Building Clustering Models with scikit-learn

BUILDING A SIMPLE CLUSTERING MODEL IN SCIKIT-LEARN

Janani Ravi
CO-FOUNDER, LOONYCORN
www.loonycorn.com

Overview

Clustering as a classic ML problem

Solving clustering using unsupervised learning

Setting up the clustering problem

Solving the clustering problem using K-means clustering

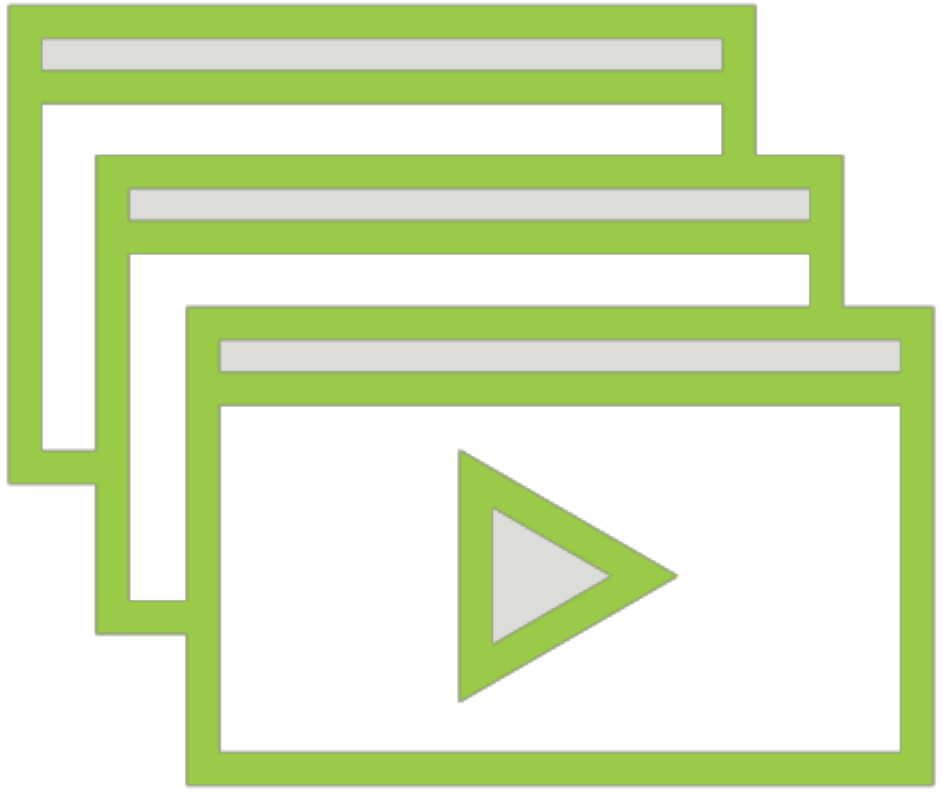
Prerequisites and Course Outline

Prerequisites

Working with Python and Python libraries

Basic understanding of machine learning algorithms

Prerequisites



Understanding Machine Learning

Understanding Machine Learning with Python

Building Your First scikit-learn Solution

Course Outline

Clustering as canonical ML problem

Implement and contrast different clustering techniques

Hyperparameter tuning in clustering

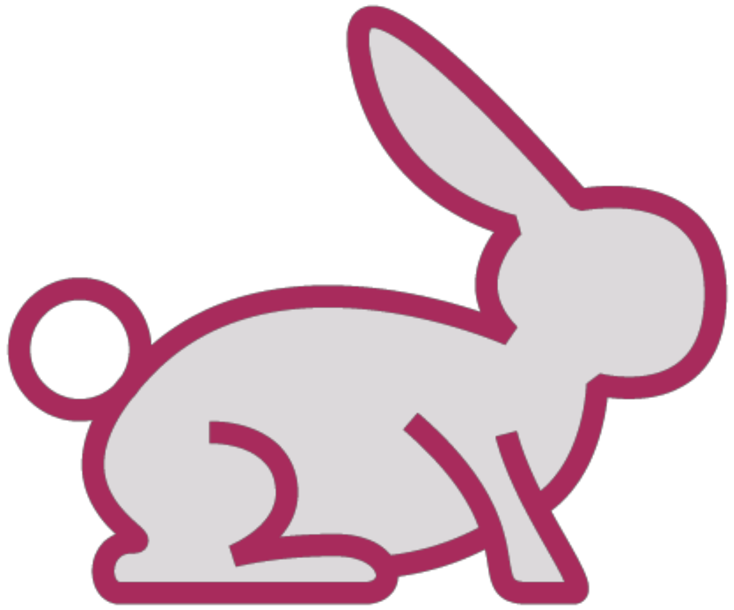
Clustering operations on image data

Supervised and Unsupervised Learning

"What lies behind us and what lies ahead of us are tiny matters compared to what lives within us"

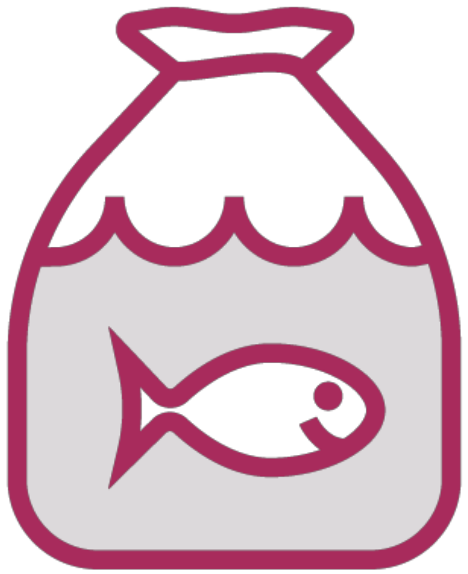
Henry David Thoreau

Whales: Fish or Mammals?



Mammals

Members of the infraorder *Cetacea*



Fish

Look like fish, swim like fish, move with fish

Whales: Fish or Mammals?

