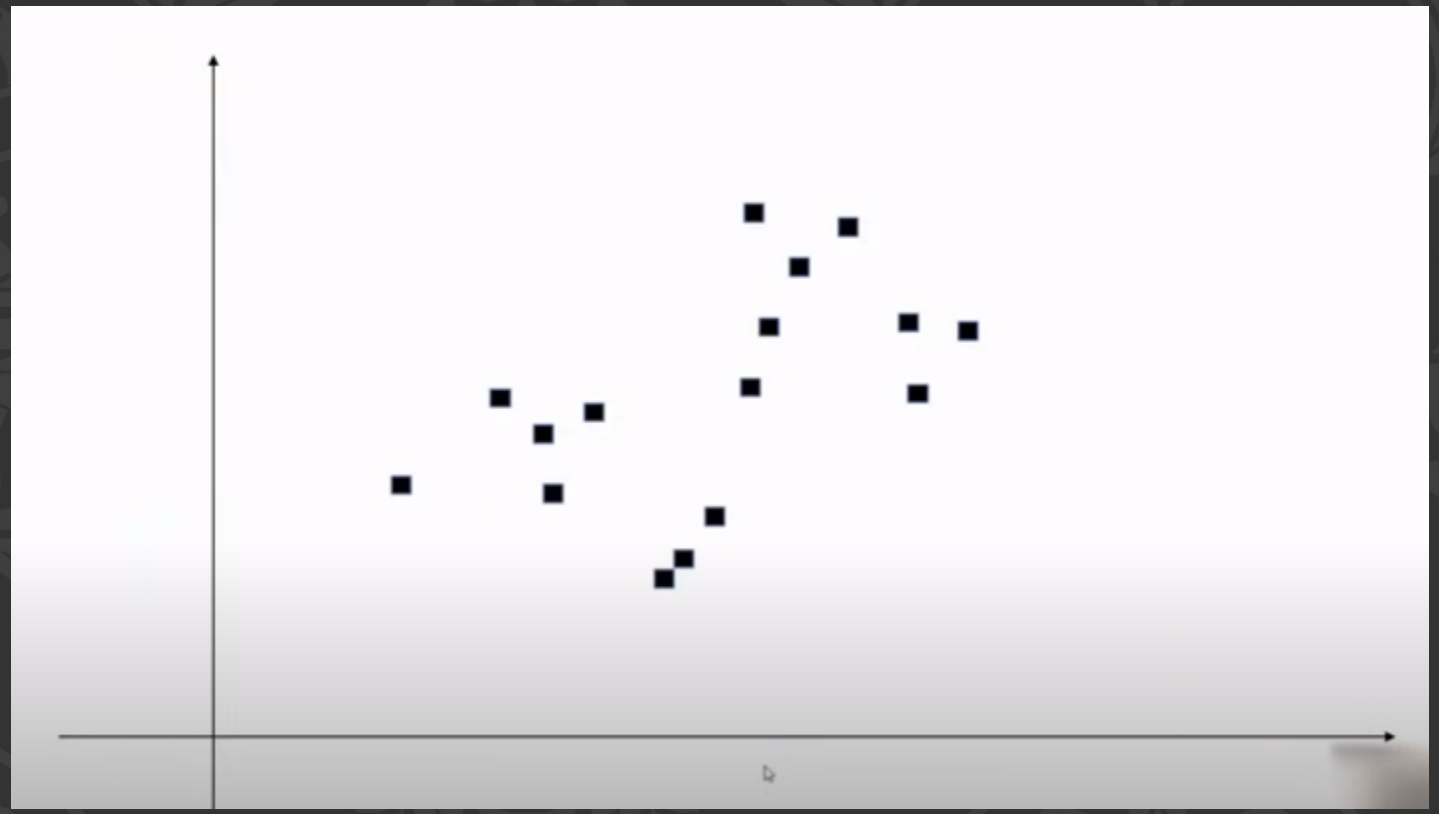


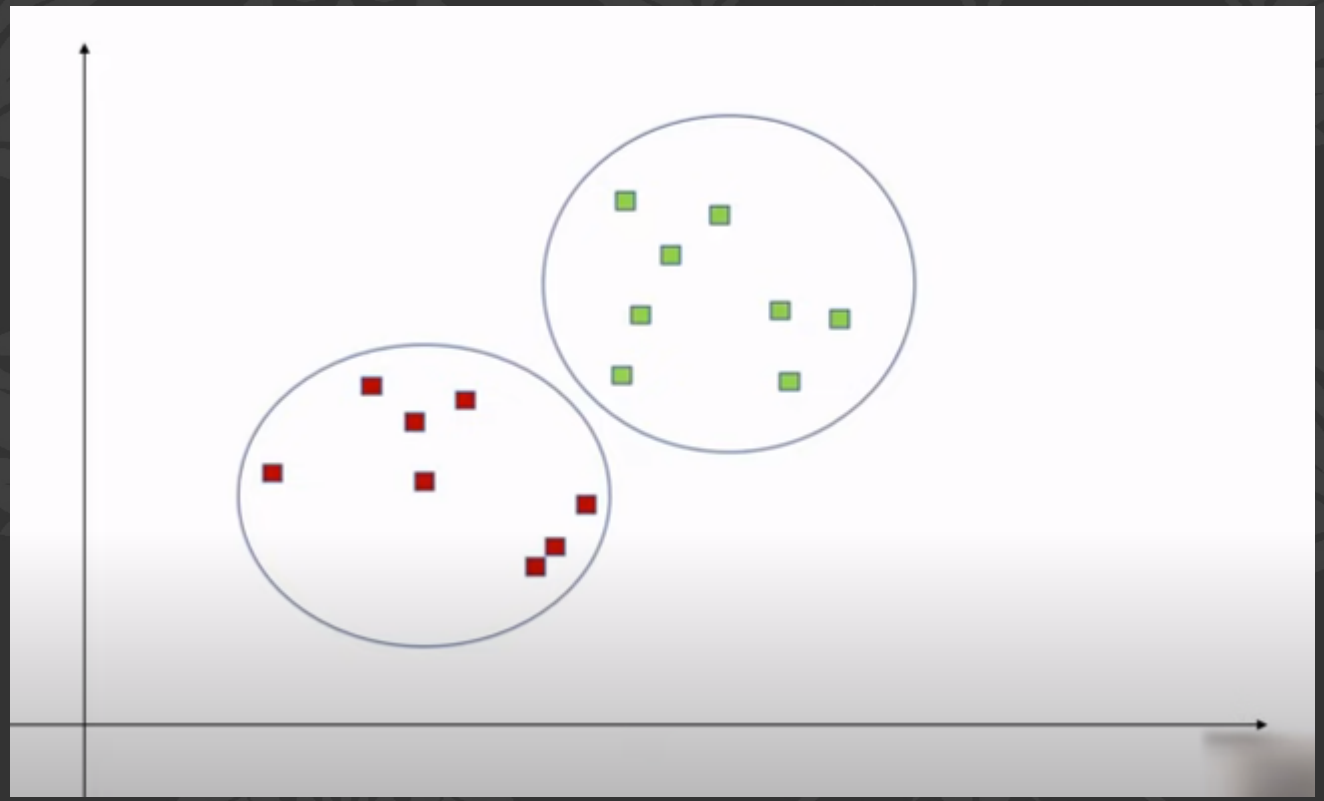
K MEANS ALGORITHM

Supervised - Target variables are known

Unsupervised - set of features are known, target or label are not known