ML-based Classifier

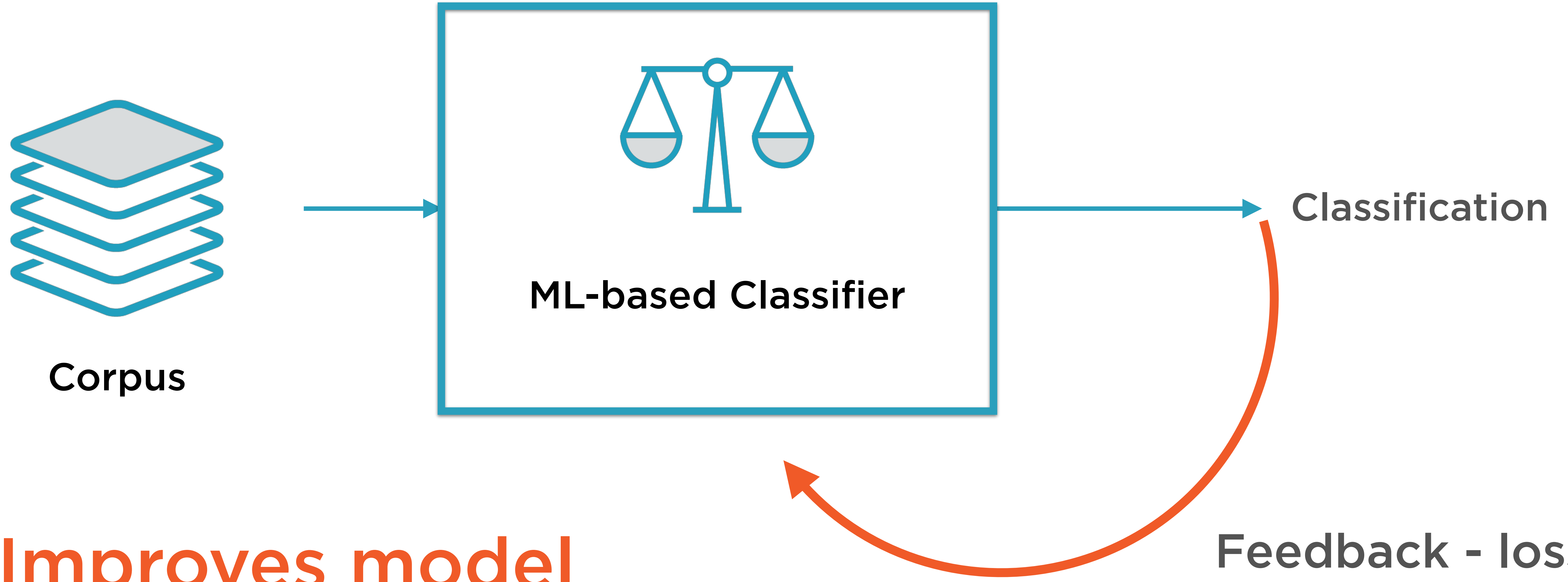
Training

Feed in a large corpus of data classified correctly

Prediction

Use it to classify new instances which it has not seen before

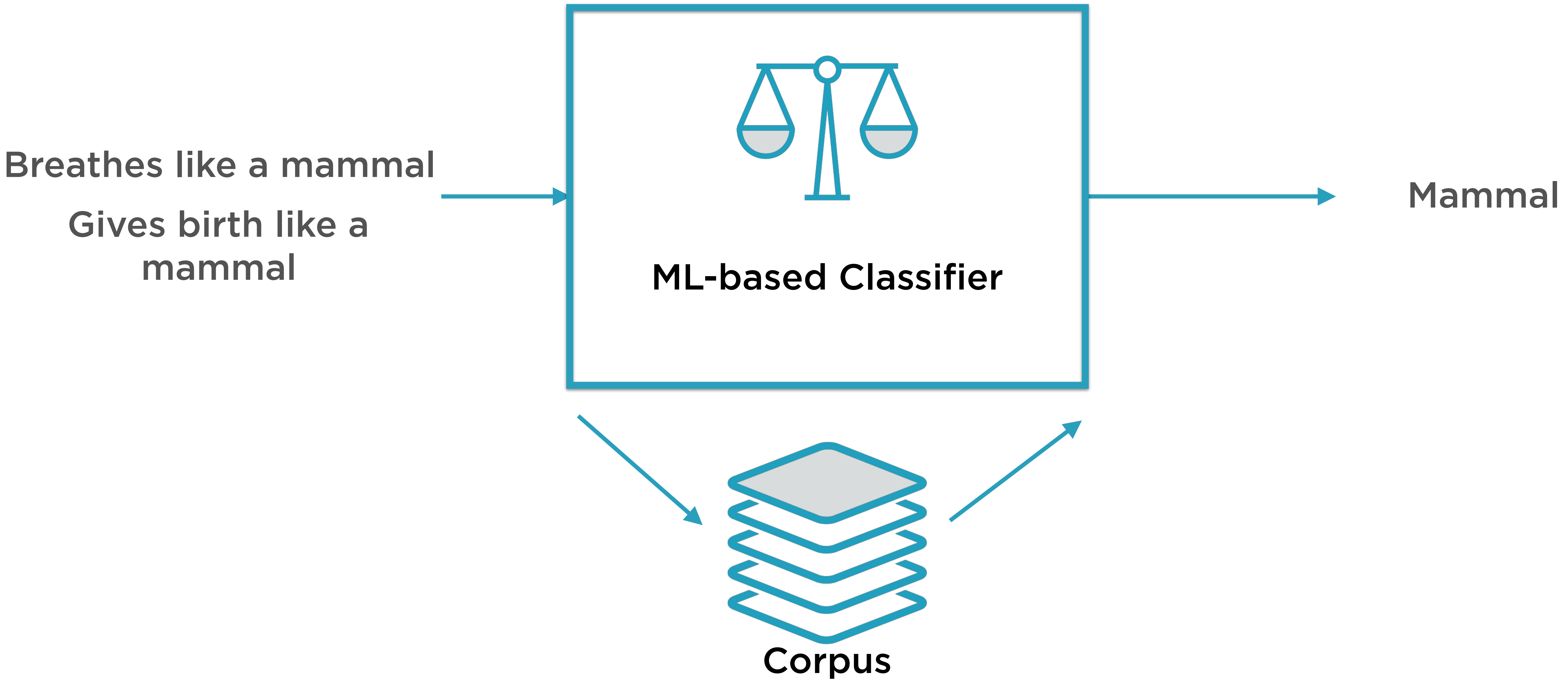
Training the ML-based Classifier



Improves model parameters

Feedback - loss function or cost function

ML-based Binary Classifier

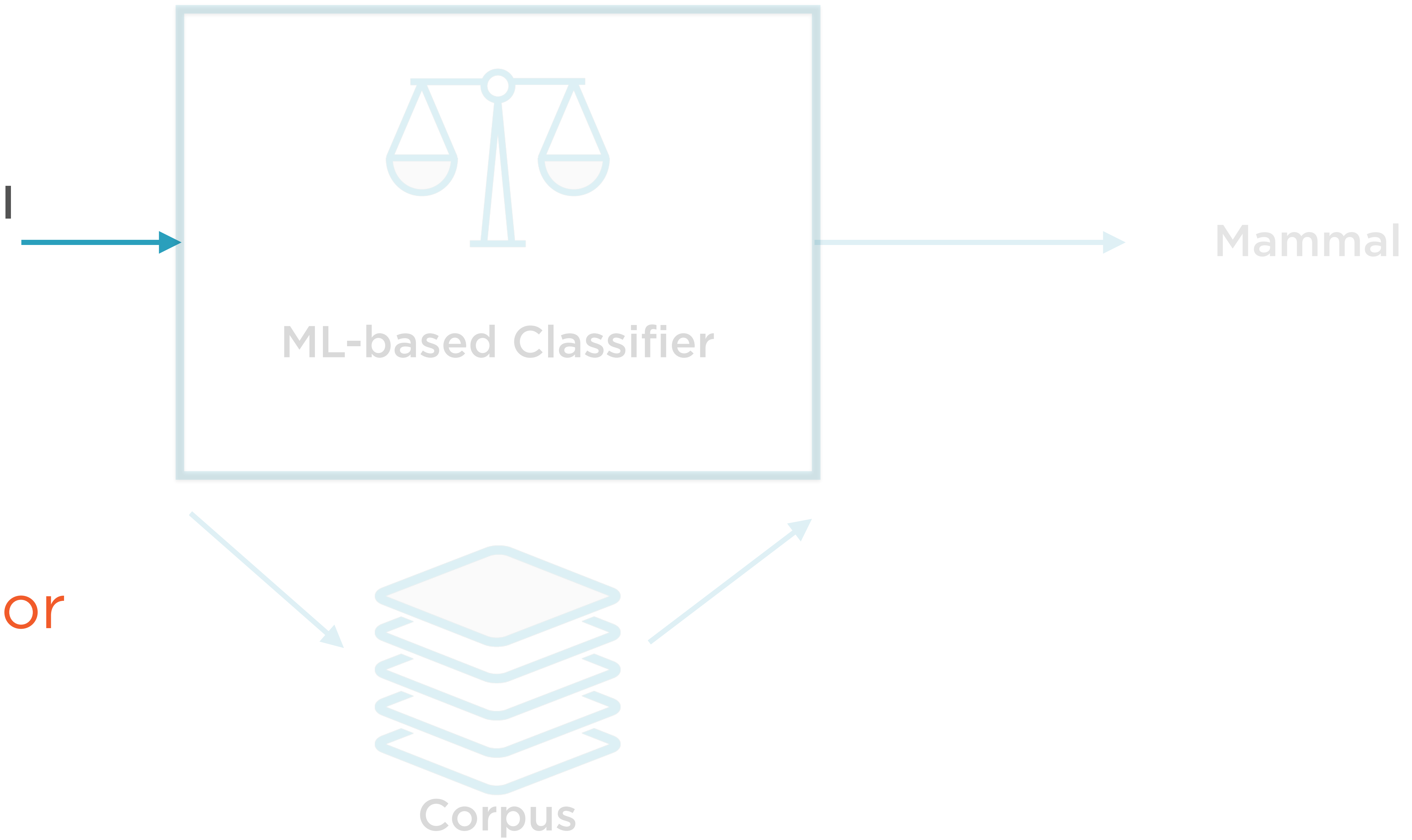


ML-based Binary Classifier

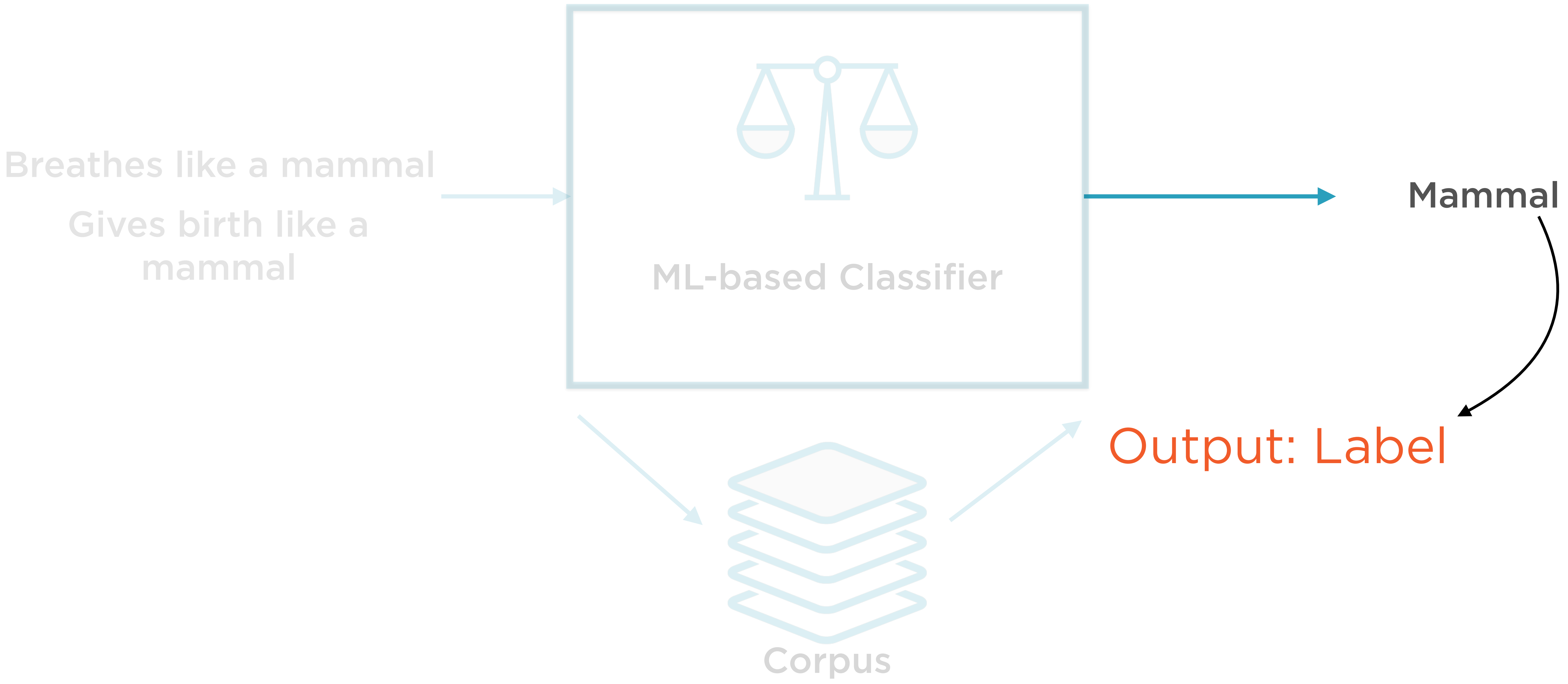
Breathes like a mammal

Gives birth like a mammal

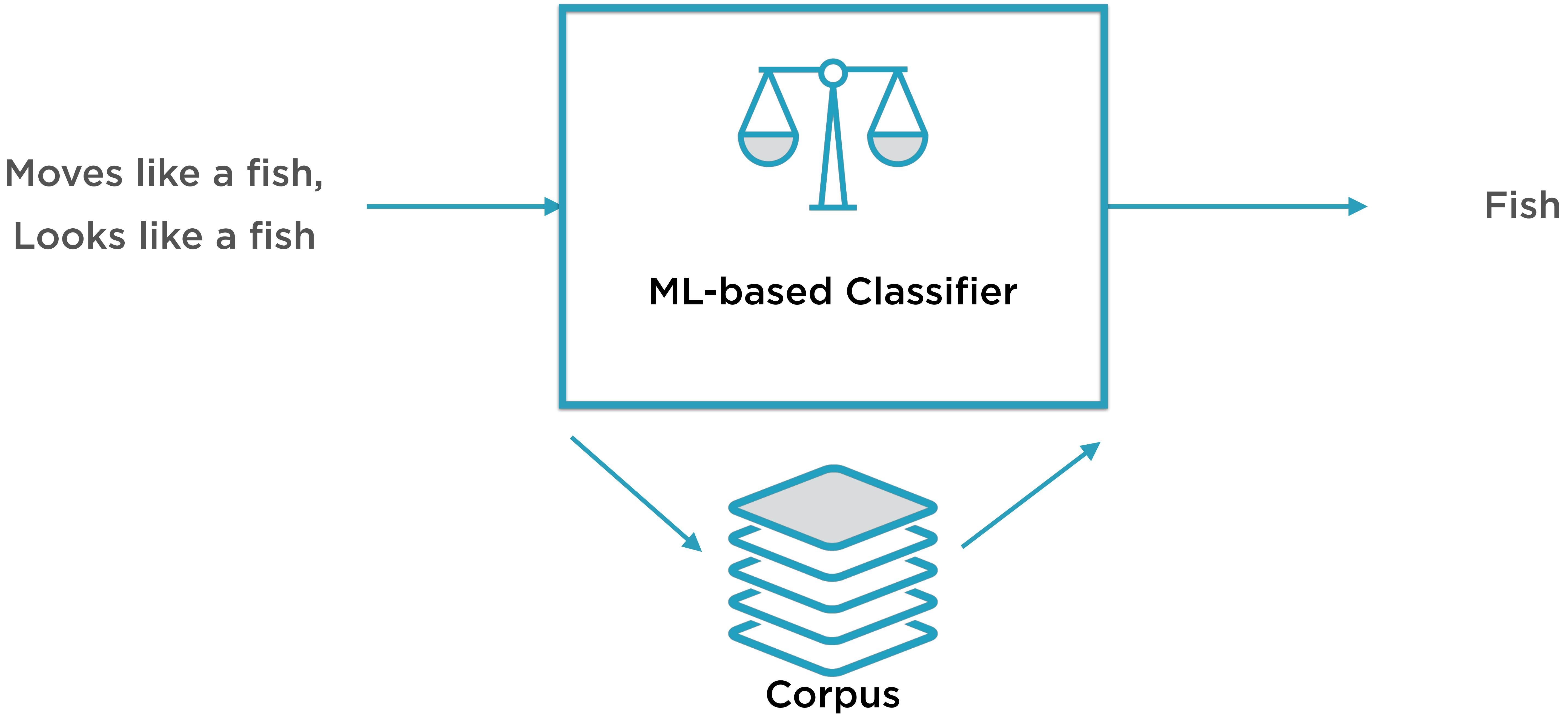
Input: Feature Vector



ML-based Binary Classifier



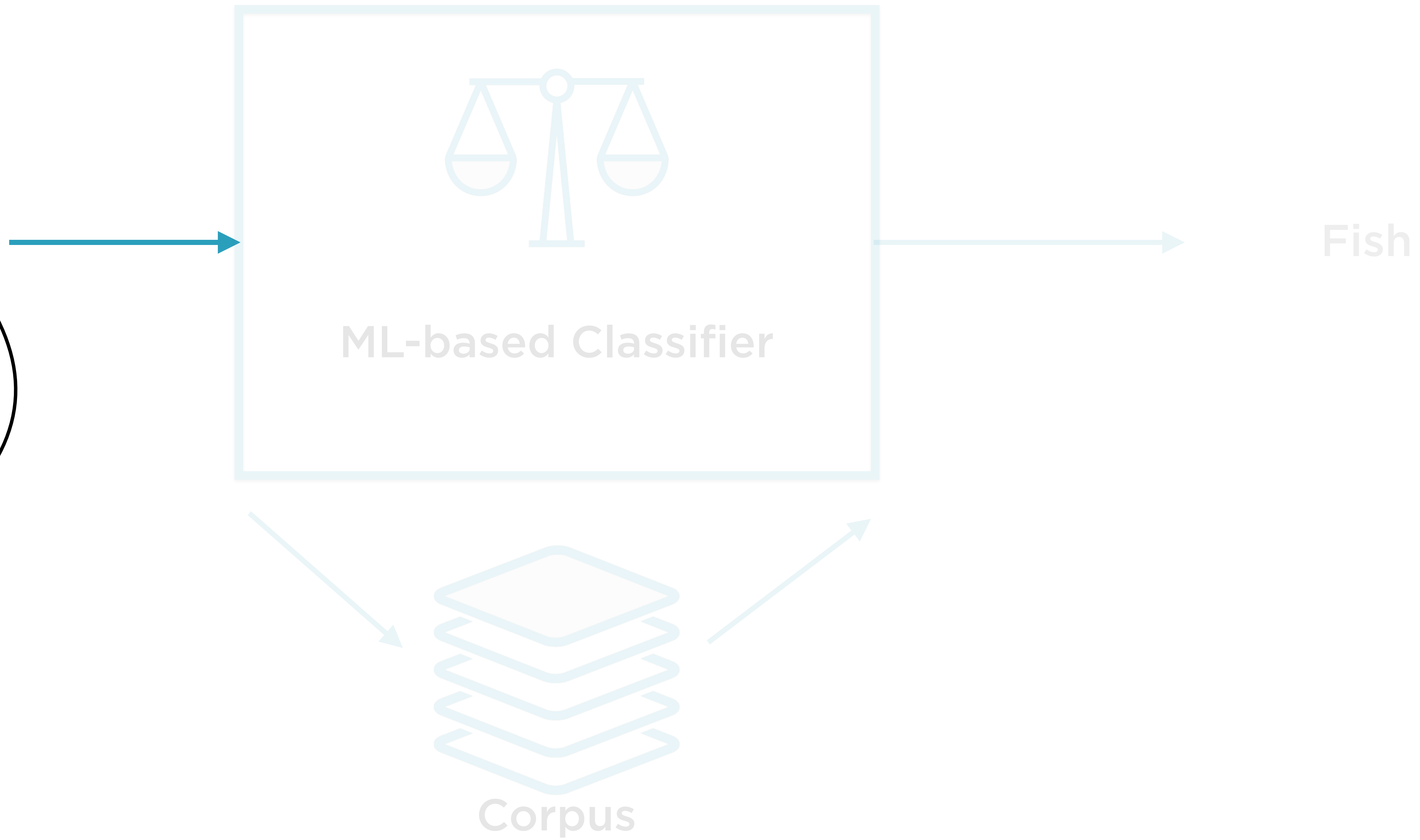
ML-based Binary Classifier



ML-based Binary Classifier

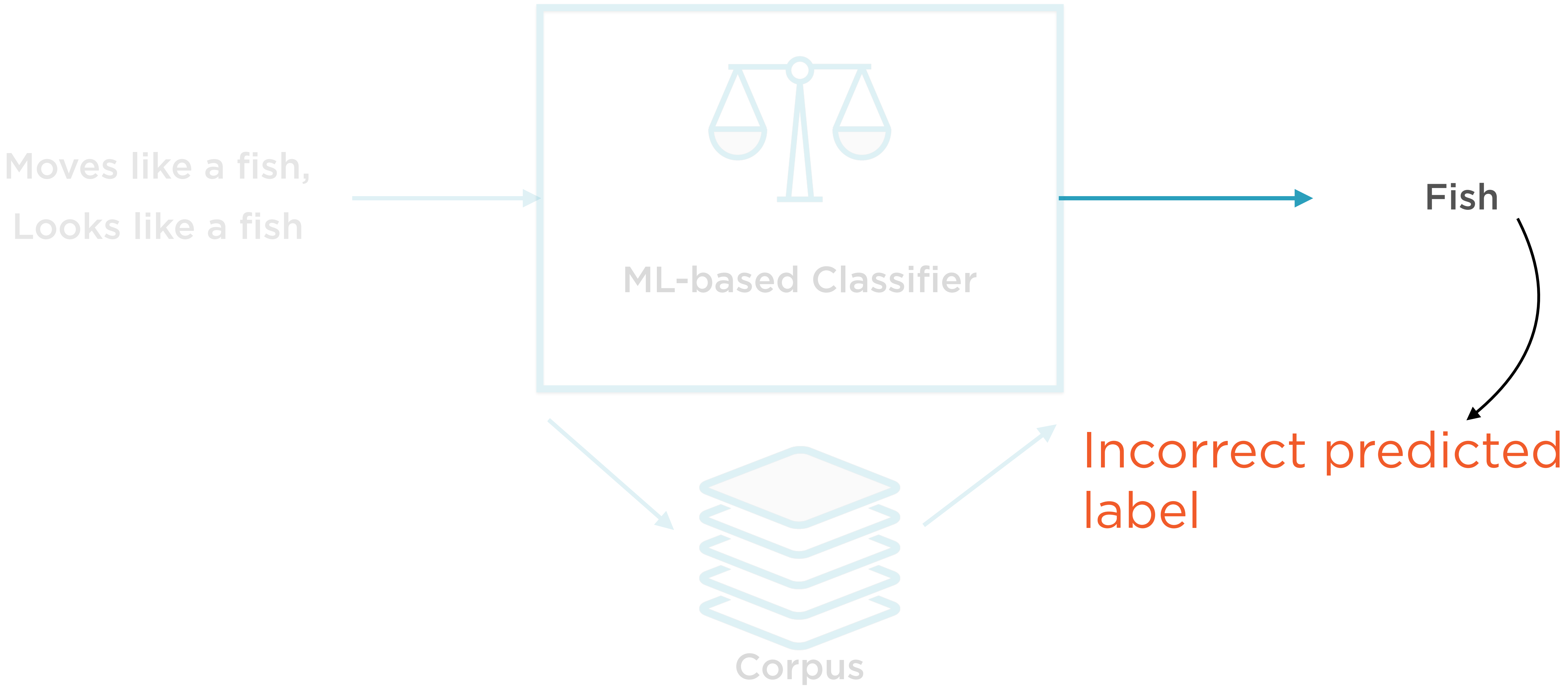
Moves like a fish,

Looks like a fish



Poor choice of features

ML-based Binary Classifier

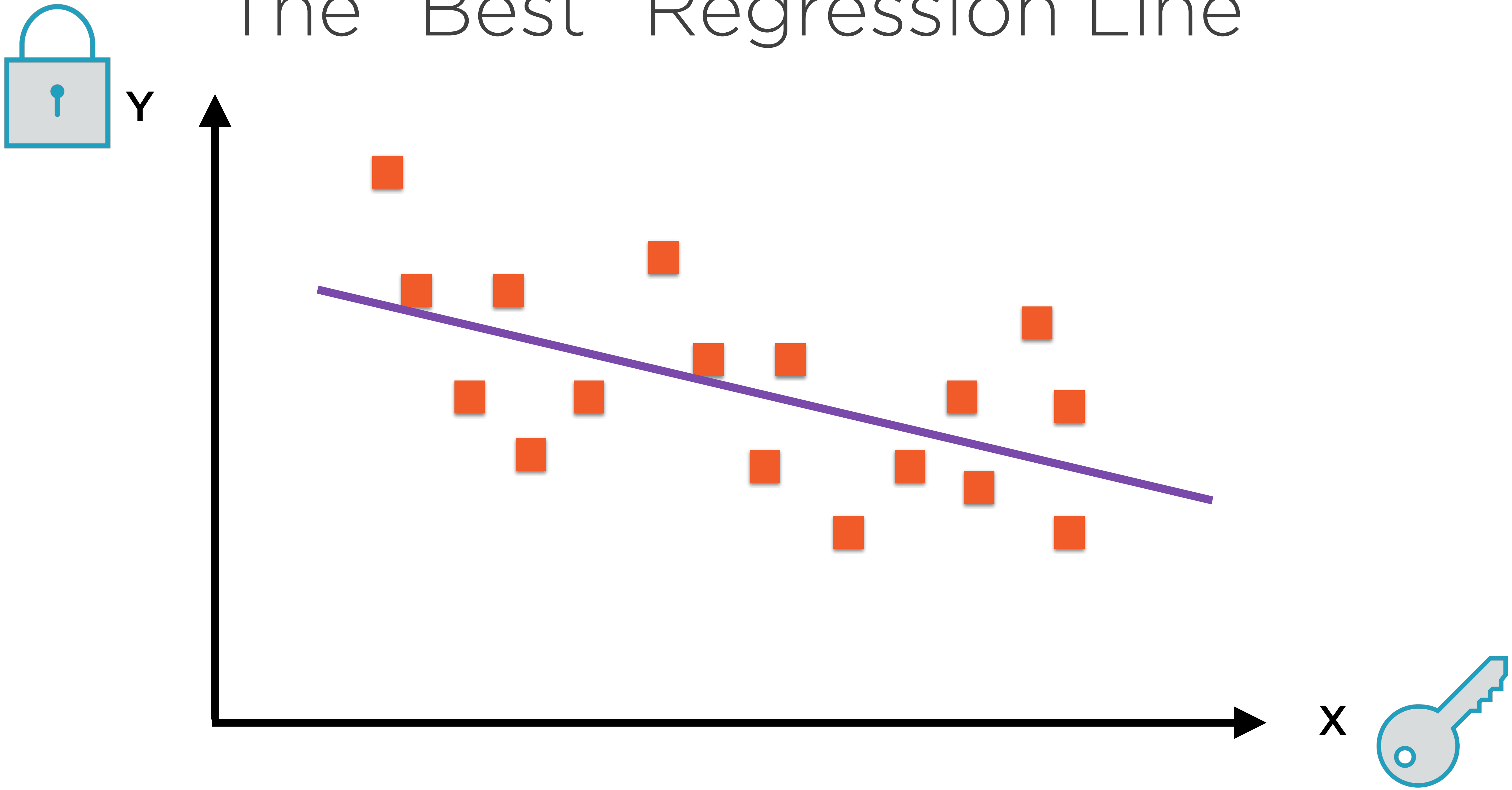


```
y = f(x)
```

Supervised Machine Learning

Most machine learning algorithms seek to “learn” the function f that links the features and the labels

The "Best" Regression Line



Linear Regression involves finding the "best fit" line via a training process

```
y = Wx + b
```

$$f(x) = Wx + b$$

Linear regression specifies, up-front, that the function f is linear

```
def doSomethingReallyComplicated(x1,x2…):
  …
  …
  …
  return complicatedResult
```

f(x) = doSomethingReallyComplicated(x)

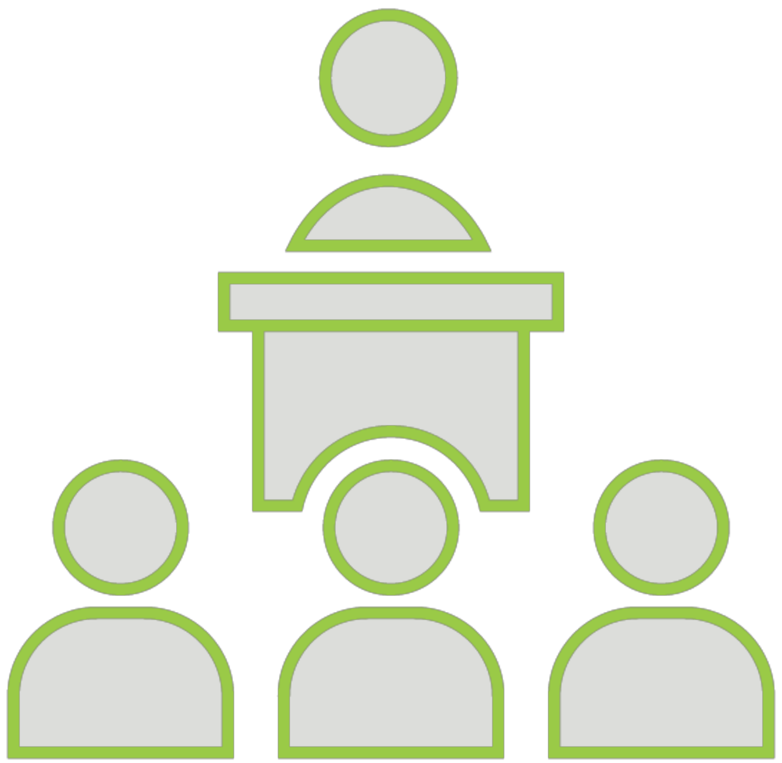
ML algorithms such as neural network can “learn” (reverse-engineer) pretty much anything given the right training data

Everything so far discussed really applied only to **Supervised Learning**

Unsupervised Learning does **not** have:

- **y** variables
- a **labeled** corpus

Types of ML Algorithms



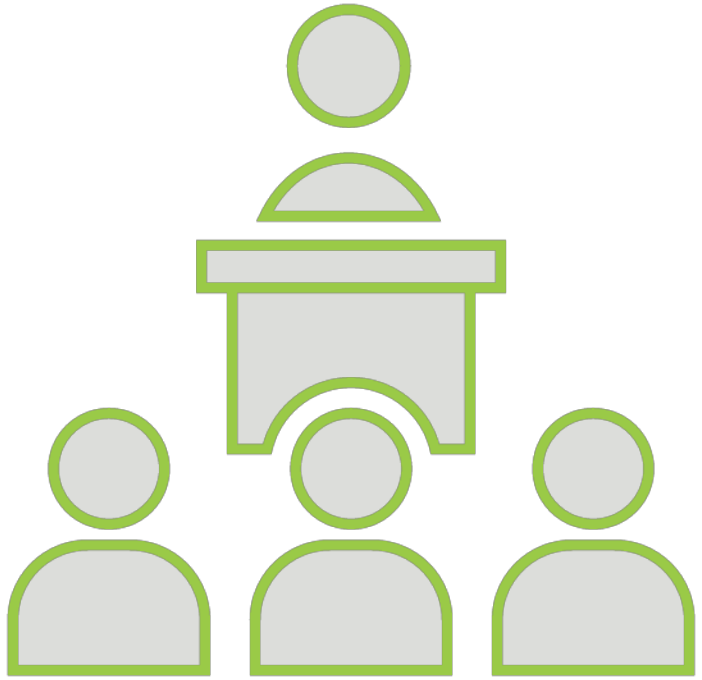
Supervised

Labels associated with the training data is used to correct the algorithm

Unsupervised

The model has to be set up right to learn structure in the data

Supervised Learning



Input variable x and output variable y

Learn the mapping function `y = f(x)`

Approximate the mapping function so for new values of x we can predict y

Use existing dataset to correct our mapping function approximation

Unsupervised Learning

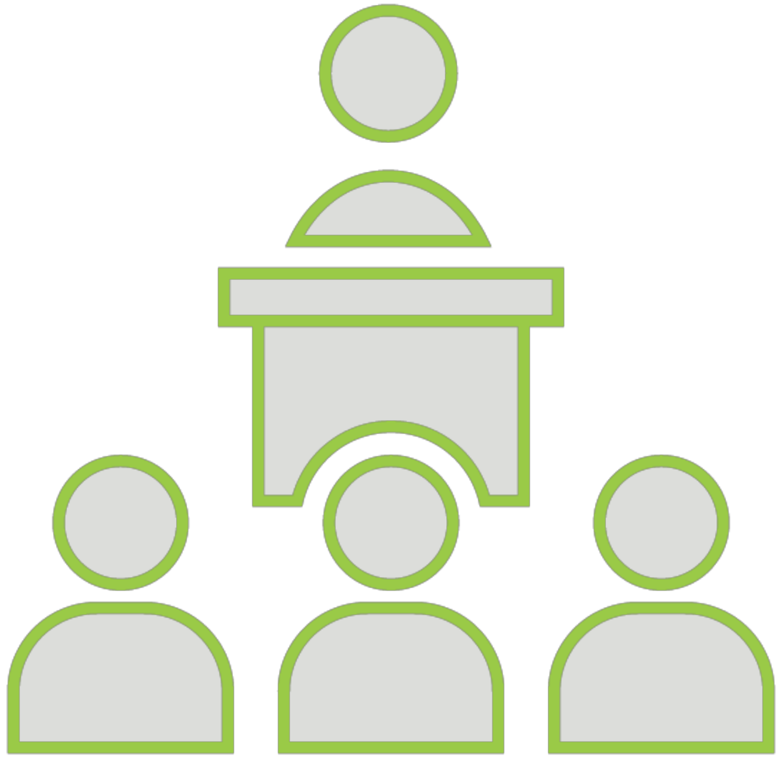
Only have input data x – no output data

Model the underlying structure to learn more about data

Algorithms discover the patterns and structure in the data

Clustering Algorithms

Types of ML Algorithms



Supervised

Labels associated with the training data is used to correct the algorithm

Unsupervised

The model has to be set up right to learn structure in the data

Types of ML Algorithms

Supervised

Labels associated with the training data is used to correct the algorithm

Unsupervised

The model has to be set up right to learn structure in the data

Why Look Within?

To be emotionally self-sufficient

To learn what values matter (to you)

Identify others who share them…

..and those who don’t

Eliminate what does not matter

In general, to train yourself to navigate the outside world

Why Look Within

In Life	In Machine Learning
To be emotionally self-sufficient	To make unlabelled data self-sufficient
To learn what values matter to you	Latent factor analysis
Identify others who share them…	Clustering
..and those who don't	Anomaly detection
Eliminate what does not matter	Quantization
In general, to train yourself to navigate the outside world	Pre-training for supervised learning problems (classification, regression)

Unsupervised Learning Use-cases

ML Technique	Use-case
To make unlabelled data self-sufficient	Identify photos of a specific individual
Latent factor analysis	Find common drivers of 200 stocks
Clustering	Find relevant document in a corpus
Anomaly detection	Flag fraudulent credit card transactions
Quantization	Compress true color (24 bit) to 8 bit
Pre-training for supervised learning problems (classification, regression)	All of the above!

Unsupervised Learning Use-cases

What	How
To make unlabelled data self-sufficient	Autoencoder
Latent factor analysis	Autoencoder
Clustering	Clustering
Anomaly detection	Autoencoder
Quantization	Clustering
Pre-training for supervised learning problems (classification, regression)	All of the above!

Unsupervised ML Algorithms

Clustering

Identify patterns in data items e.g. K-means clustering

Autoencoding

Identify latent factors that drive data e.g. PCA

Unsupervised ML Algorithms

Clustering

Identify patterns in data items e.g. K-means clustering

Autoencoding

Identify latent factors that drive data e.g. PCA

Patterns in Data

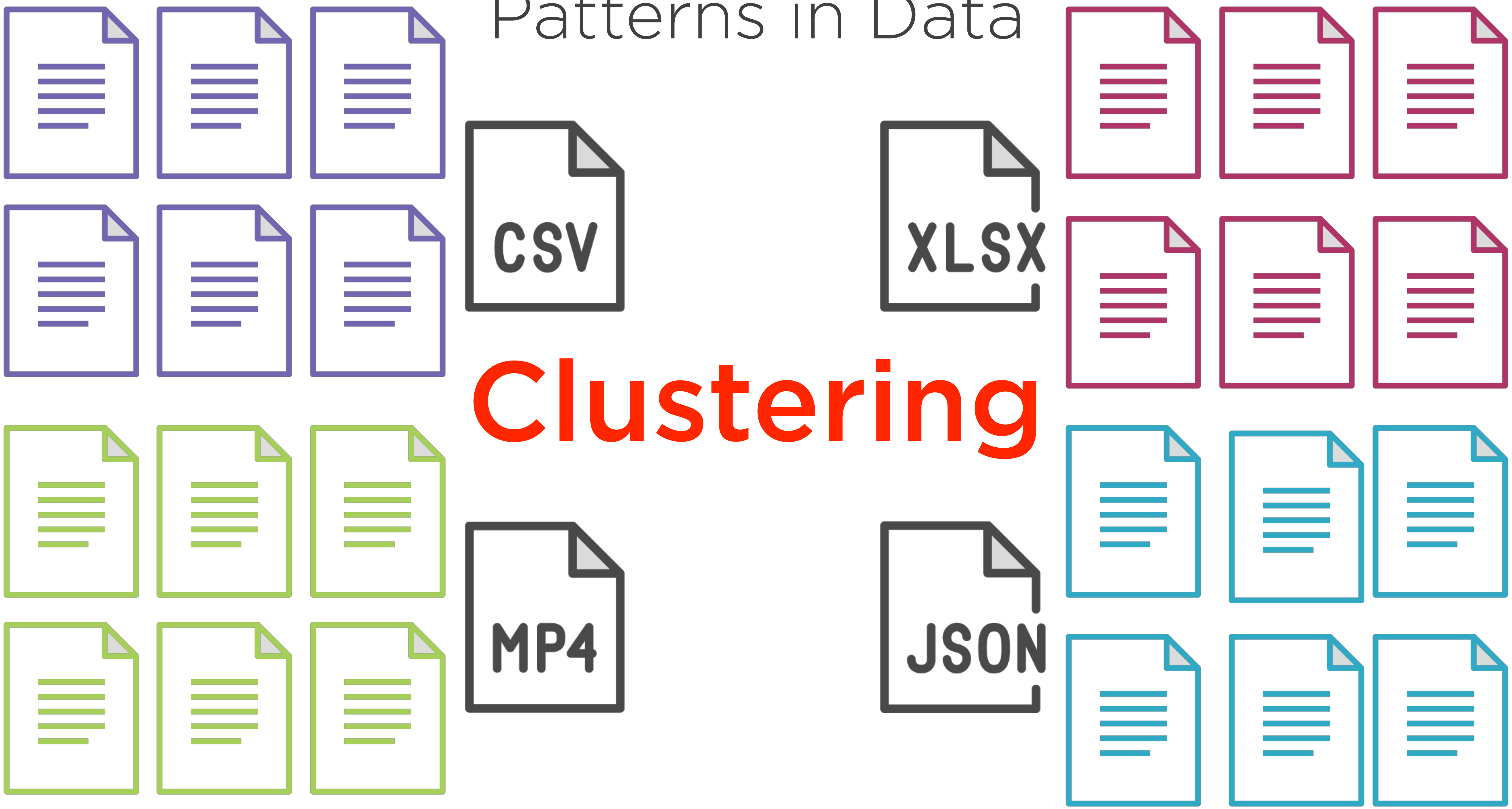
Patterns in Data

How do you make sense of this?