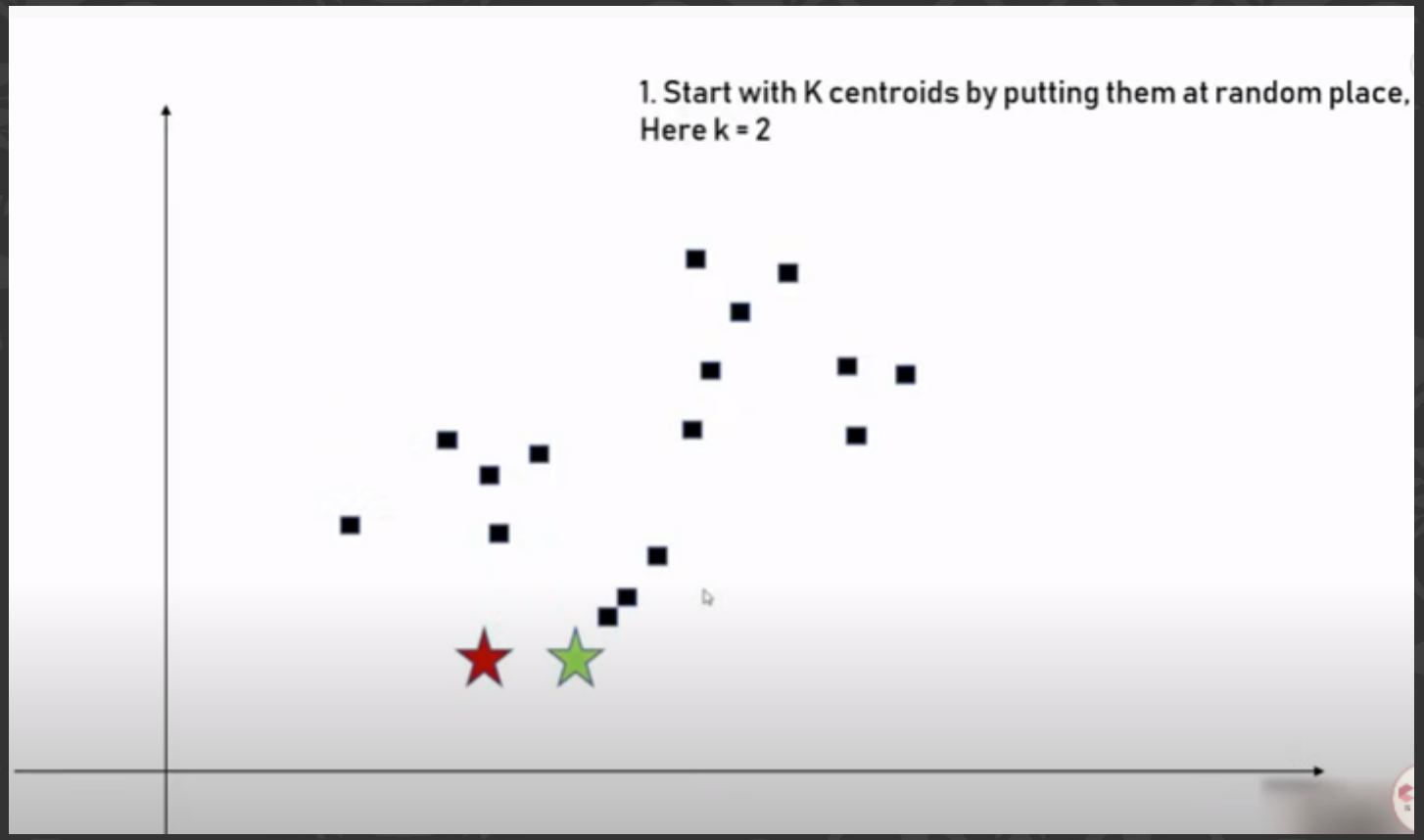
Clustering – find clusters of data



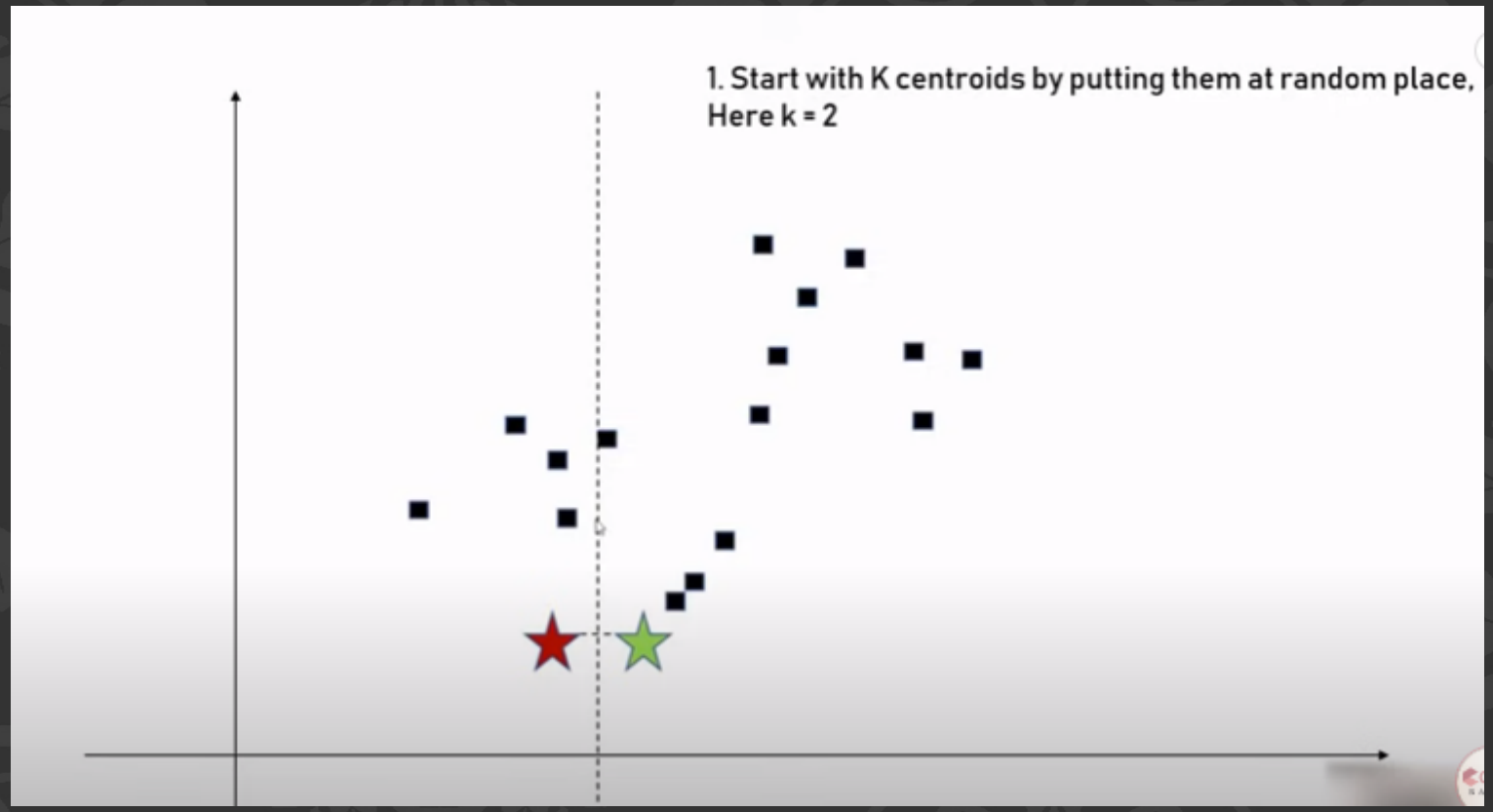


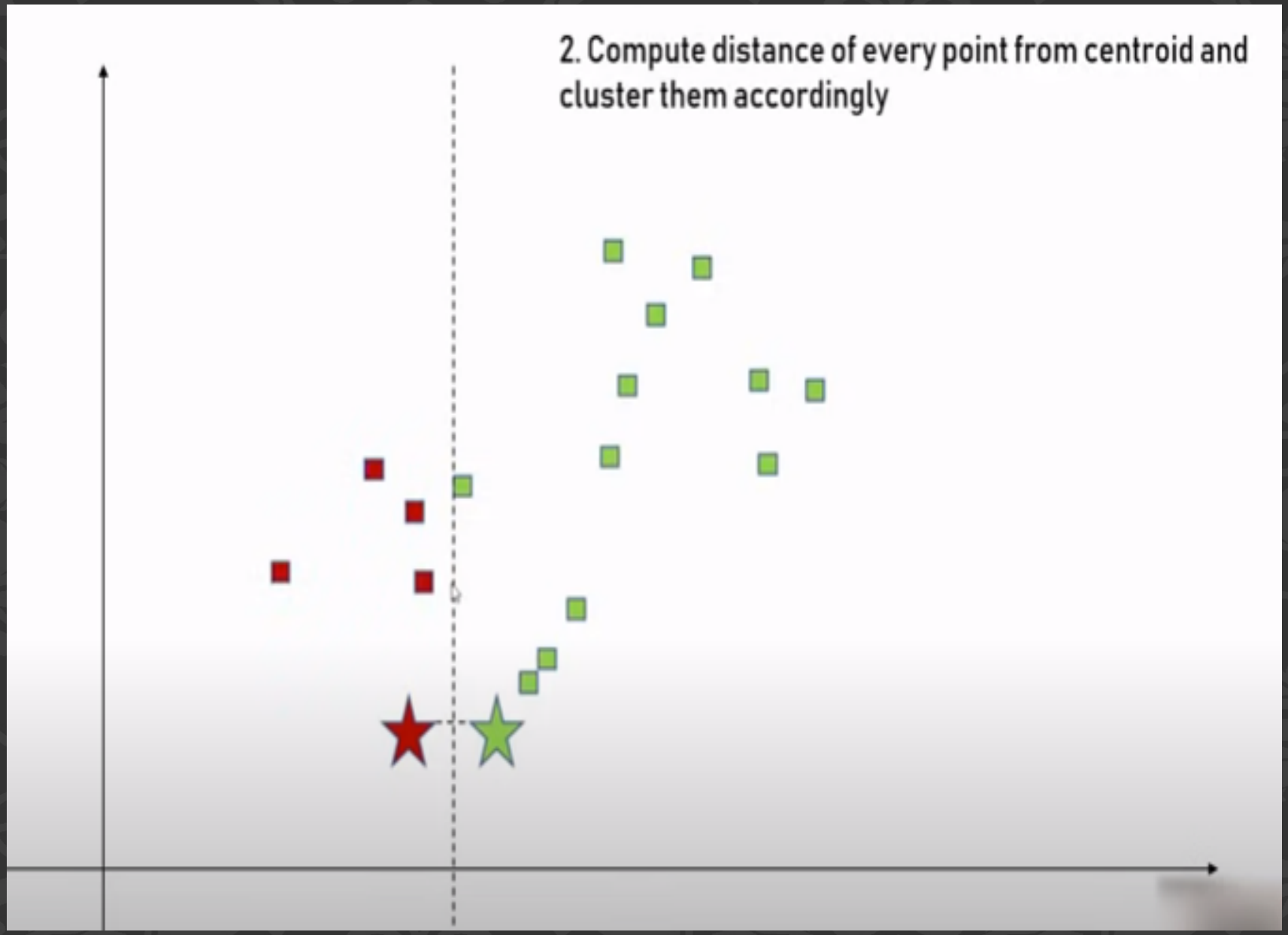
K – means – free parameter, here K =2