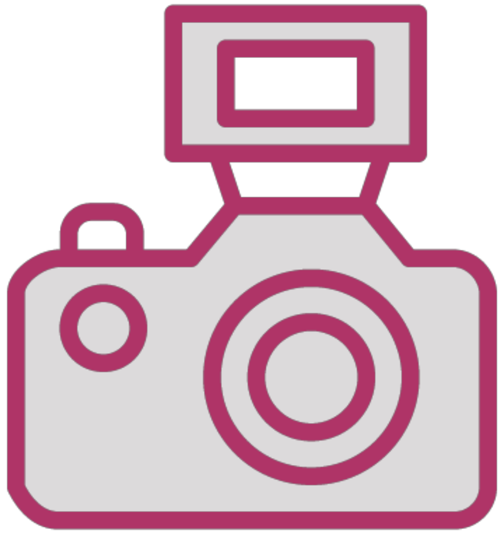
Patterns in Data

Group them based on some common attributes

Patterns in Data

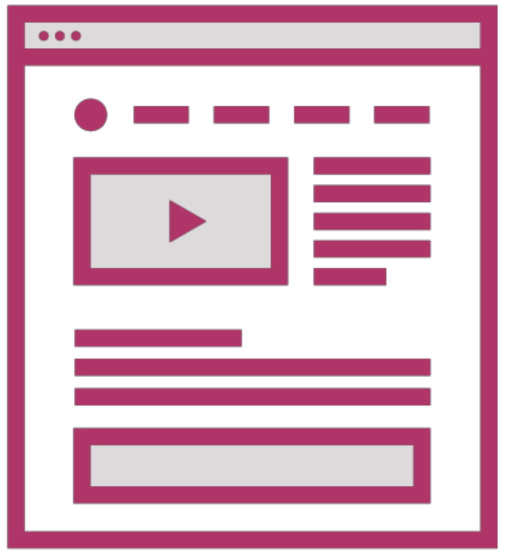


Clustering



Products sold on Amazon

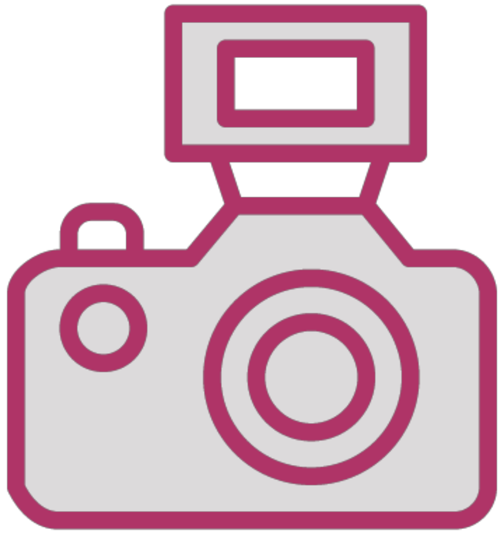
People on Facebook



Websites indexed by Google

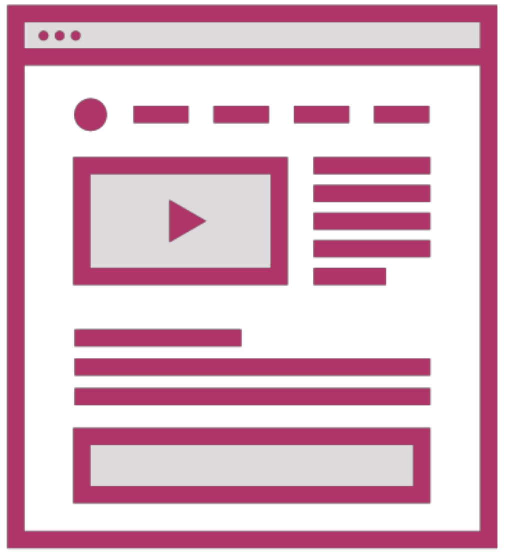
What if you want to group more complex entities?

Clustering



Products sold
on Amazon

People on
Facebook

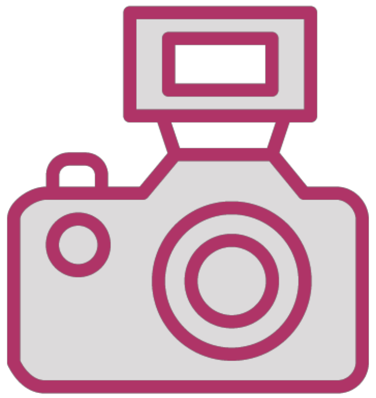


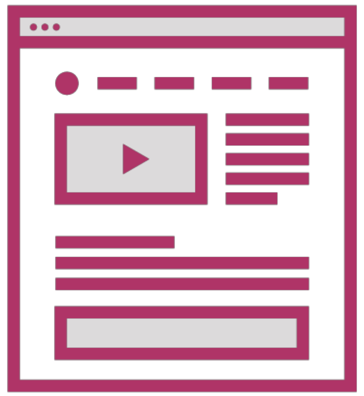
Websites indexed
by Google

Too many entities, too many
attributes per entity

Huge complexity

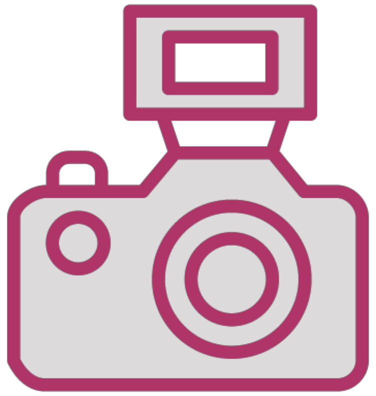
Clustering





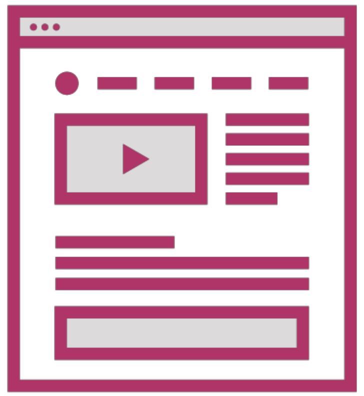
Anything can be represented by a set of numbers