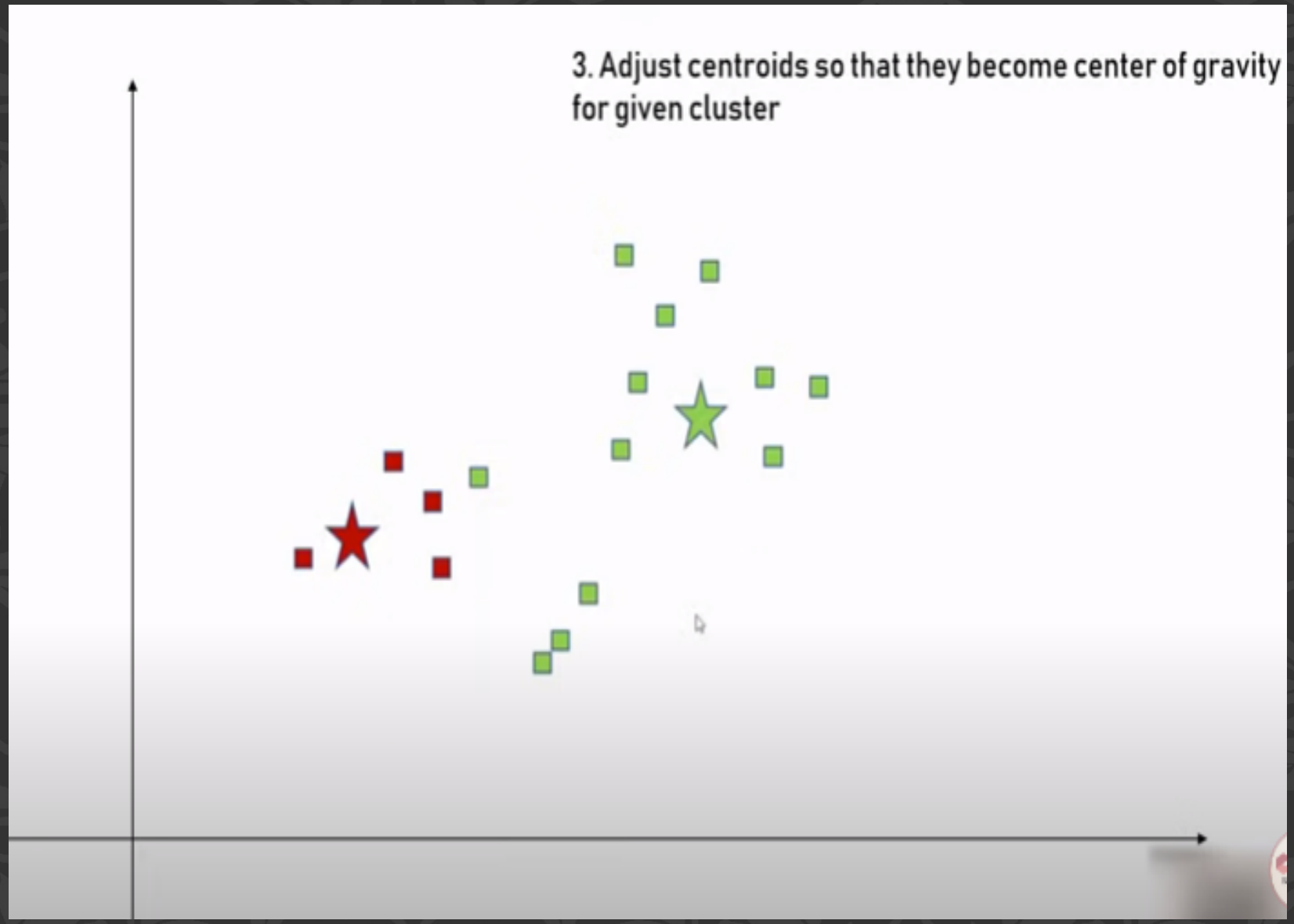
Step 1: Identify two random points

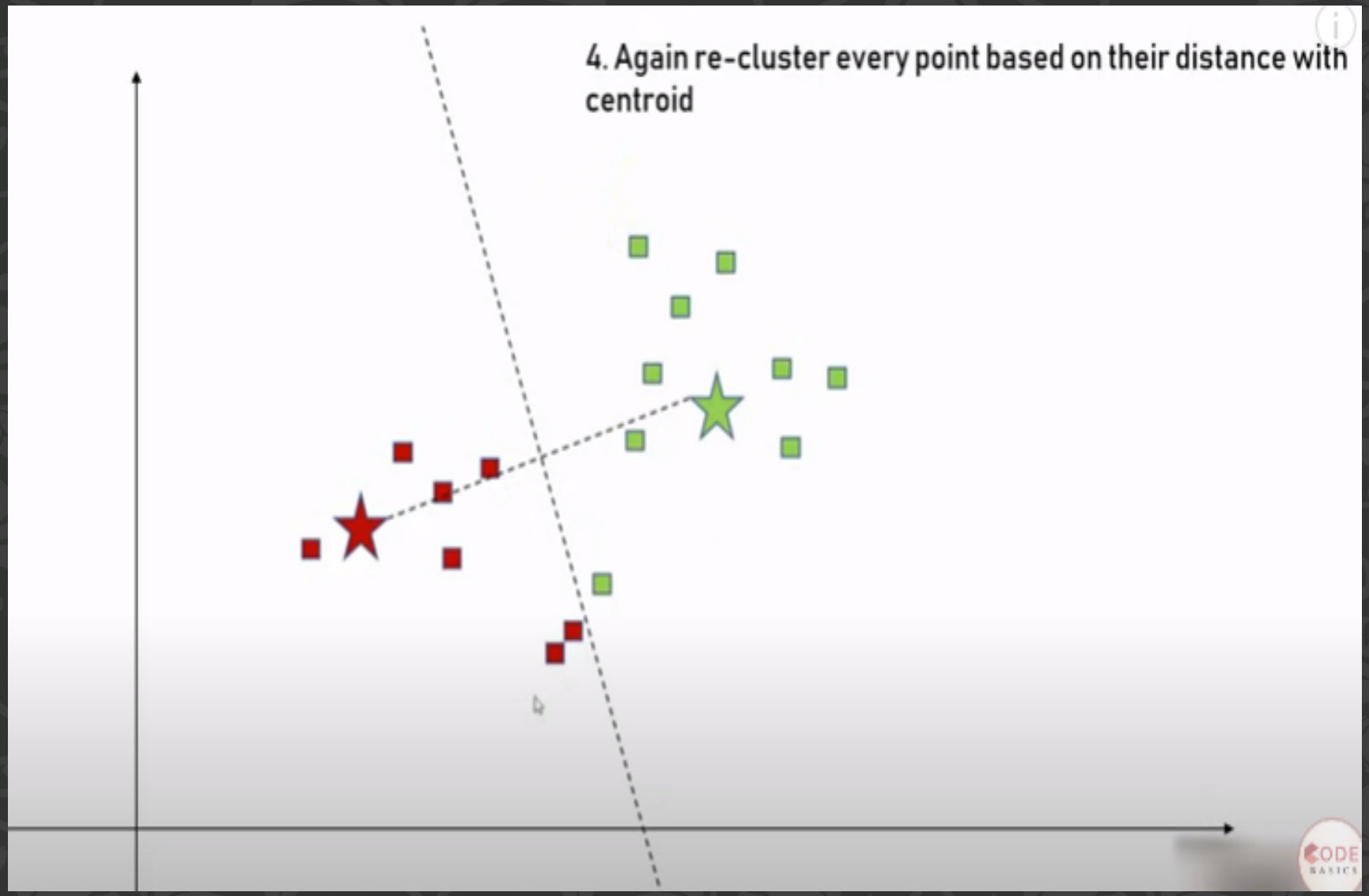


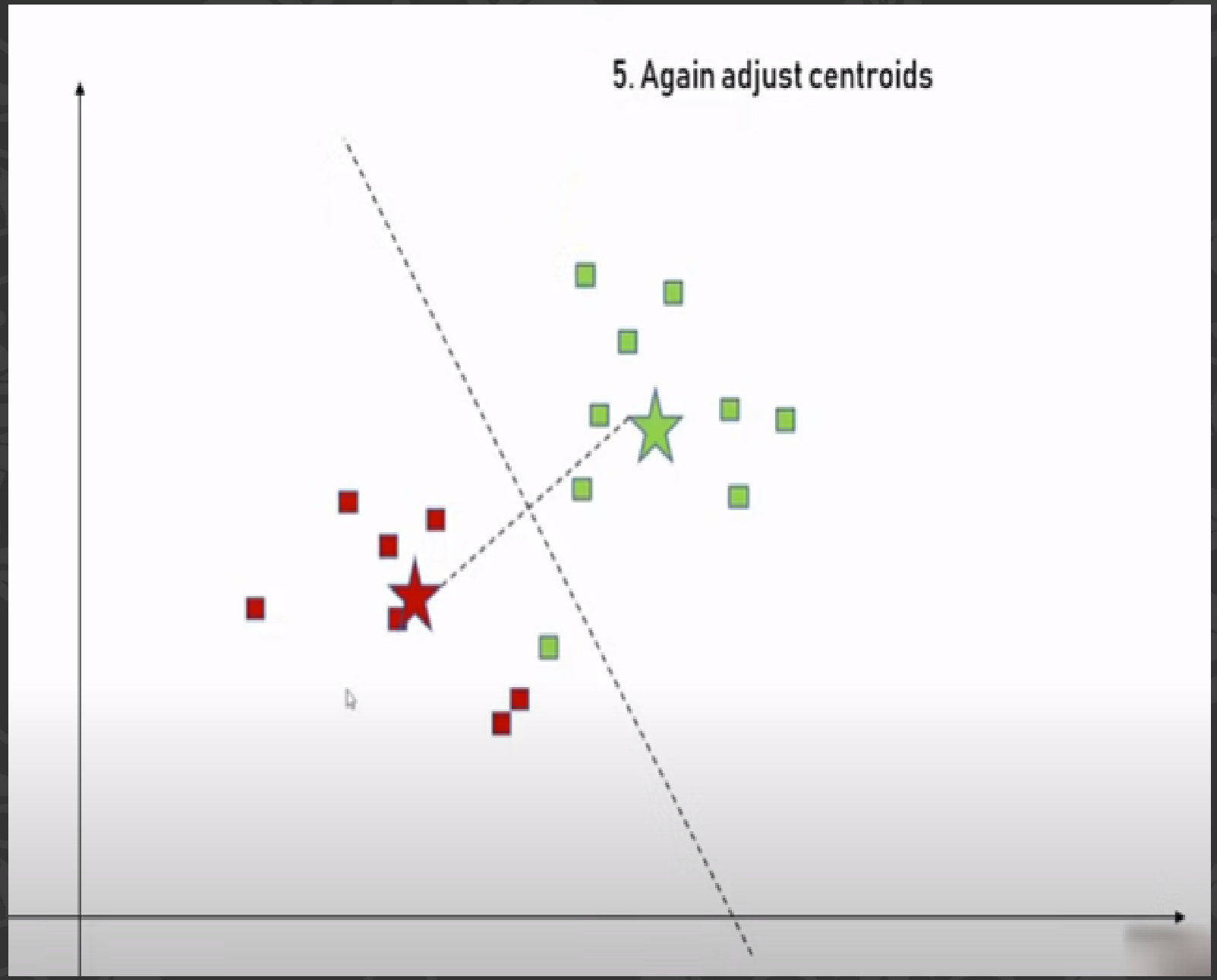
Step 2: Identify the distance each data point from the centroid

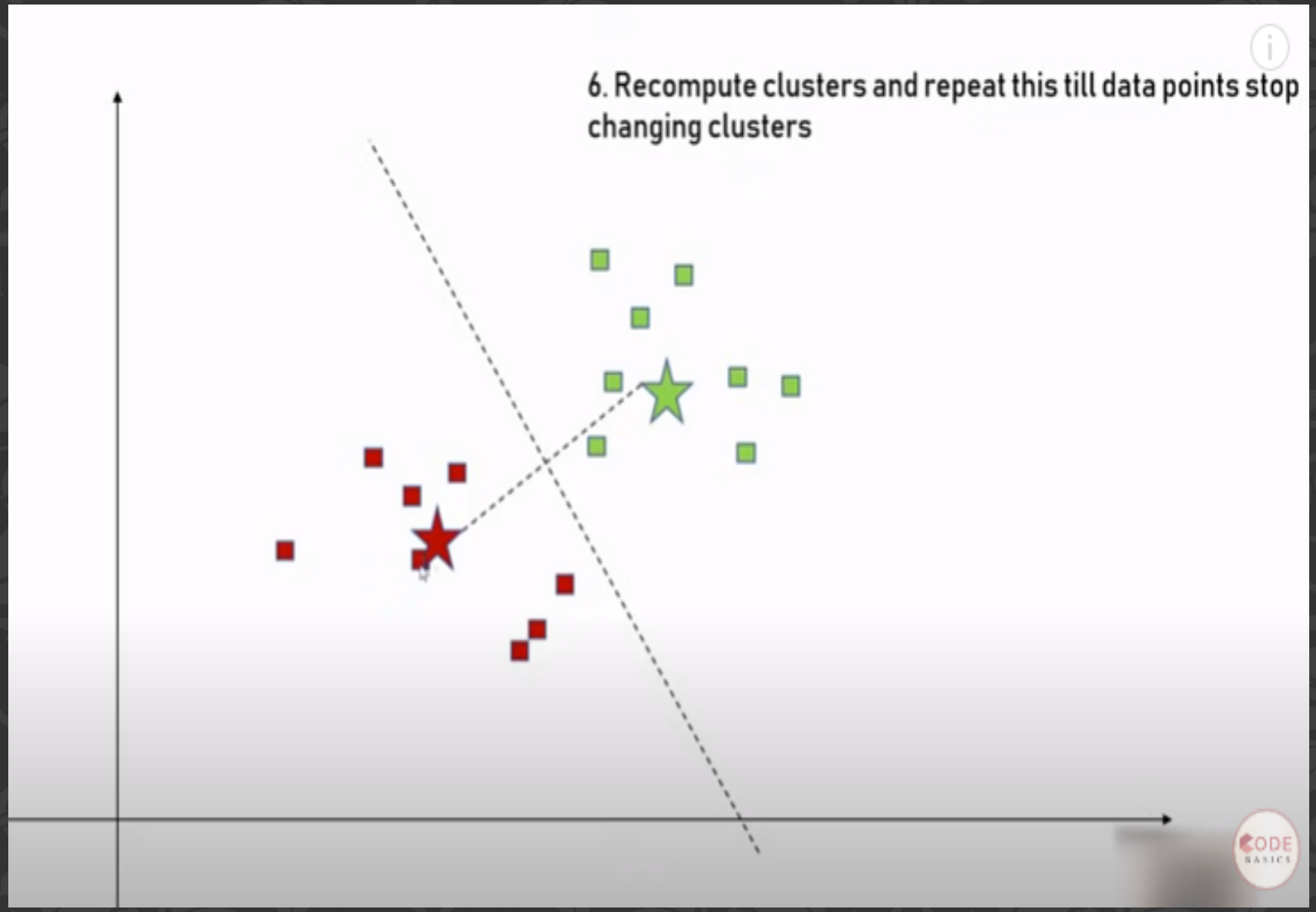


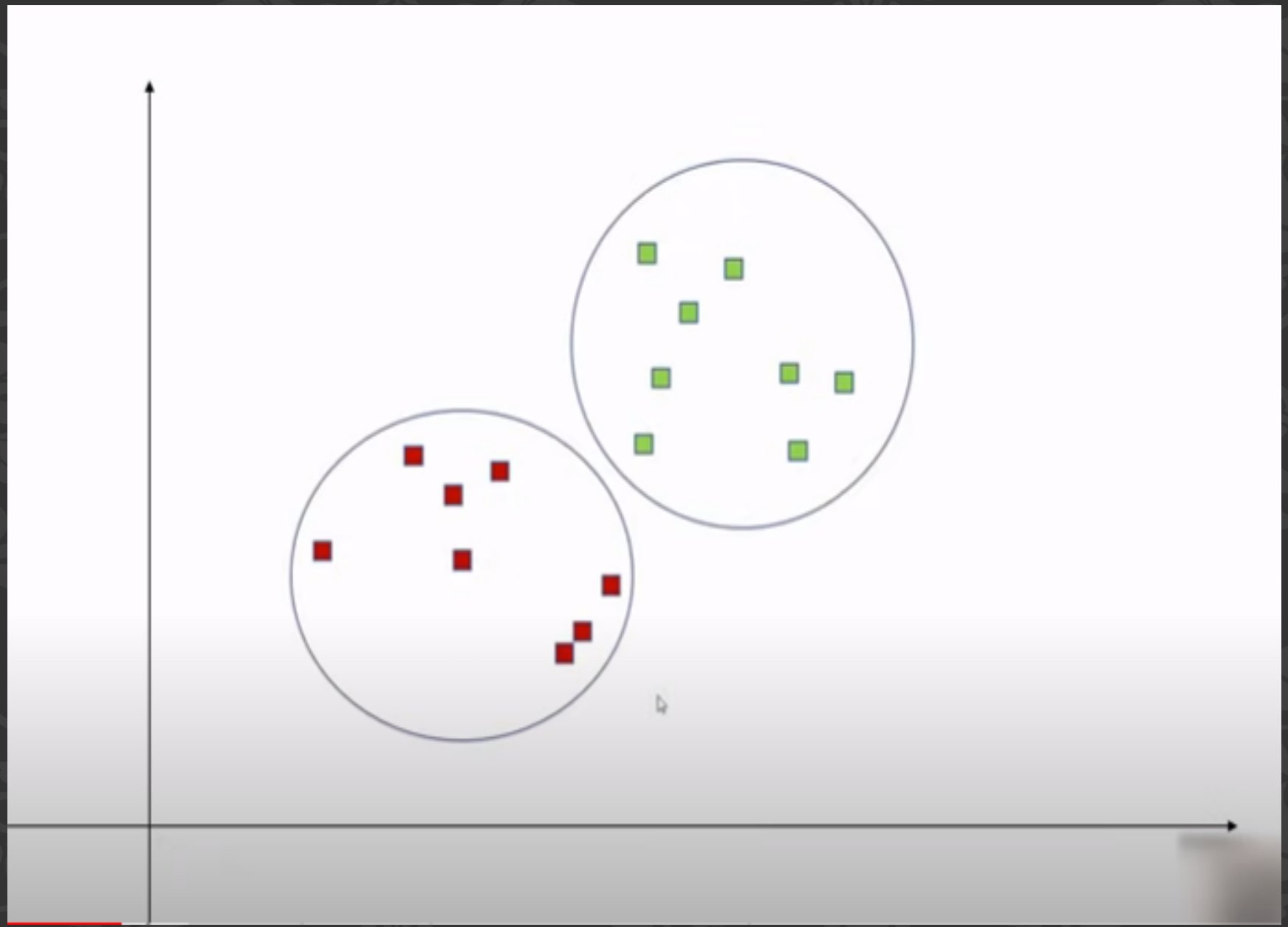

2. Compute distance of every point from centroid and cluster them accordingly


3. Adjust centroids so that they become center of gravity for given cluster

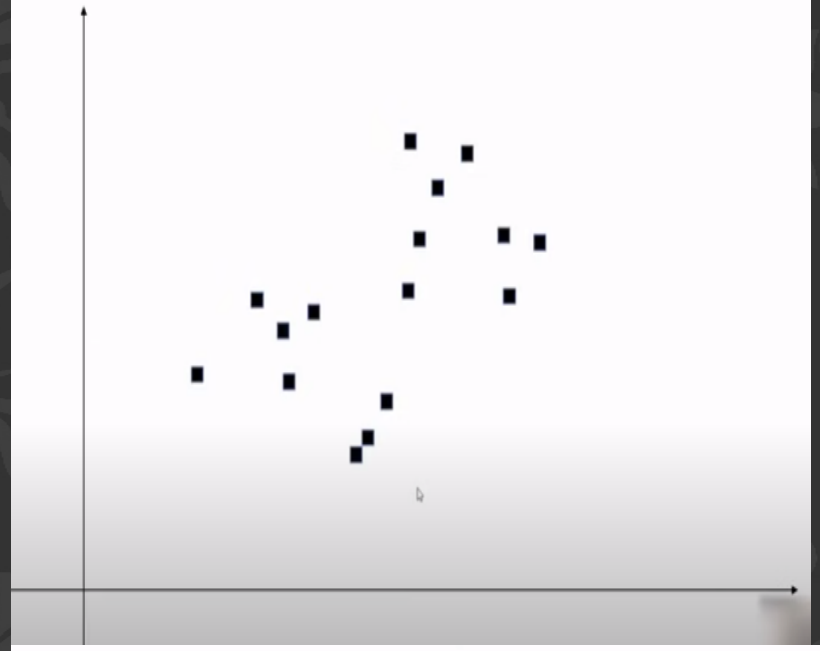

4. Again re-cluster every point based on their distance with centroid