Clustering



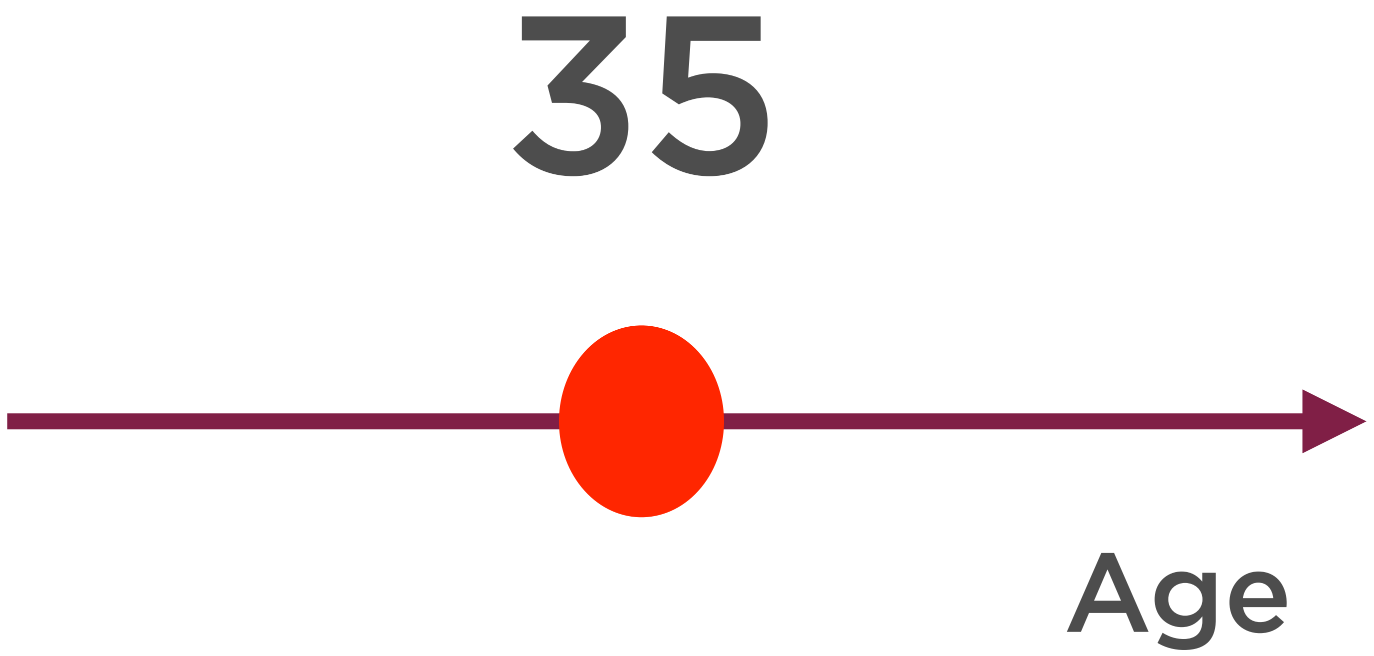
Product ID, Timestamp, Amount

Age, Height, Weight

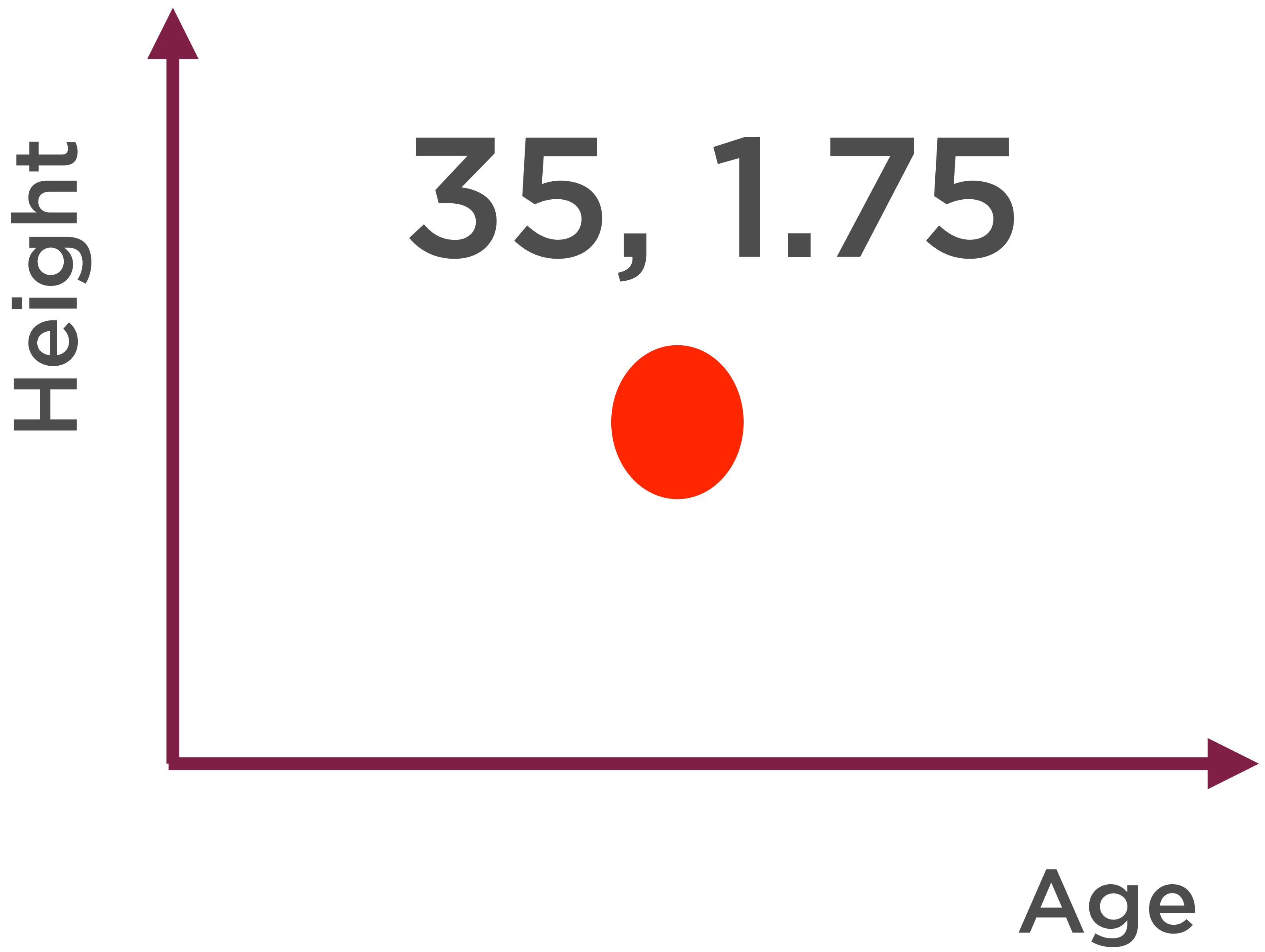


Length, word frequencies

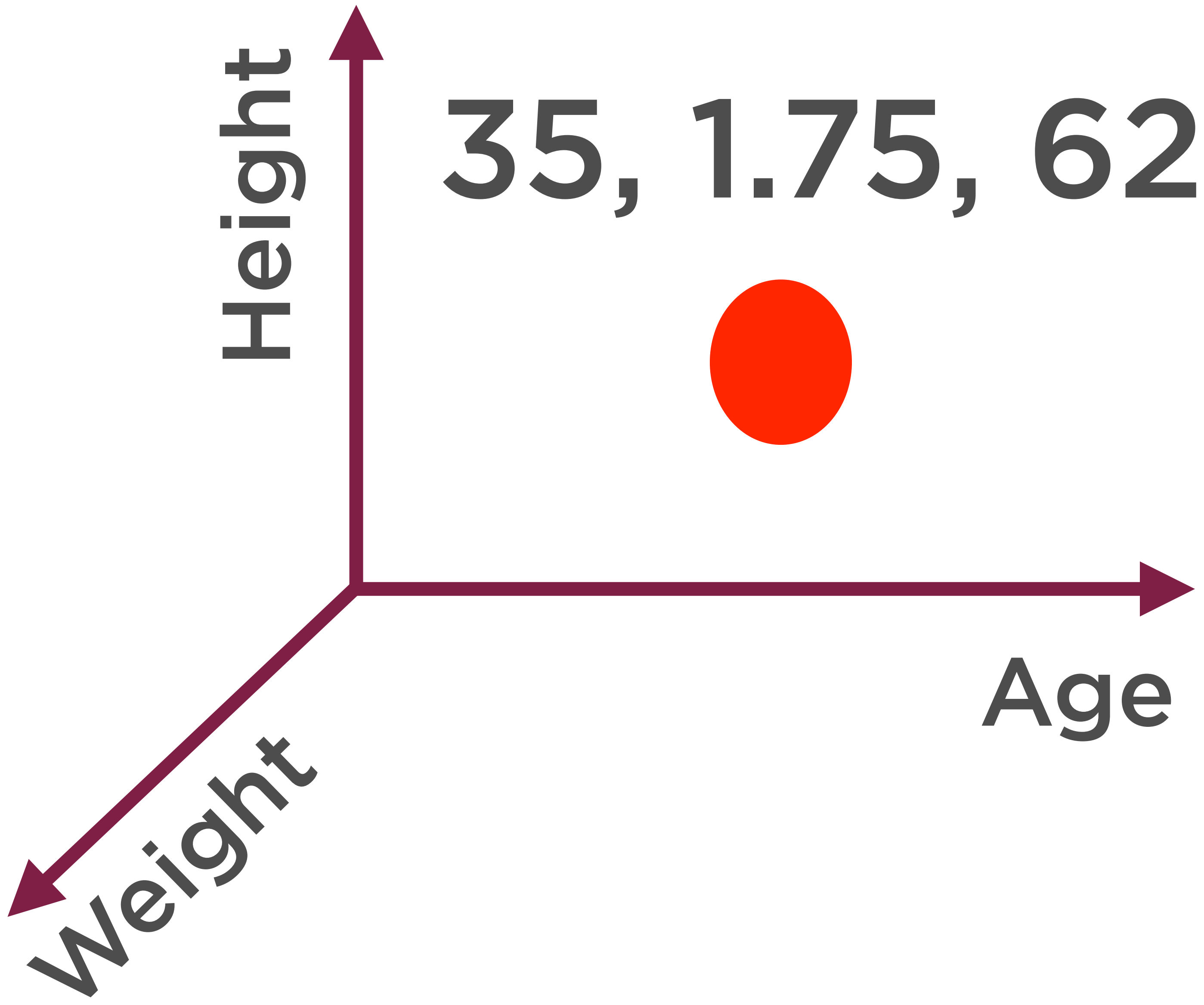
Age, Height, Weight



Age, Height, Weight



Age, Height, Weight

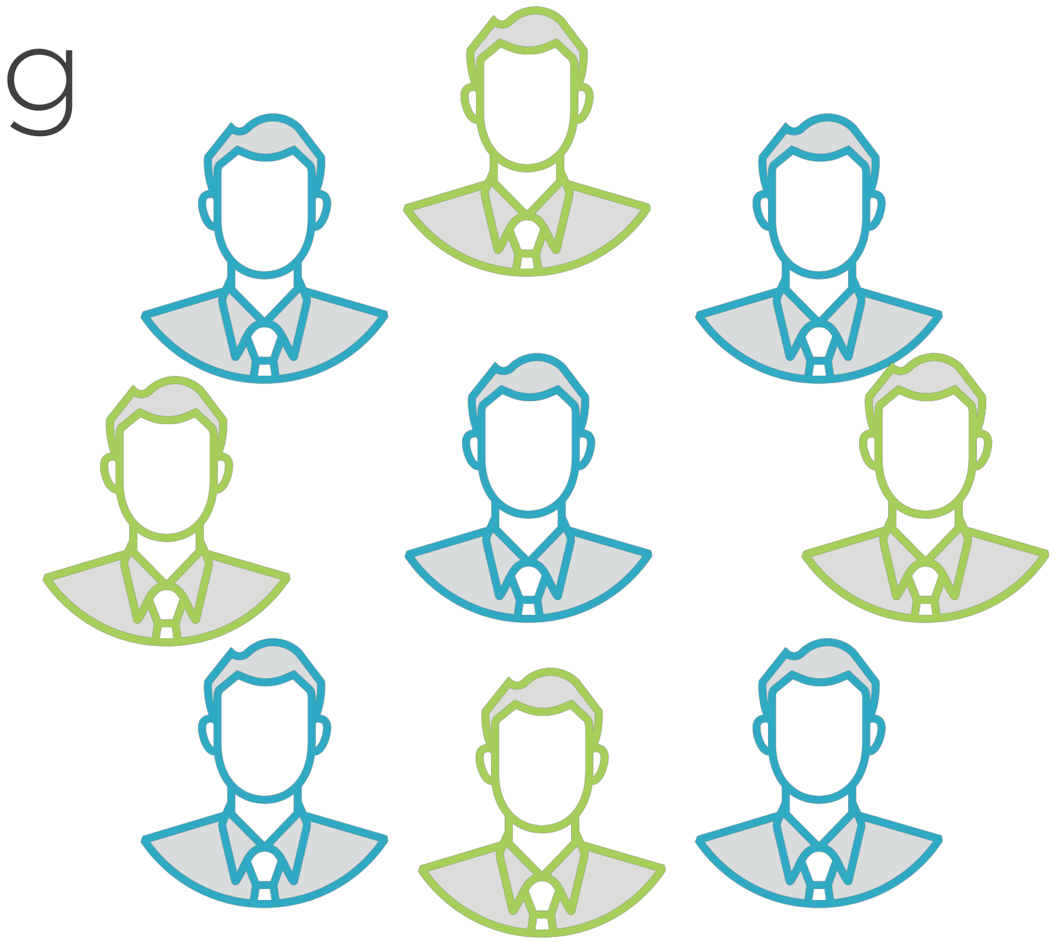


A set of N numbers represents a point in an **N-dimensional Hypercube**

Clustering

A set of points, each representing a Facebook user

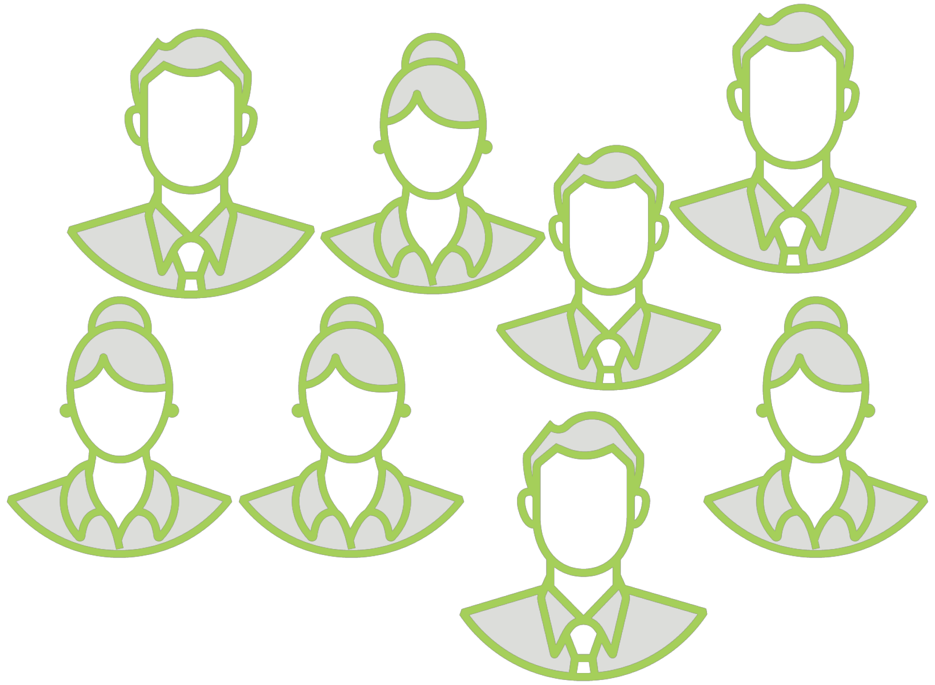
Clustering



Same group = similar

Different group = different

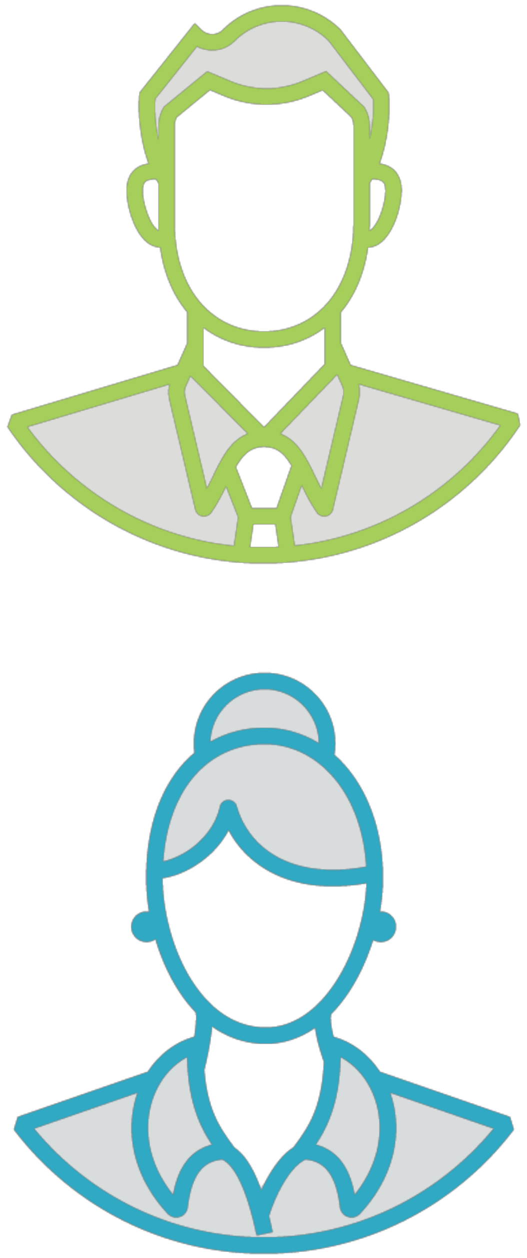
Clustering



Same group = similar

Different group = different

Users in a Cluster

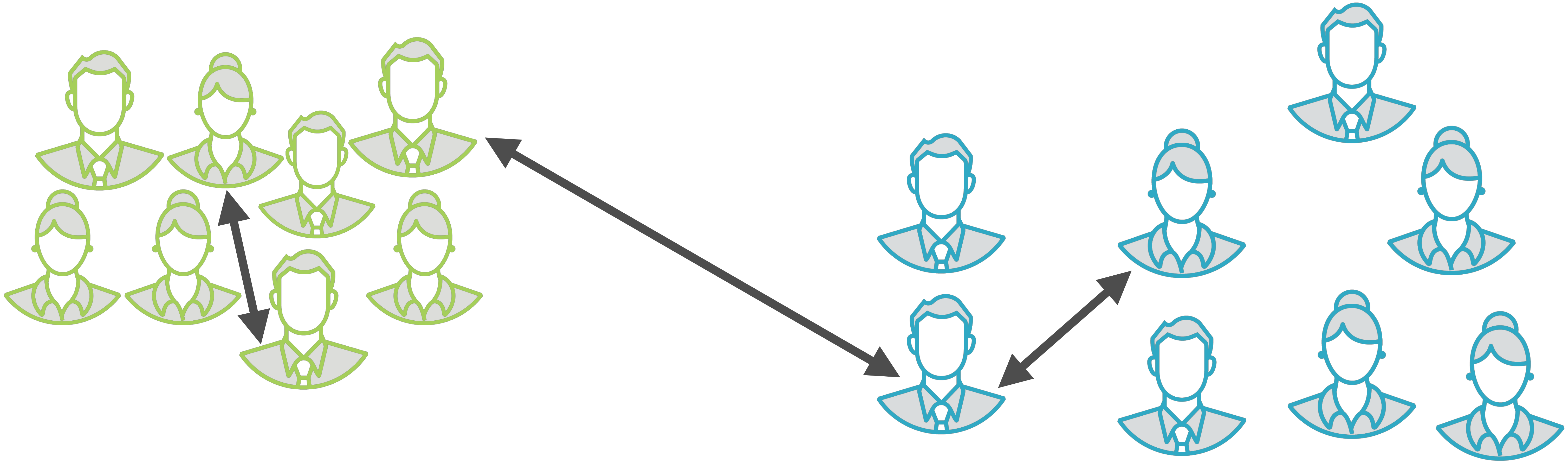


May like the same kind of music

May have gone to the same high school

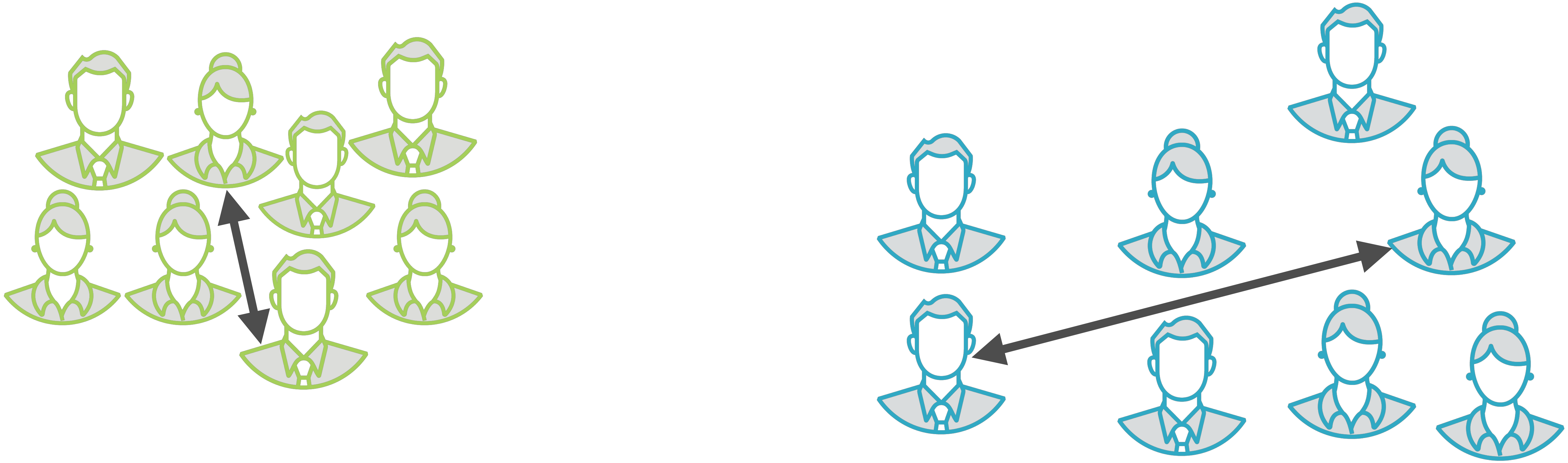
May enjoy the same kinds of movies

Clustering



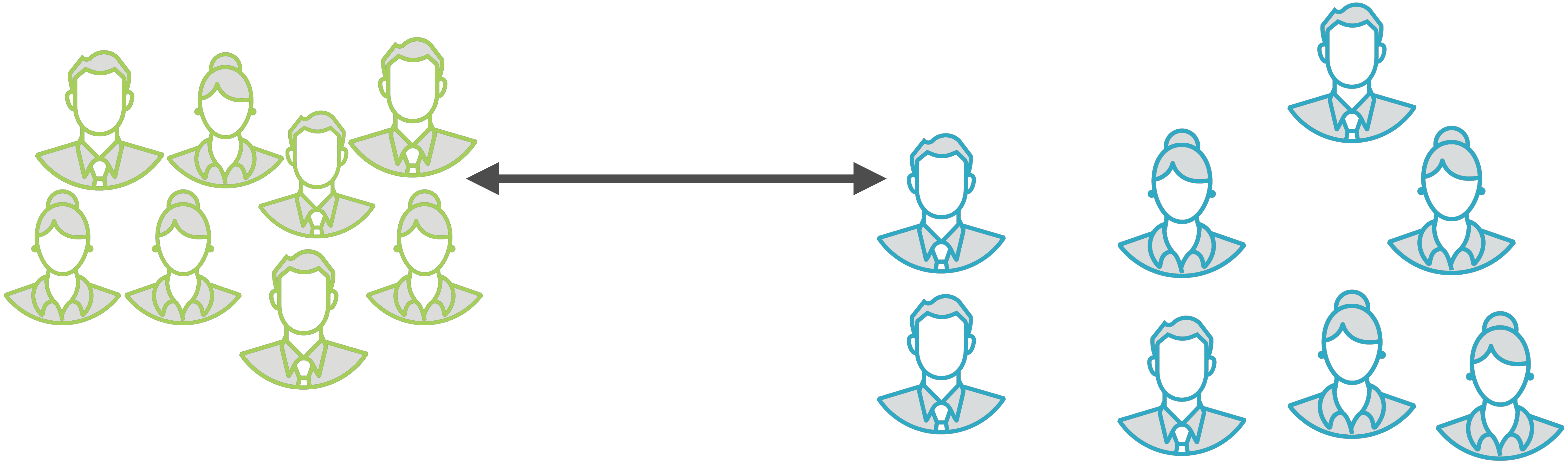
The **distance** between users indicates how **similar** they are

Maximize Inter-cluster Similarity



Distances between users in the same cluster should be small

Maximize Inter-cluster Similarity



Between users in different clusters distances should be large

Entities in the **same group are very similar** and entities in **different groups are very different**

Relevant Documents in a Corpus

Rich document archives (Wikipedia, digitized books)

Hard to identify content relevant to specific user or query

Clump documents into semantically similar groups

Clustering

Color Quantization

True color images represent each image with 24 bits/pixel

Many displays and image formats use 8 bits/pixel

Statically choosing 256 not optimal - too few blues in a seascape!