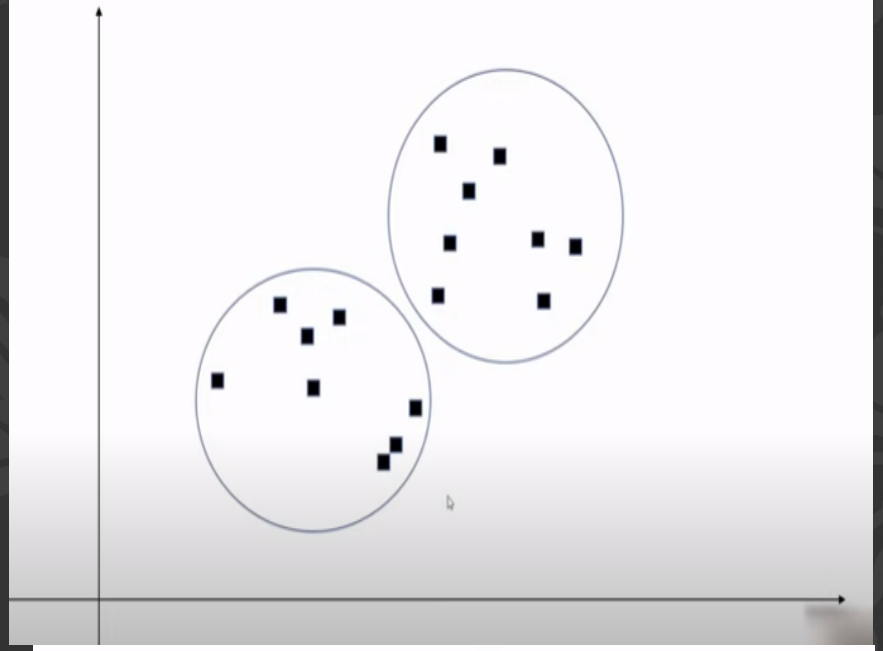

5. Again adjust centroids

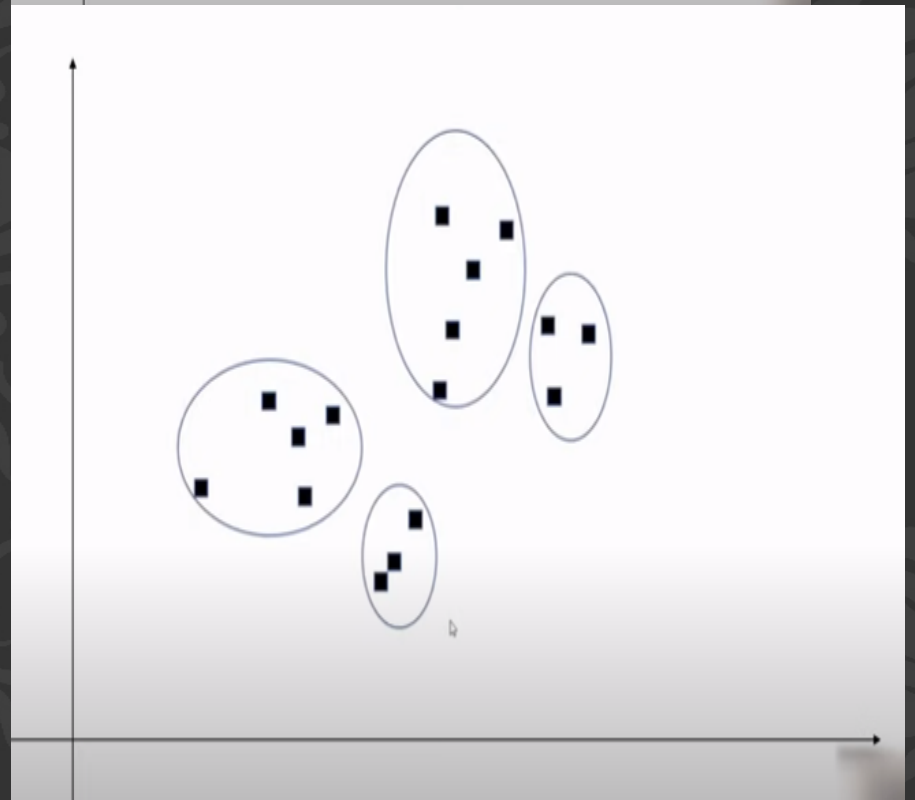

6. Recompute clusters and repeat this till data points stop changing clusters



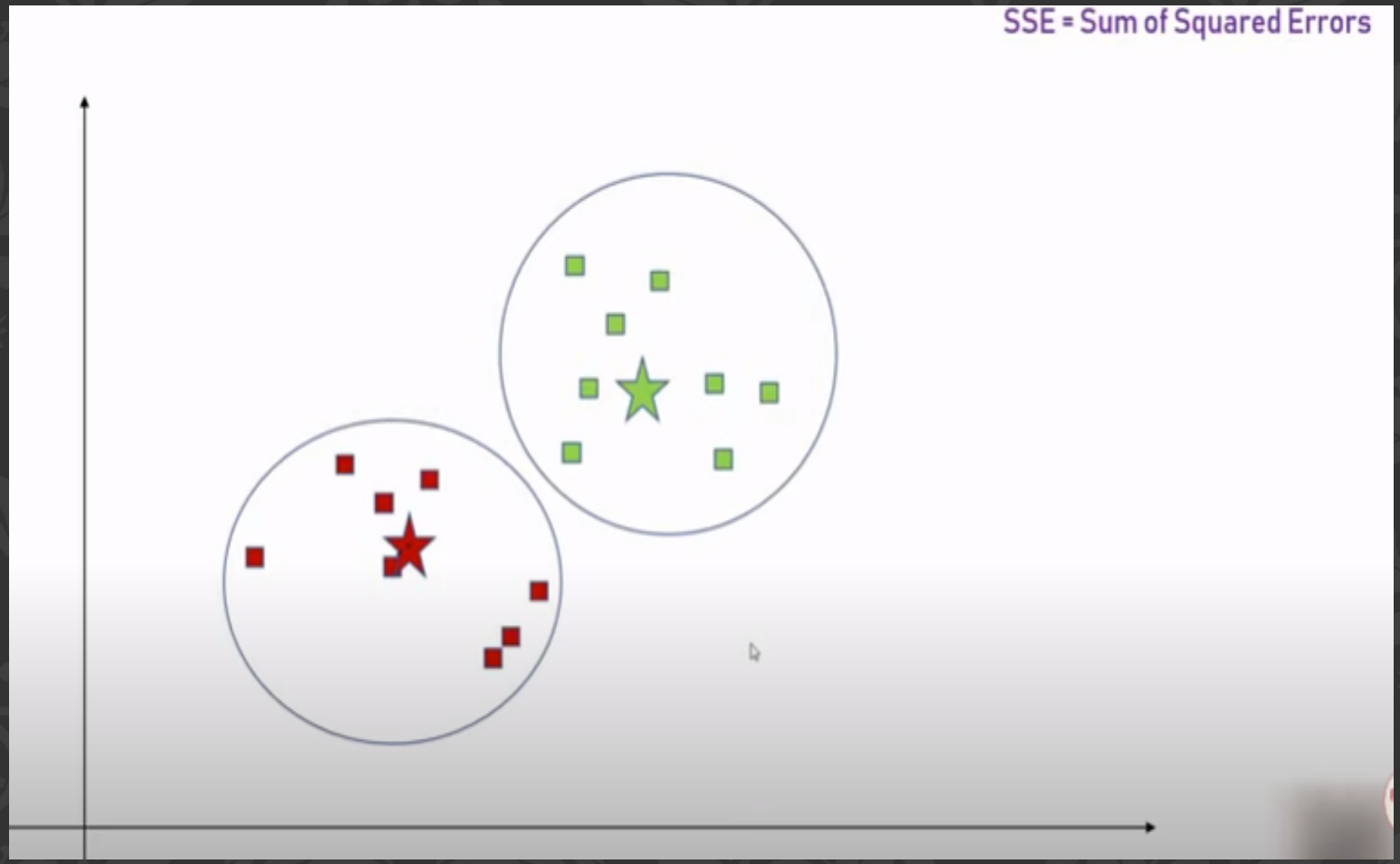
How to determine correct number of clusters (k)?