Use clustering to identify the 256 most representative colors

Quantize each true color to nearest shade

K-means Clustering

Clustering Objective

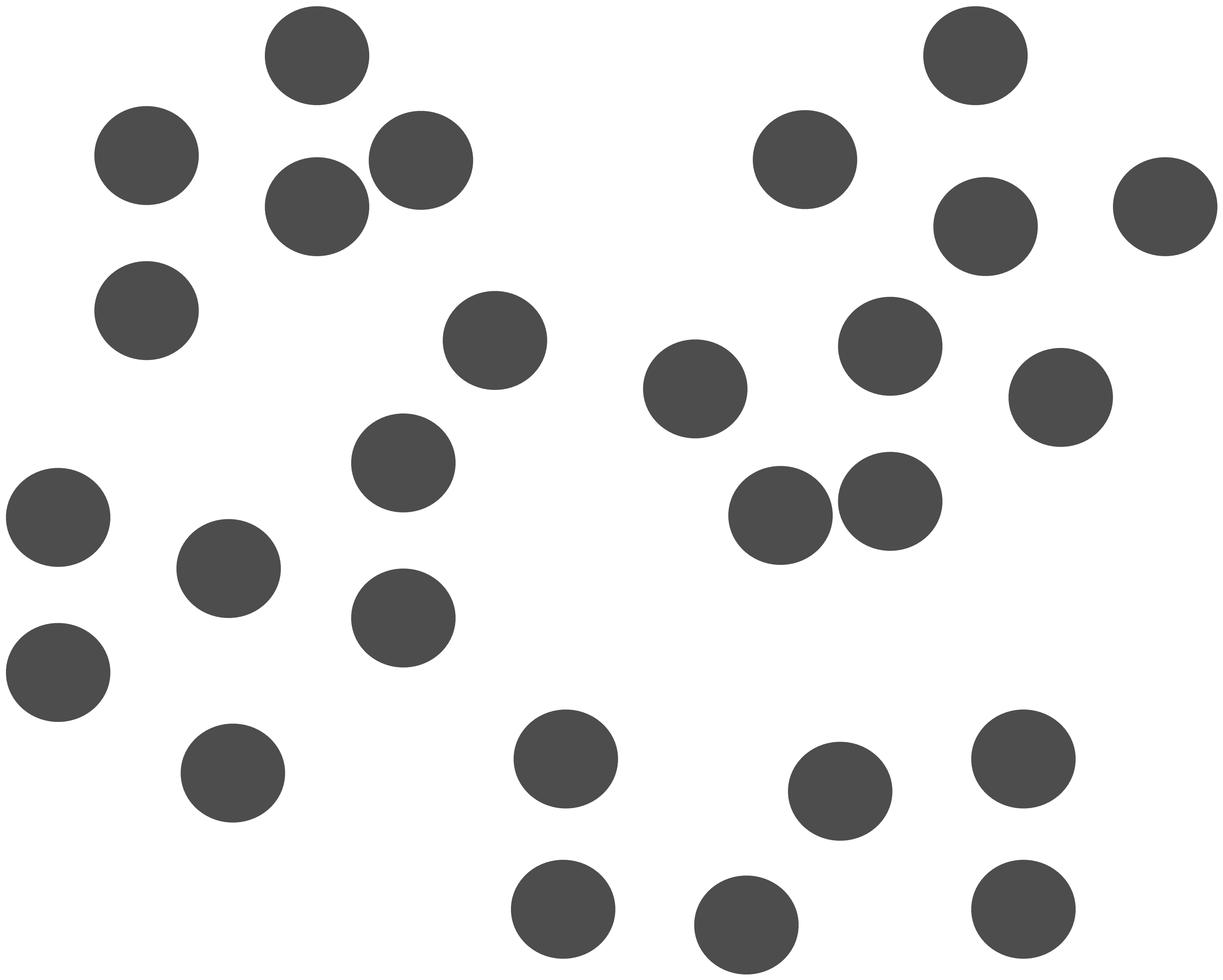
Maximize intra-cluster similarity

Minimize inter-cluster similarity

The **K-means Clustering** algorithm is a famous Machine Learning algorithm to achieve this

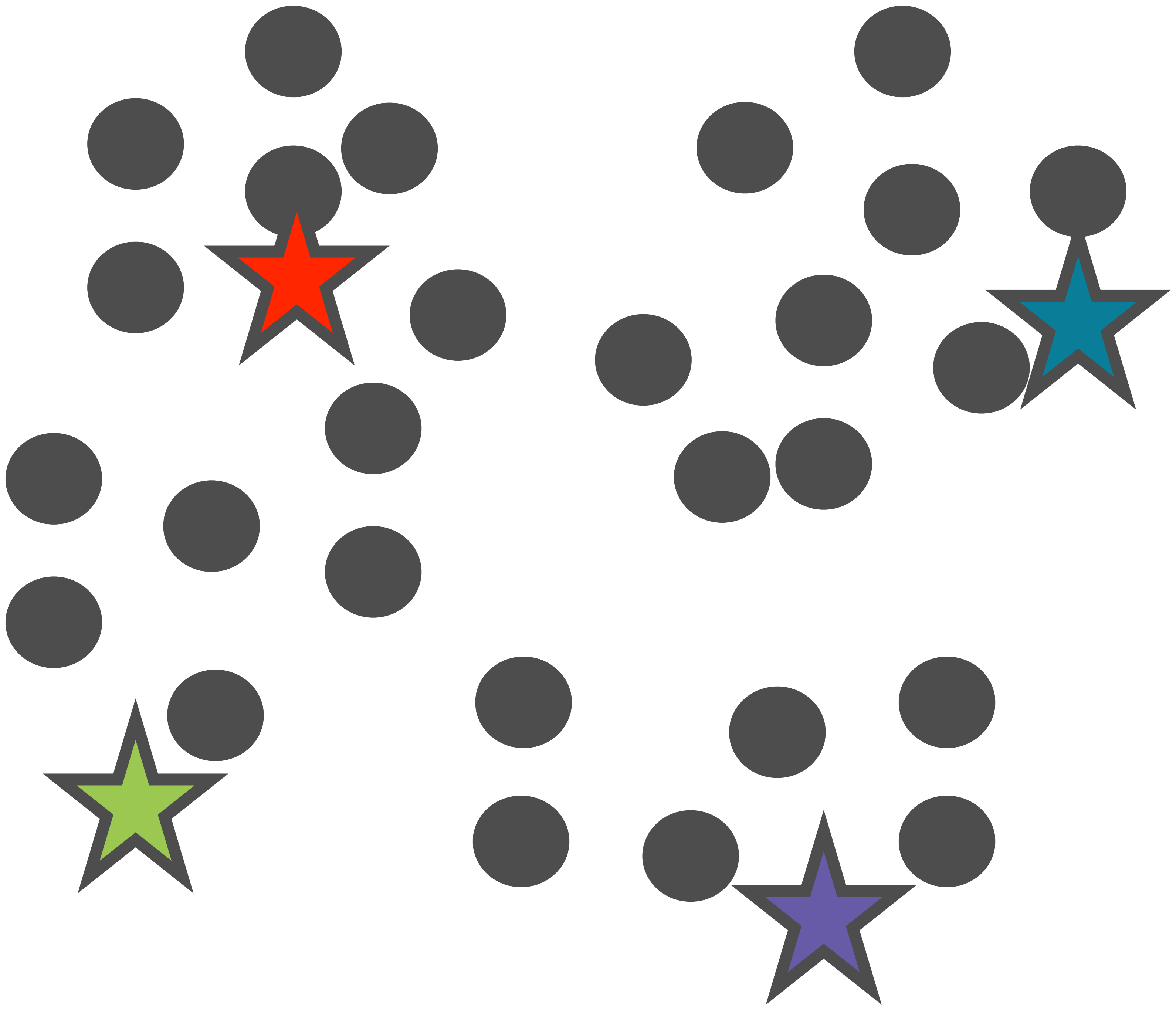
K-means Clustering

Initialize K centroids i.e means



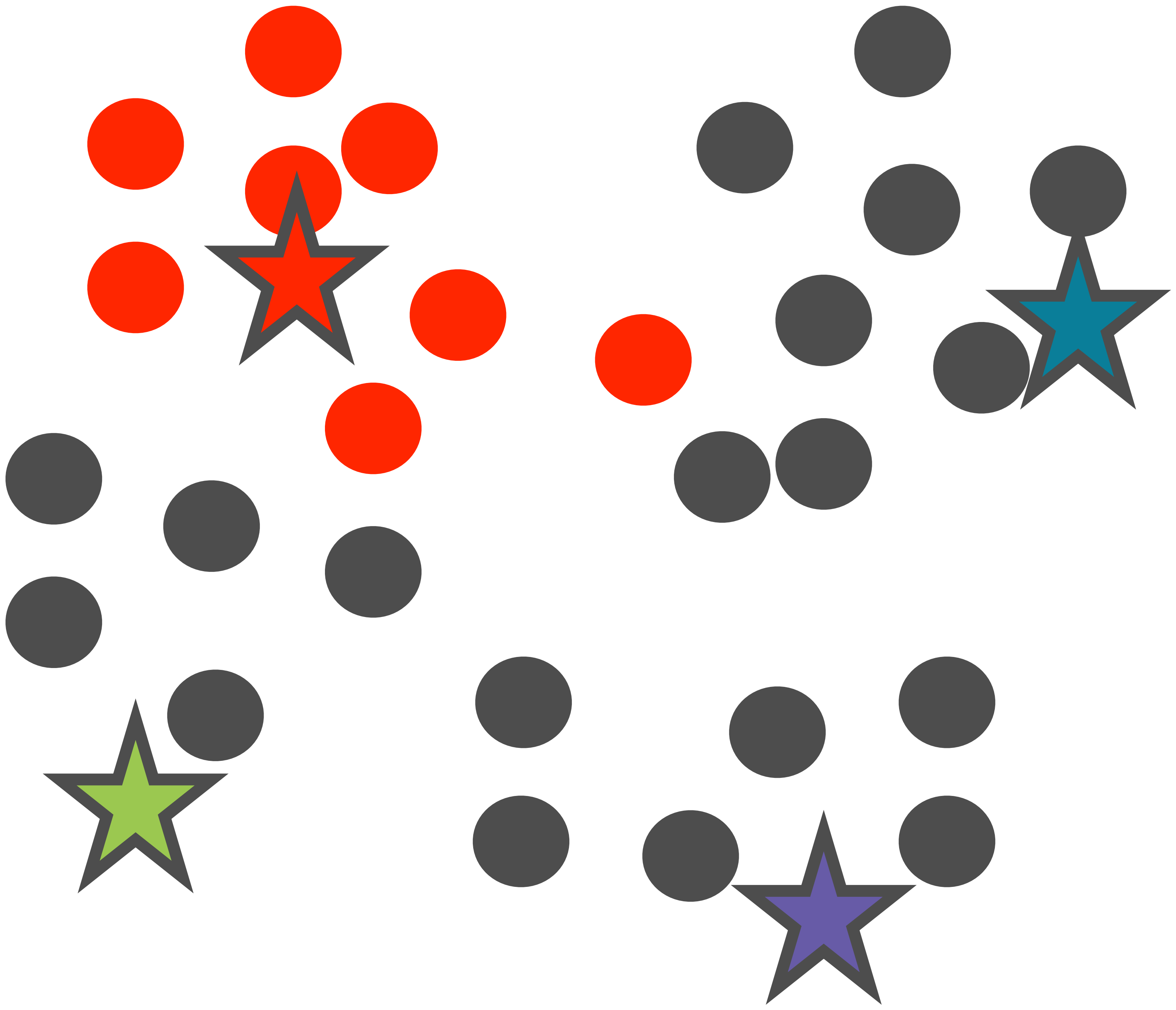
K-means Clustering

Assign each point to a cluster



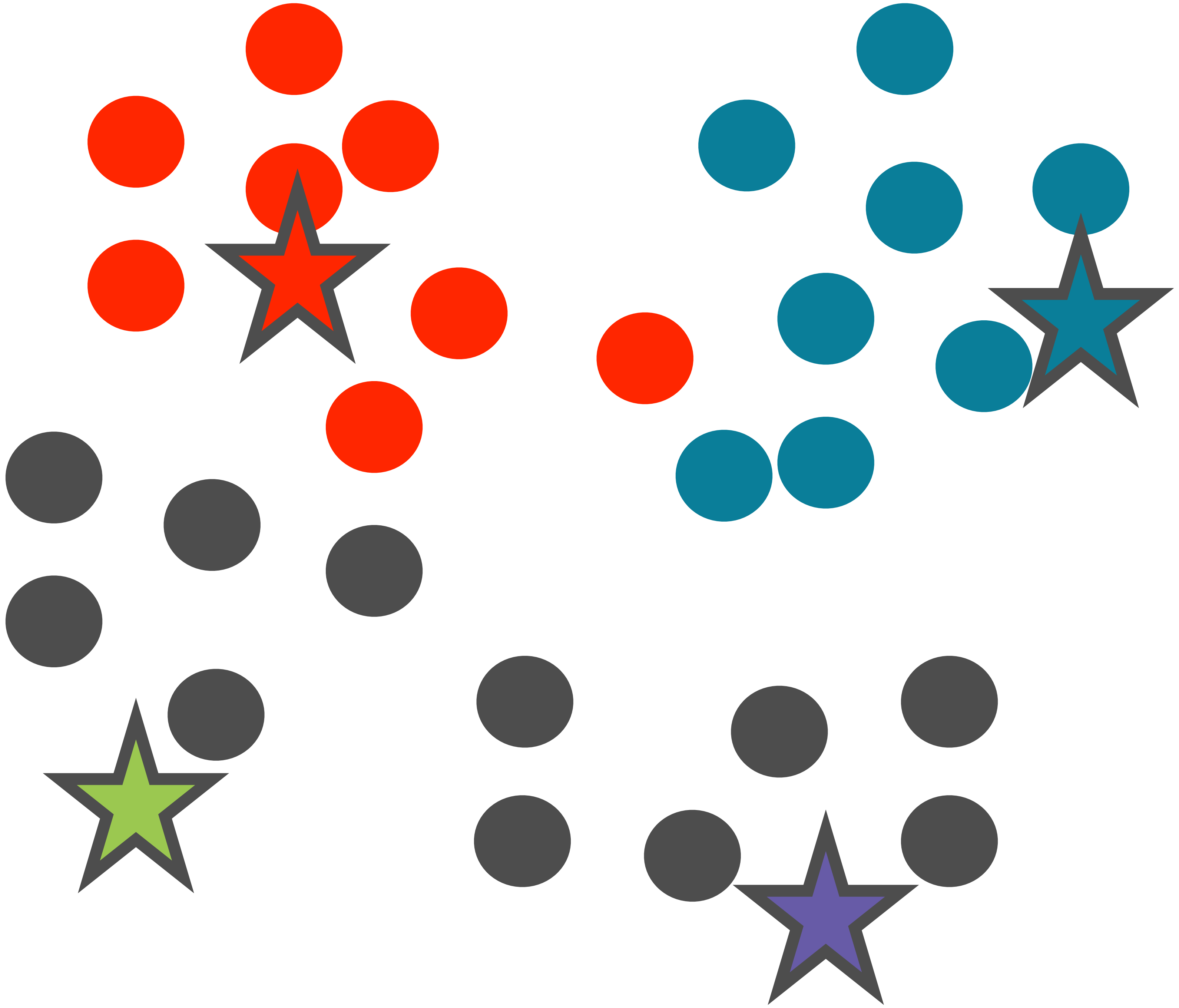
K-means Clustering

Assign each point to a cluster



K-means Clustering

Assign each point to a cluster



K-means Clustering

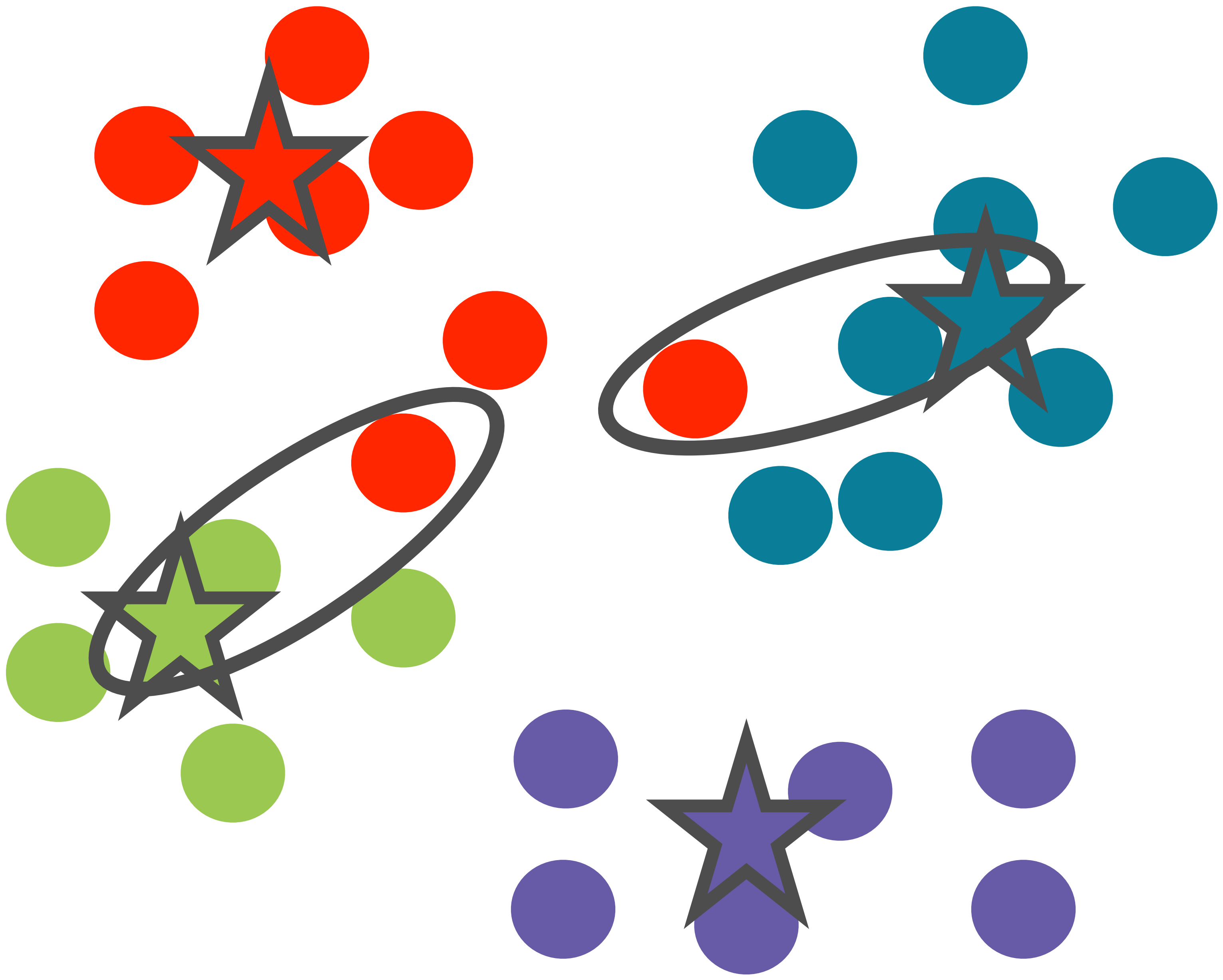
Assign each point to a cluster

Recalculate the mean for each cluster

K-means Clustering

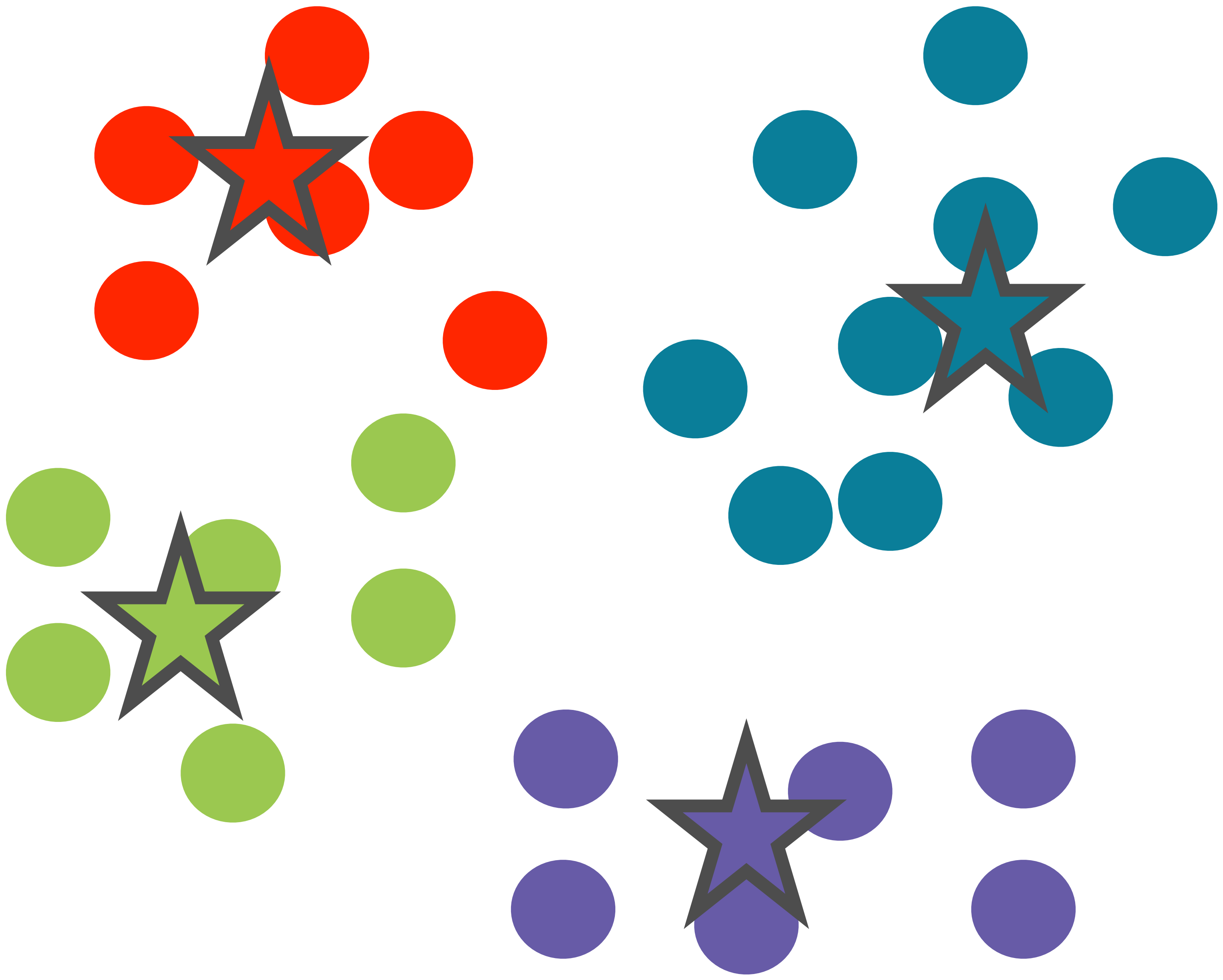
K-means Clustering

Re-assign the points to clusters

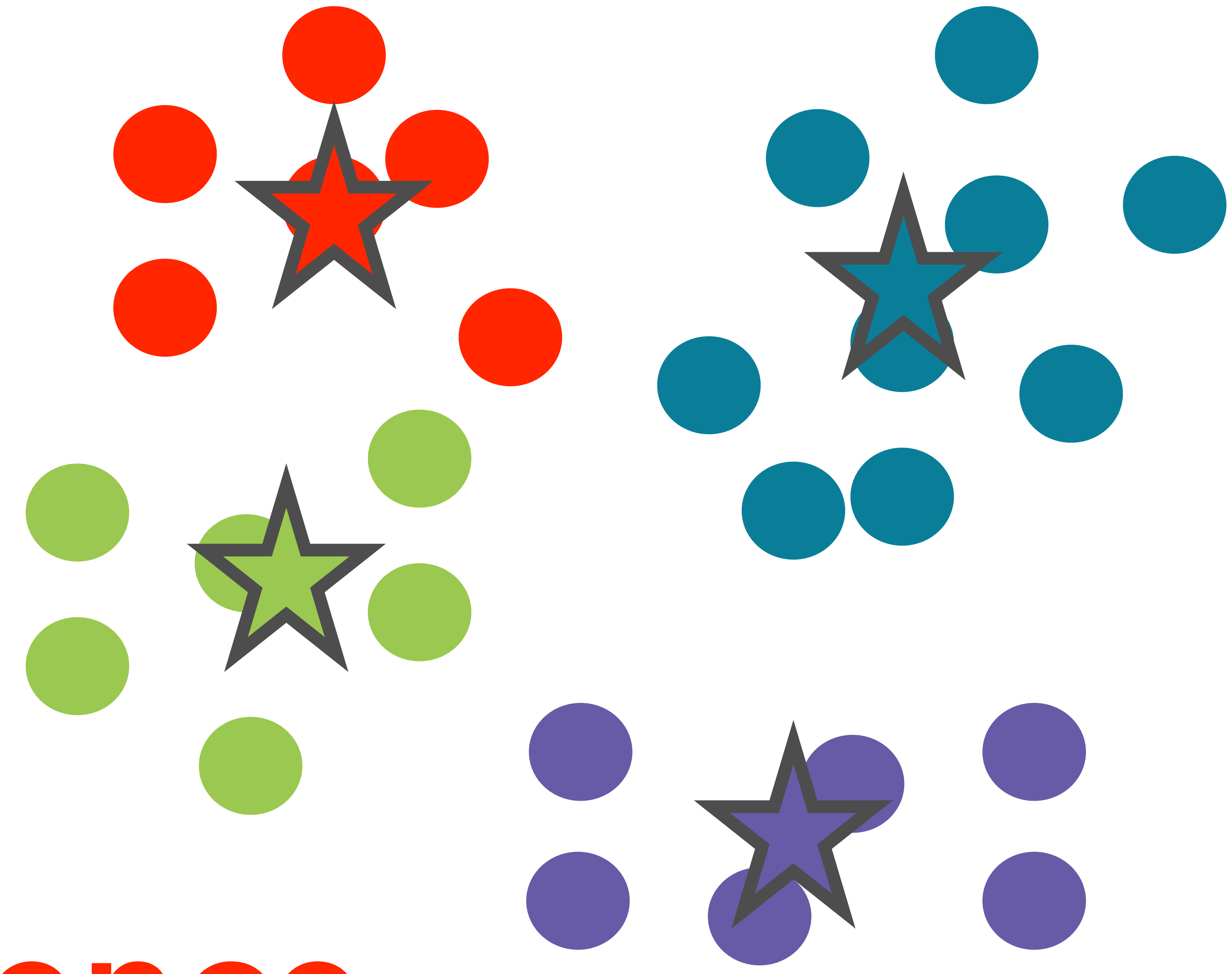


K-means Clustering

Iterate until points are in their final clusters

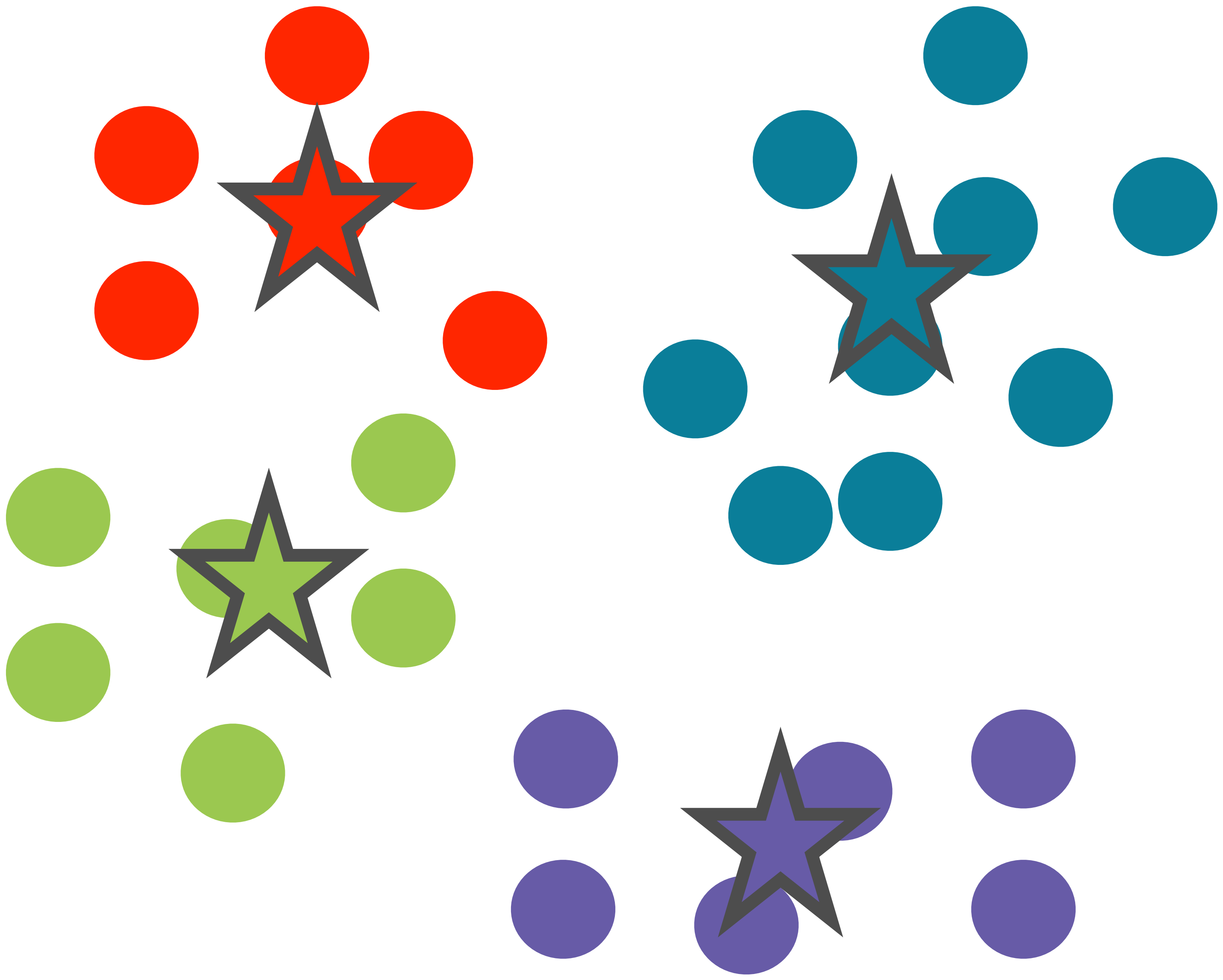


K-means Clustering



Convergence

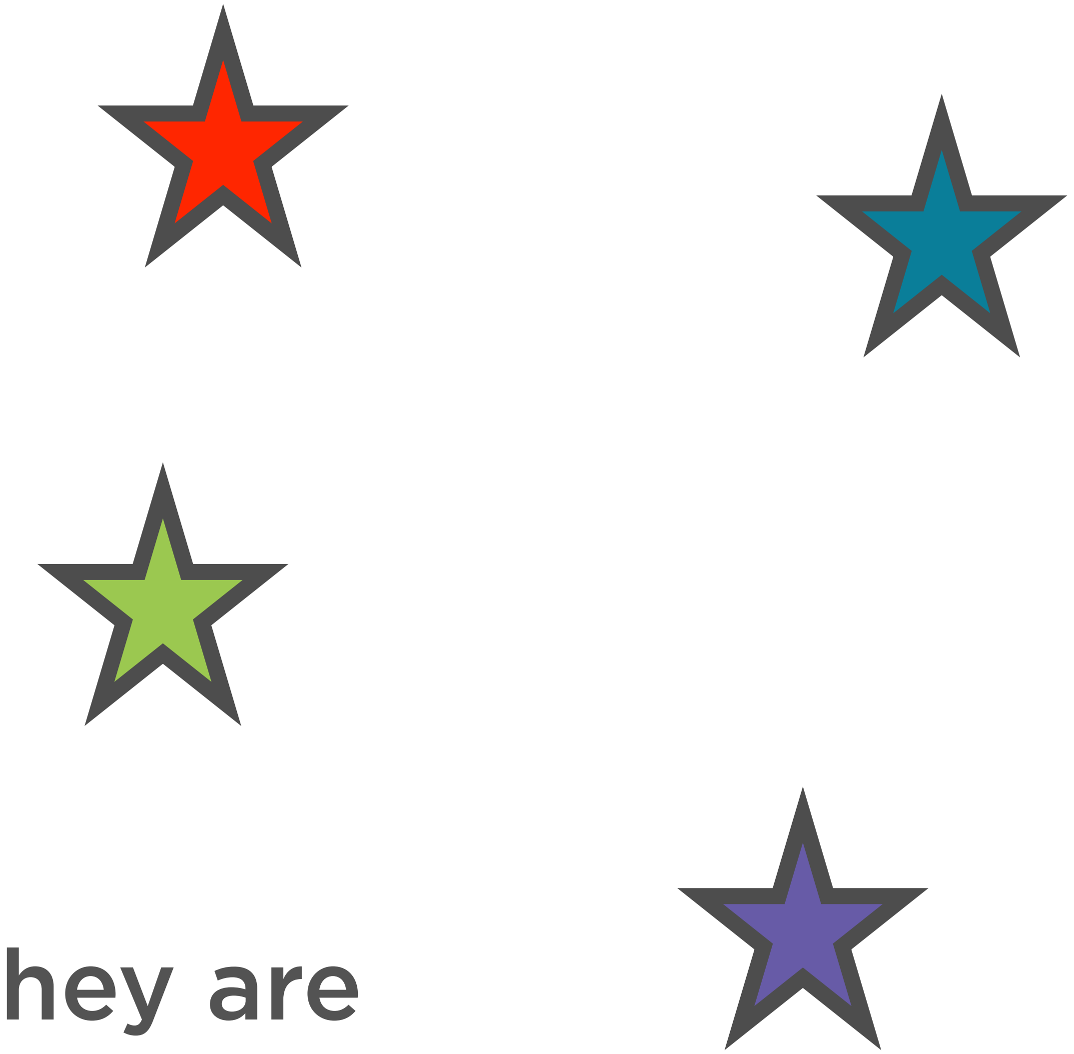
K-means Clustering



K-means Clustering

Each cluster has a representative point called a reference vector

K-means Clustering



Because of how they are calculated, these reference vectors are often called centroids

```
Initialise K centroids


Repeat:
  For each data point:
    Assign to “nearest” cluster



  For each centroid:
    Update coordinates

Have centroids converged?
   Yes: Stop, we’re done
    No: Keep iterating
```

- Pick an initial solution (algorithms exist to pick well)
- Iterate until convergence