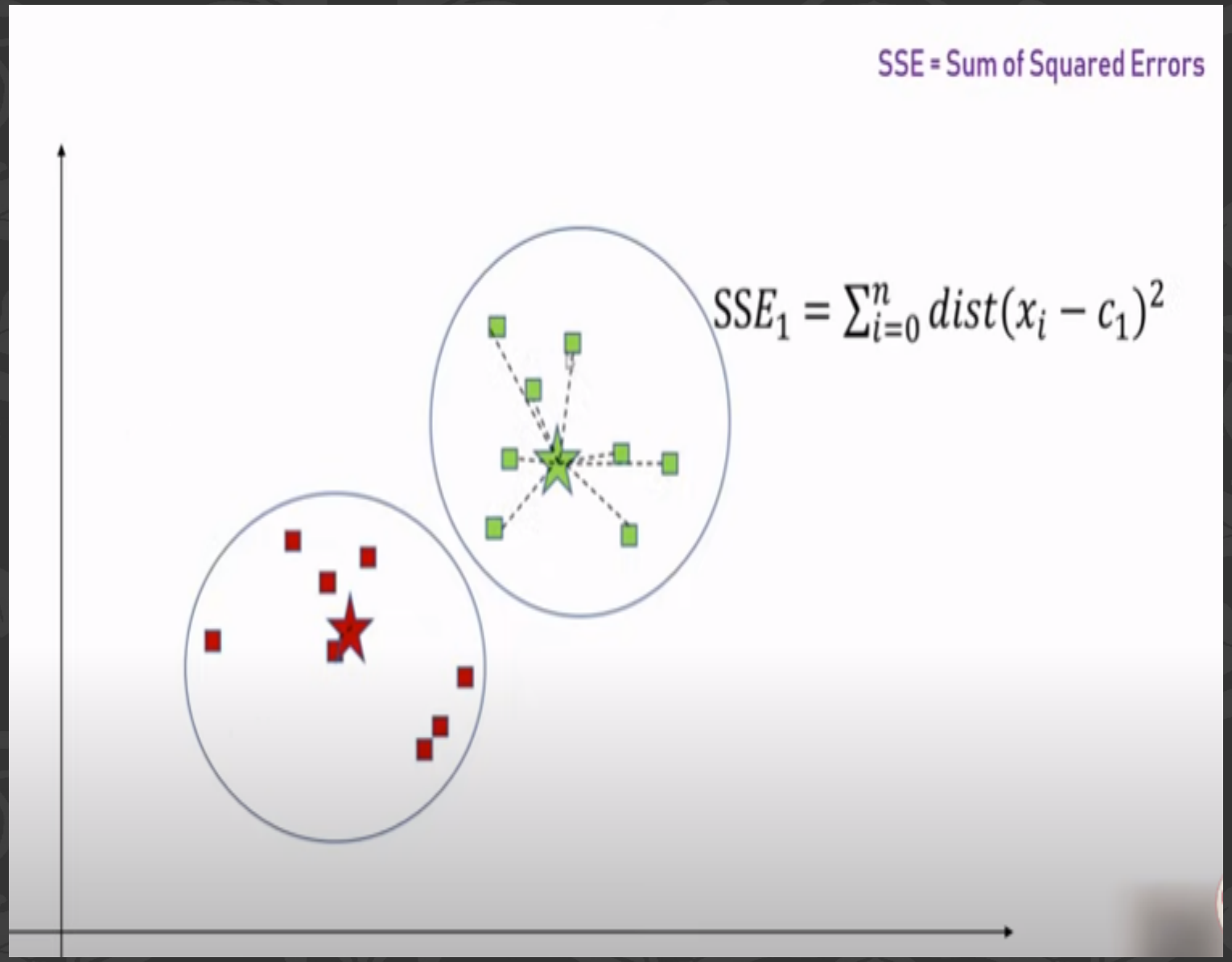




WHAT IS BEST VALUE OF K – TO START WITH TAKE K-=2

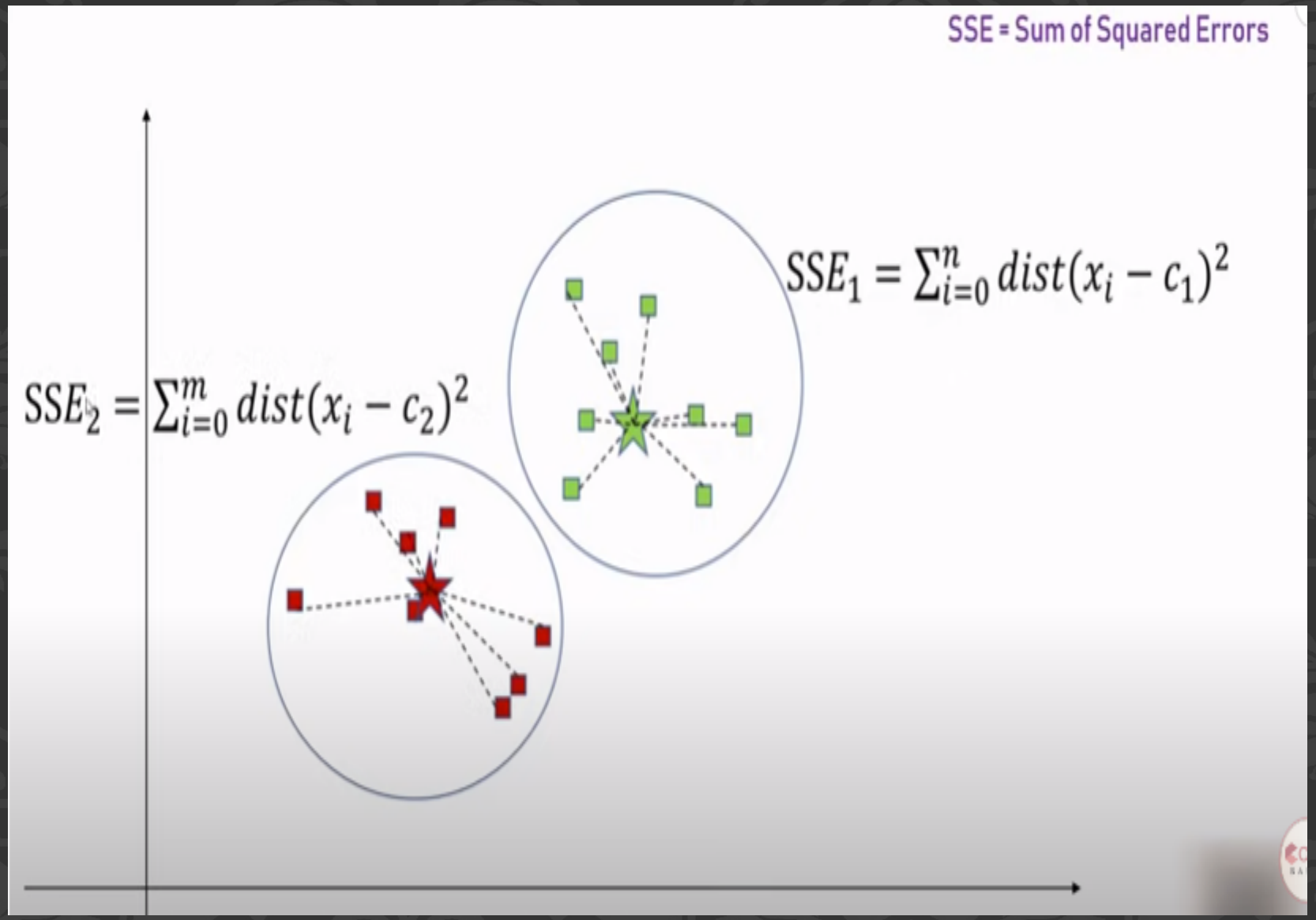


SSE = Sum of Squared Errors

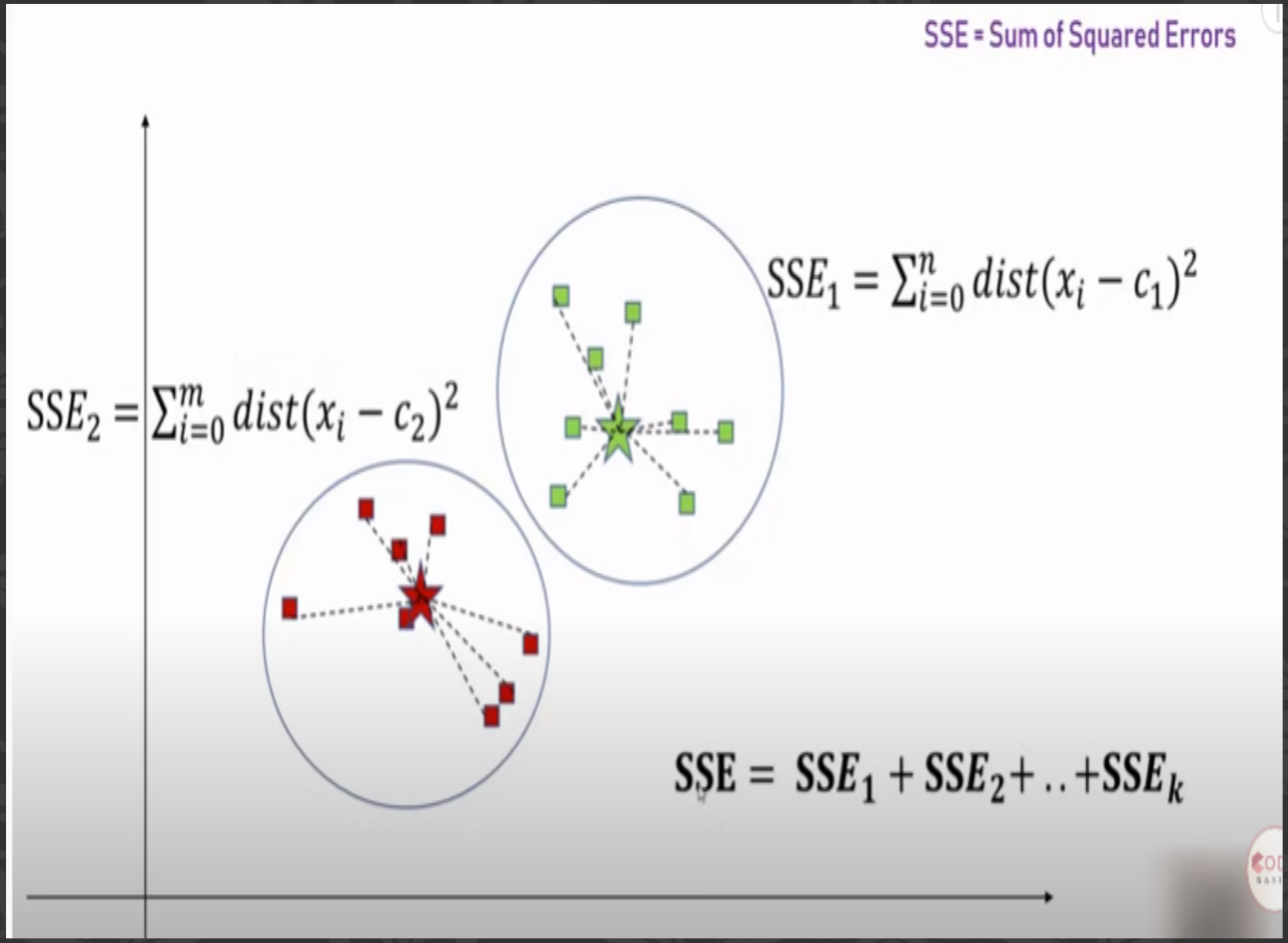


$$SSE_1 = \sum_{i=0}^{n} dist(x_i - c_1)^2$$

SSE = Sum of Squared Errors



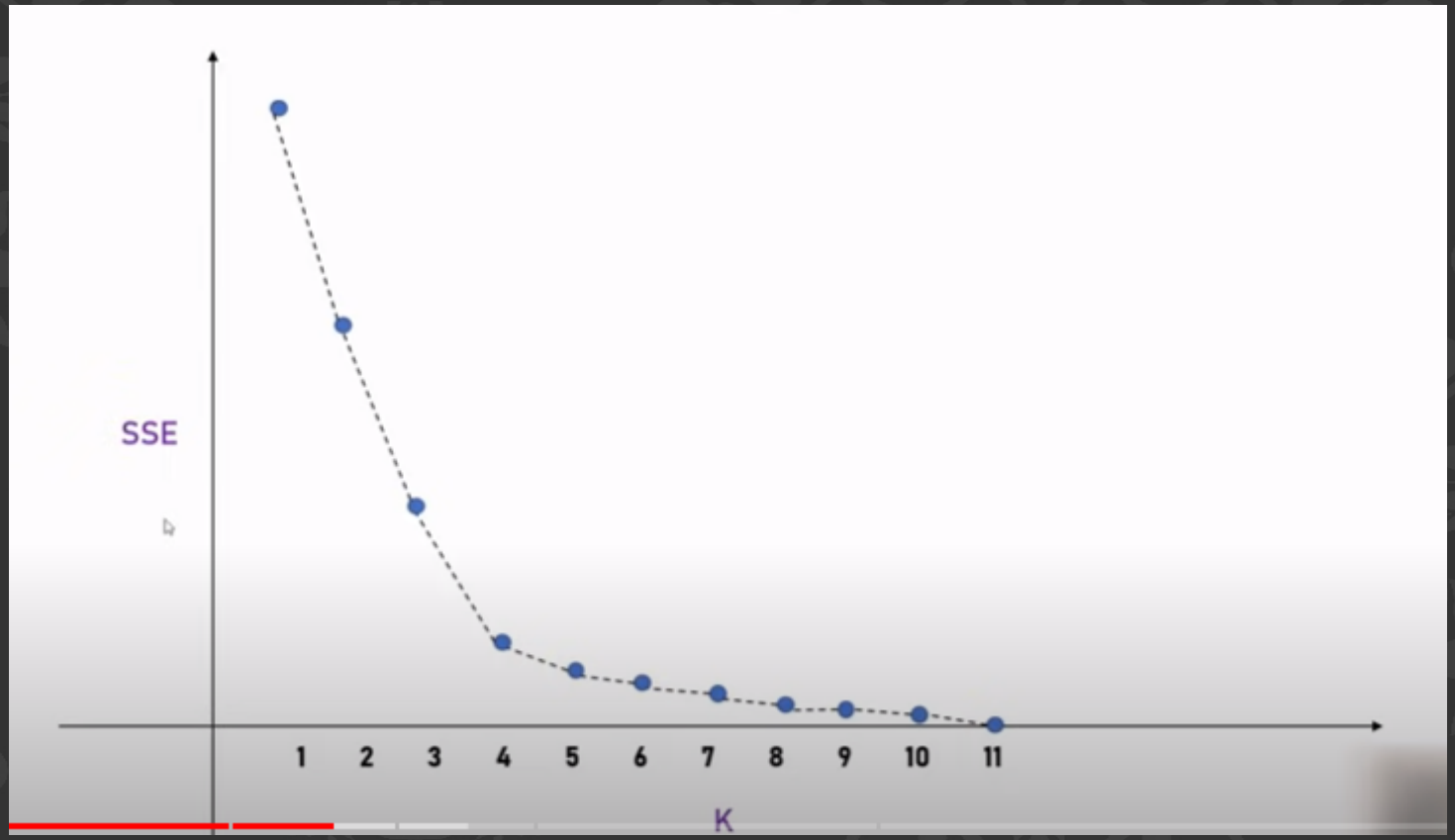
SSE = Sum of Squared Errors



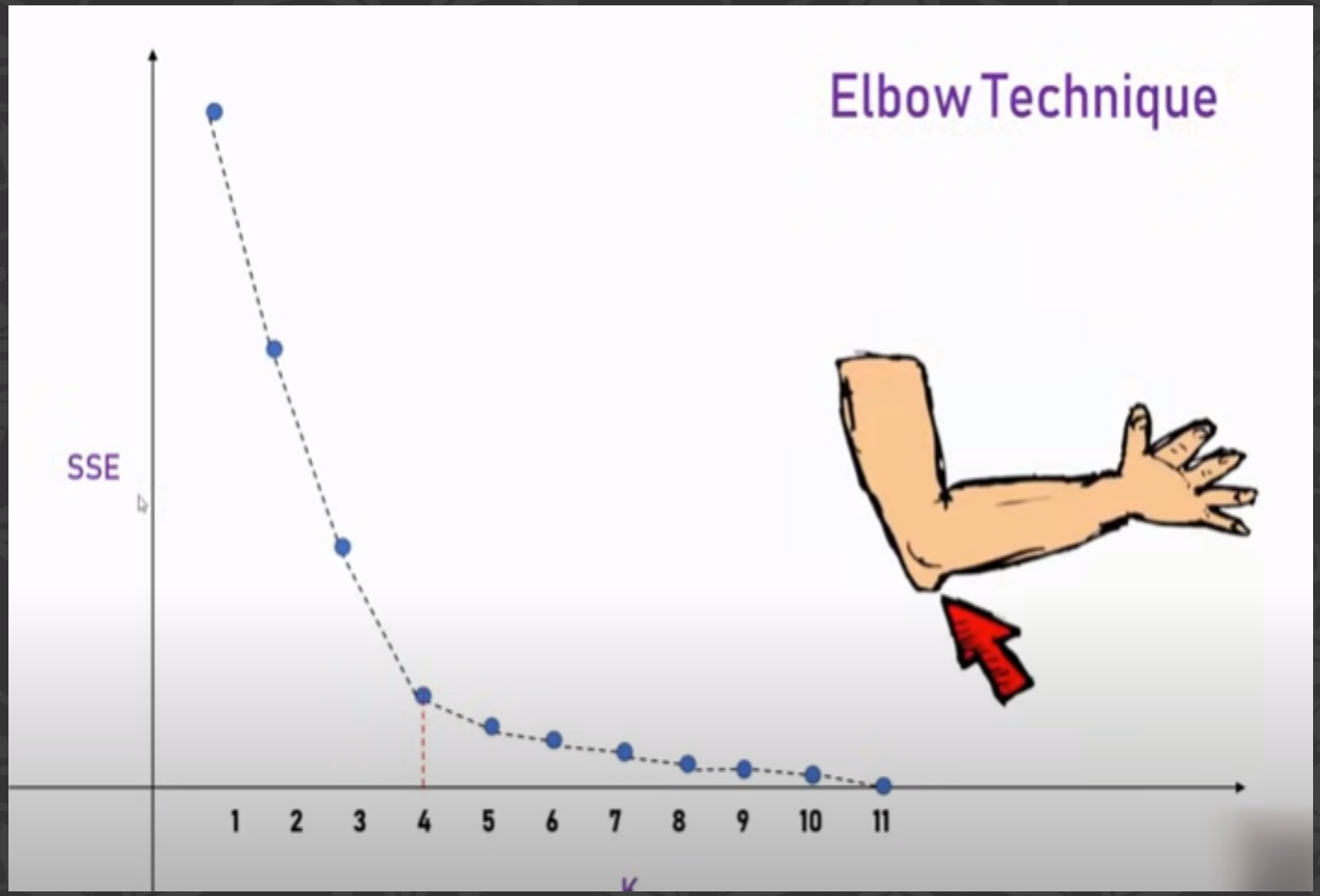
$$SSE_1 = \sum_{i=0}^{n} dist(x_i - c_1)^2$$

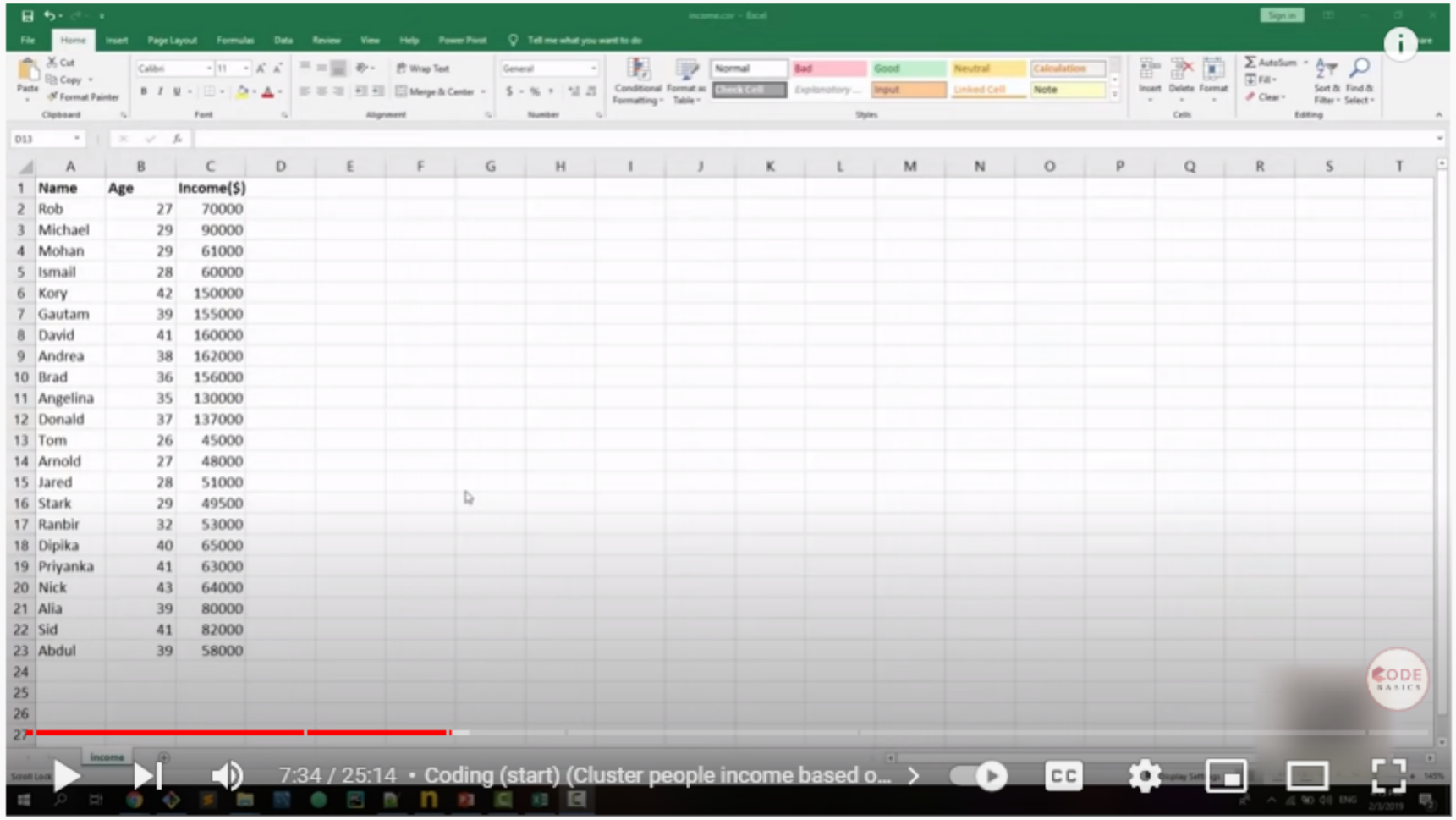
$$SSE_2 = \sum_{i=0}^{m} dist(x_i - c_2)^2$$

$$SSE = SSE_1 + SSE_2 + .. + SSE_k$$

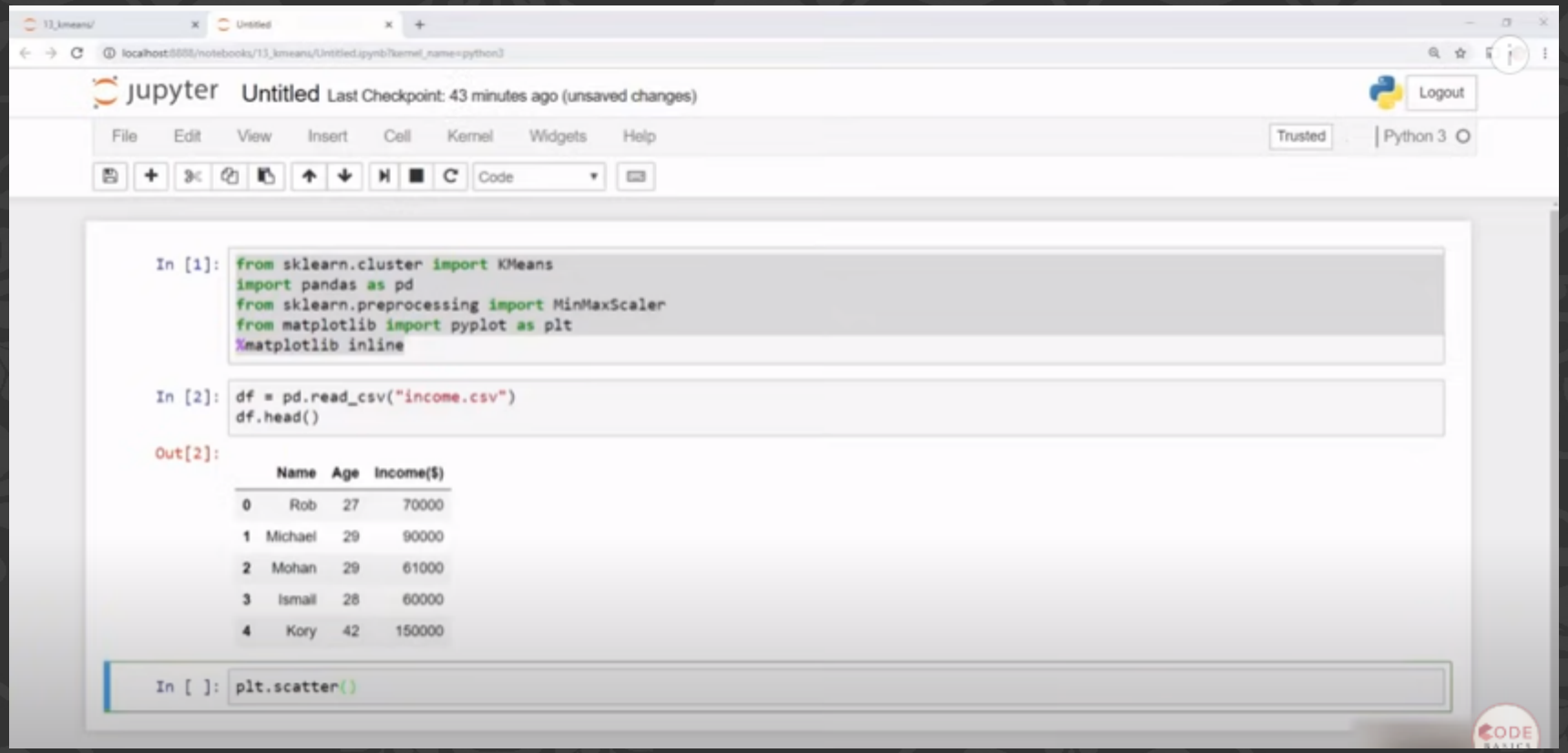

SSE
1
2
3
4
5
6
7
8
9
10
11
K

Elbow Technique





	Name	Age	Income($)
2	Rob	27	70000
3	Michael	29	90000
4	Mohan	29	61000
5	Ismail	28	60000
6	Kory	42	150000
7	Gautam	39	155000
8	David	41	160000
9	Andrea	38	162000
10	Brad	36	156000
11	Angelina	35	130000
12	Donald	37	137000
13	Tom	26	45000
14	Arnold	27	48000
15	Jared	28	51000
16	Stark	29	49500
17	Ranbir	32	53000
18	Dipika	40	65000
19	Priyanka	41	63000
20	Nick	43	64000
21	Alia	39	80000
22	Sid	41	82000
23	Abdul	39	58000


jupyter
Untitled Last Checkpoint: 43 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Code
In [1]:
from sklearn.cluster import KMeans
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from matplotlib import pyplot as plt
%matplotlib inline
In [2]:
df = pd.read_csv("income.csv")
df.head()
Out[2]:
Name Age Income($)
0 Rob 27 70000
1 Michael 29 90000
2 Mohan 29 61000
3 Ismail 28 60000
4 Kory 42 150000
In [ ]:
plt.scatter()

jupyter Untitled Last Checkpoint: 43 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

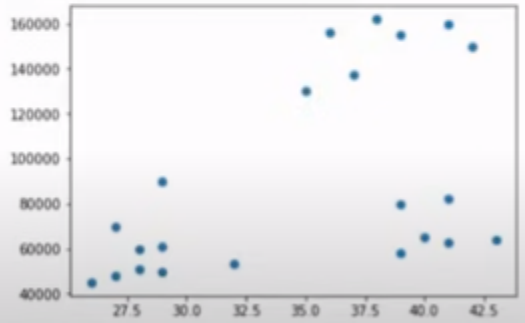
Trusted Python 3

Code

2	Mohan	29	61000
3	Ismail	28	60000
4	Kory	42	150000

```
In [3]: plt.scatter(df['Age'],df['Income($)'])
```

```
Out[3]: <matplotlib.collections.PathCollection at 0x2651e01c978>
```



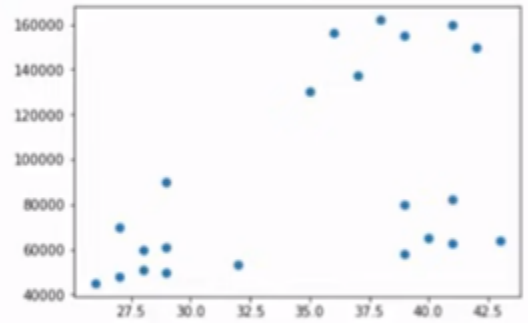
jupyter Untitled Last Checkpoint: an hour ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Code

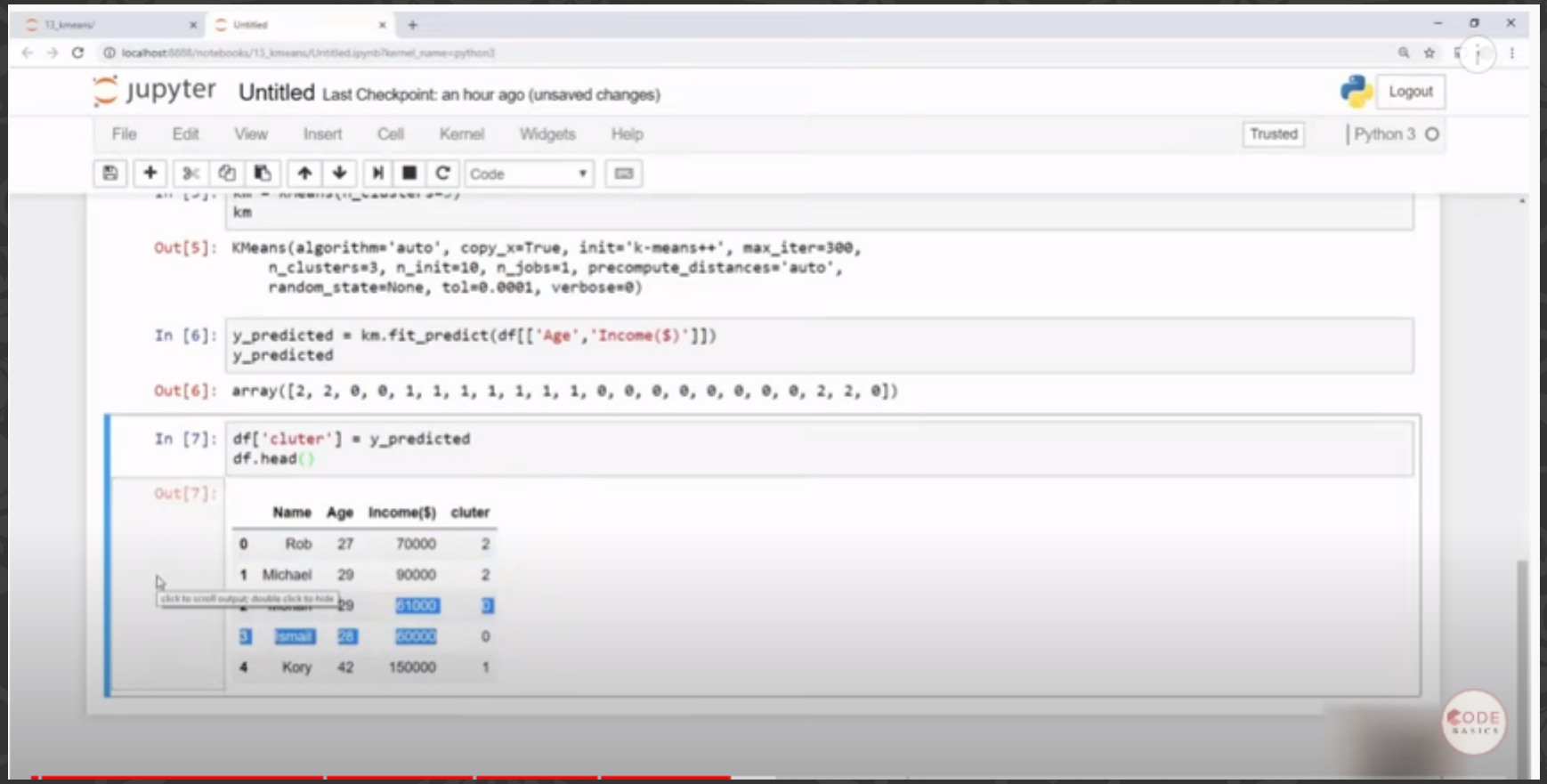


In [5]:
```
km = KMeans(n_clusters=3)
km
```

Out[5]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=3, n_init=10, n_jobs=1, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=0)
```

In [6]:
```
y_predicted = km.fit_predict(df[['Age','Income($)']])
y_predicted
```

Out[6]:
```
array([2, 2, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0])
```


jupyter Untitled Last Checkpoint: an hour ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Code
km
Out[5]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
n_clusters=3, n_init=10, n_jobs=1, precompute_distances='auto',
random_state=None, tol=0.0001, verbose=0)
In [6]: y_predicted = km.fit_predict(df[['Age','Income($)']])
y_predicted
Out[6]: array([2, 2, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0])
In [7]: df['cluter'] = y_predicted
df.head()
Out[7]:
Name Age Income($) cluter
0 Rob 27 70000 2
1 Michael 29 90000 2
3 Ismail 28 60000 0
4 Kory 42 150000 1

jupyter Untitled Last Checkpoint: an hour ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

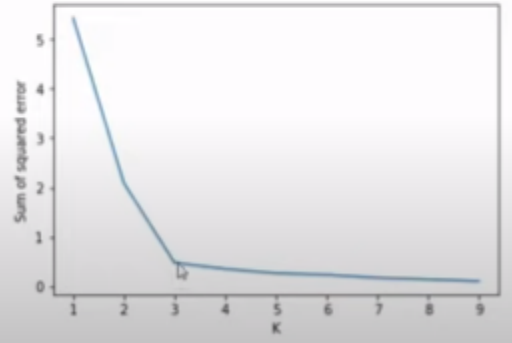
Trusted Python 3

Code

```
0.23022348815682736,
0.16869711728567782,
0.13706529523670863,
0.10188787724979426]
```

In [22]:
```
plt.xlabel('K')
plt.ylabel('Sum of squared error')
plt.plot(k_rng,sse)
```

Out[22]:
```
[<matplotlib.lines.Line2D at 0x26520242438>]
```



Weaknesses of K-Mean Clustering

1. When the numbers of data are not so many, initial grouping will determine the cluster significantly.
2. The number of cluster, K, must be determined before hand. Its disadvantage is that it does not yield the same result with each run, since the resulting clusters depend on the initial random assignments.
3. We never know the real cluster, using the same data, because if it is inputted in a different order it may produce different cluster if the number of data is few.
4. It is sensitive to initial condition. Different initial condition may produce different result of cluster. The algorithm may be trapped in the *local optimum*.

CONCLUSION

- *K-means algorithm is* useful for undirected knowledge discovery and is relatively simple. K-means has found wide spread usage in lot of fields, ranging from unsupervised learning of neural network, Pattern recognitions, Classification analysis, Artificial intelligence, image processing, machine vision, and many others.