    - Update assignments of points to clusters
    - Update coordinates of reference vectors
- Keep iterating until we converge

```
Initialise K centroids


Repeat:
  For each data point:
    Assign to "nearest" cluster


  For each centroid:
    Update coordinates

Have centroids converged?
   Yes: Stop, we're done
    No: Keep iterating
```

- **Hyperparameters**
  - Number of clusters
  - Initial values of centroids

```
Initialise K centroids

Repeat:
  For each data point:
    Assign to "nearest" cluster

  For each centroid:
    Update coordinates

Have centroids converged?
   Yes: Stop, we're done
    No: Keep iterating
```

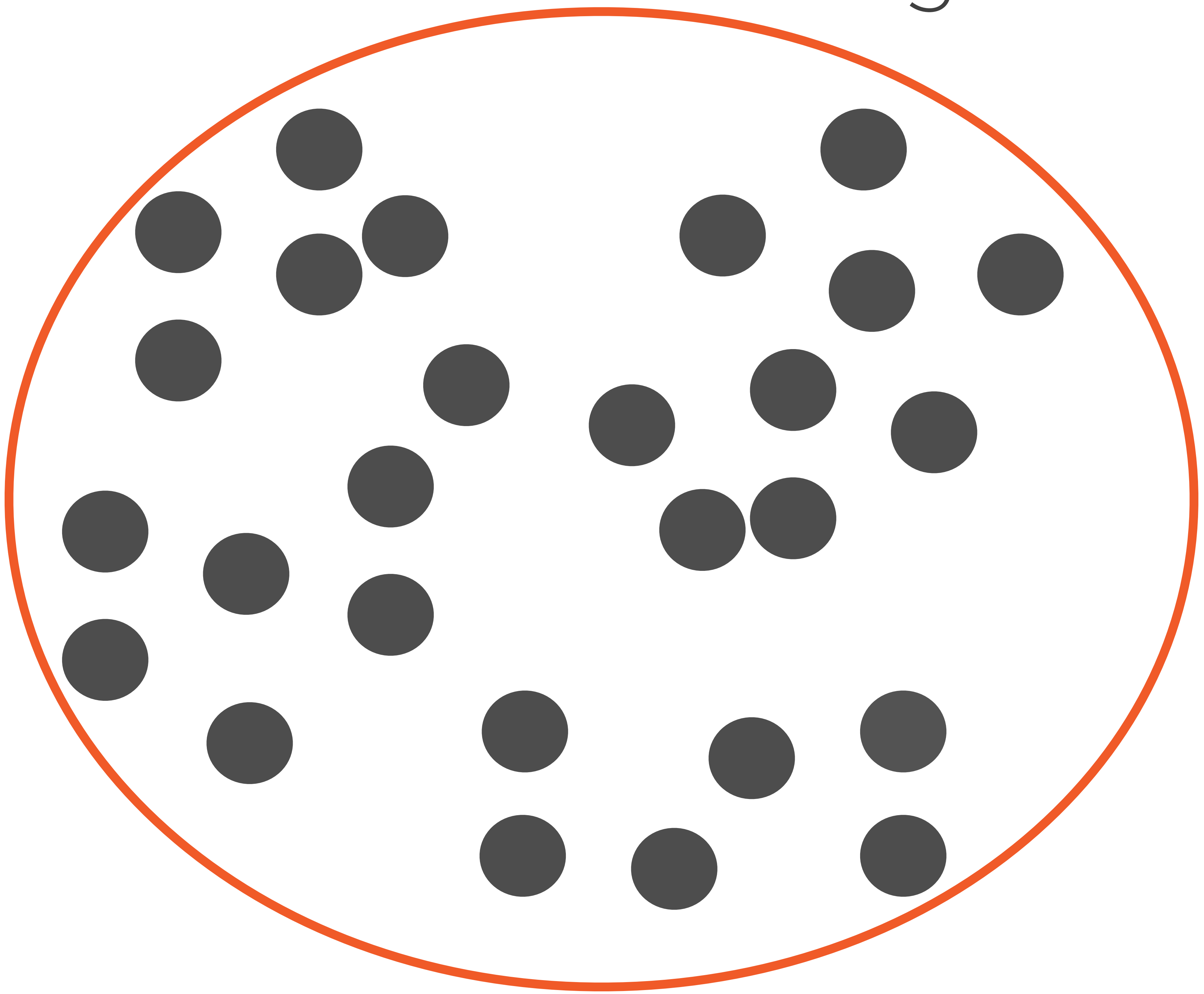
- Design choice #1:
  - Distance measure between point, cluster
  - Euclidean distance often used

```
Initialise K centroids

Repeat:
  For each data point:
    Assign to "nearest" cluster

  For each centroid:
    Update coordinates

Have centroids converged?
   Yes: Stop, we're done
    No: Keep iterating
```

- Design choice #2:
  - Calculating cluster center from points in cluster
  - Centroid (simple average) often used

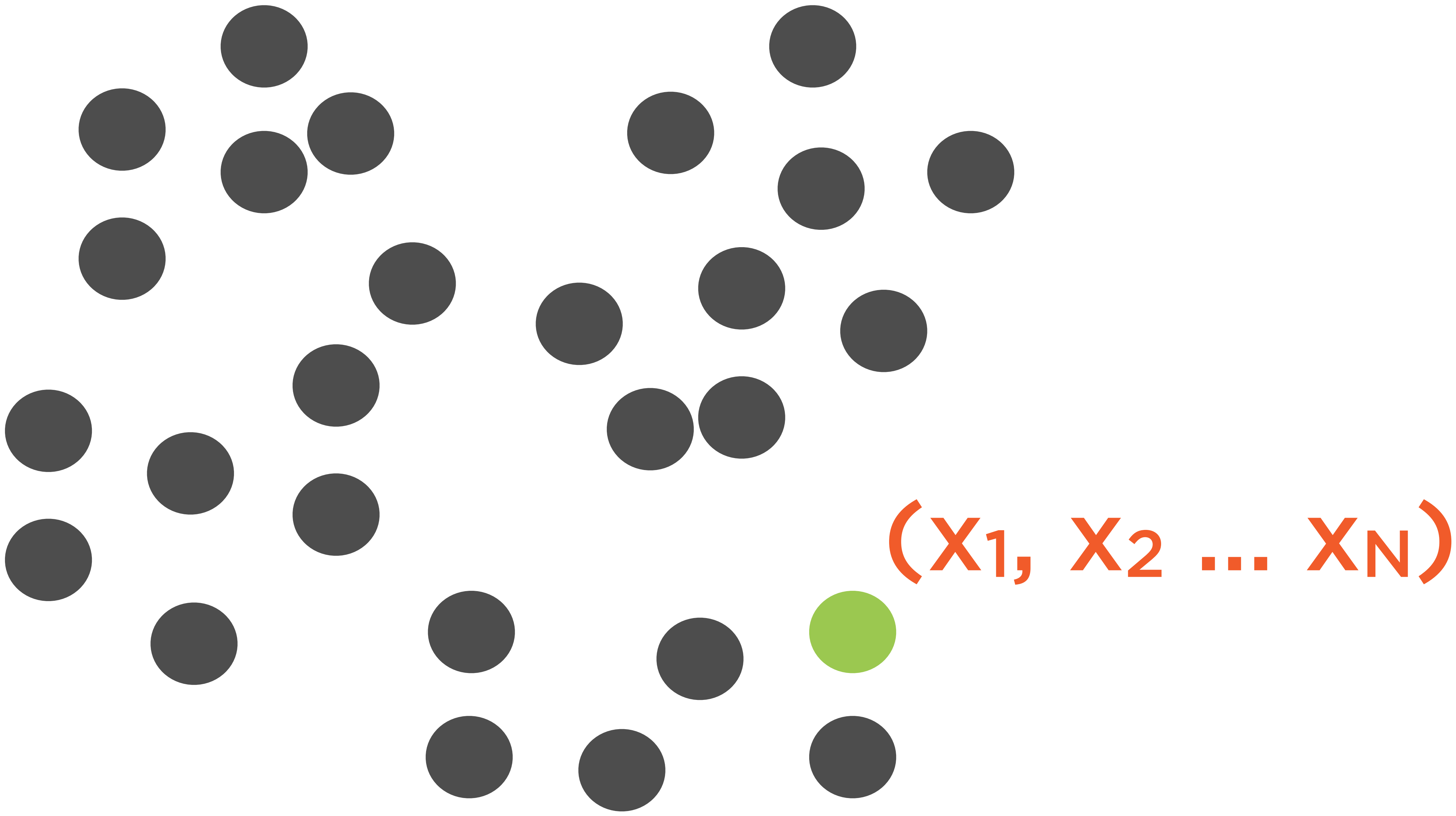
K-means Clustering

Given t data points



K-means Clustering

Each in N dimensional space

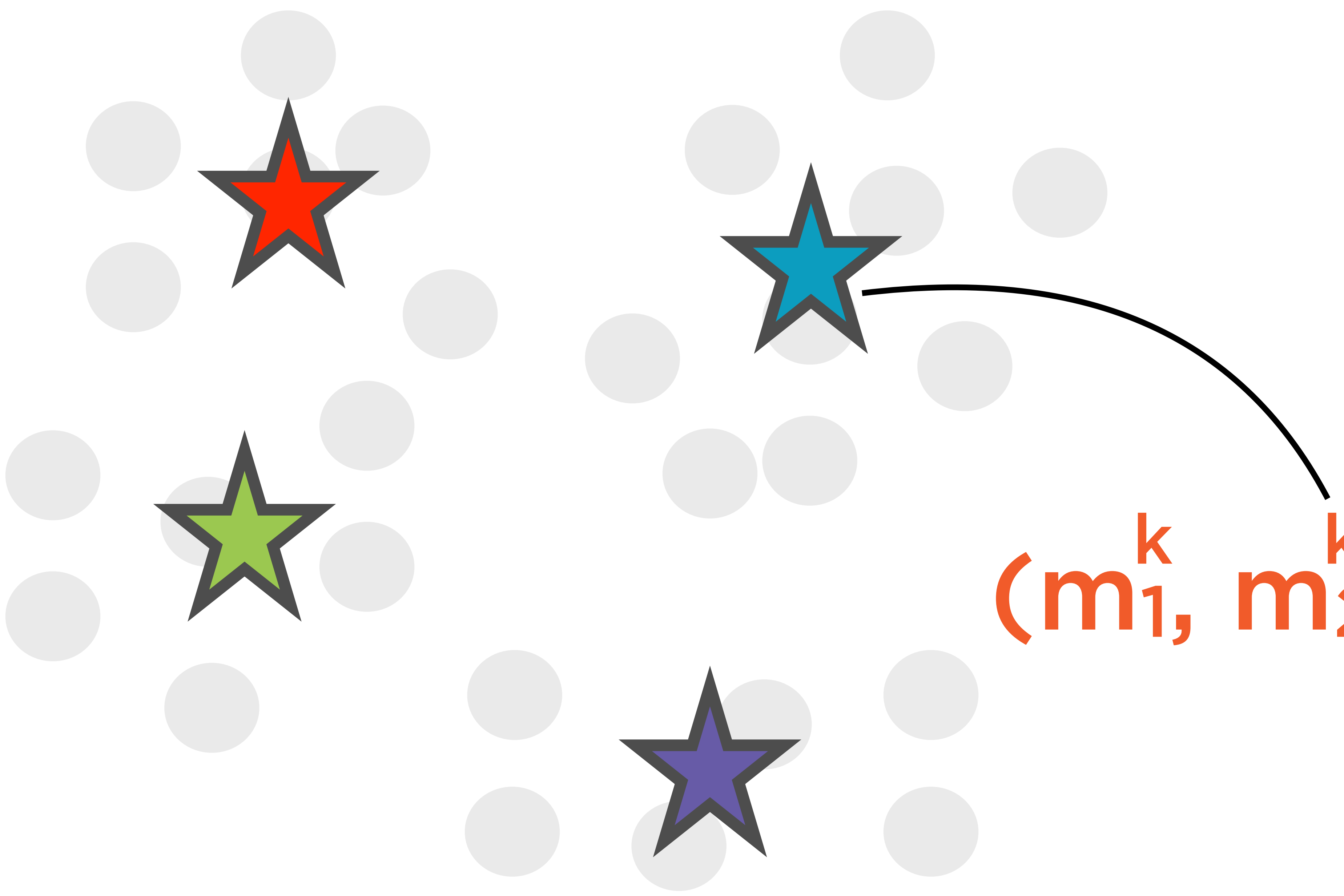


K Reference Vectors

Find K "best" reference vectors

$(m^1, \ldots\ m^K)$

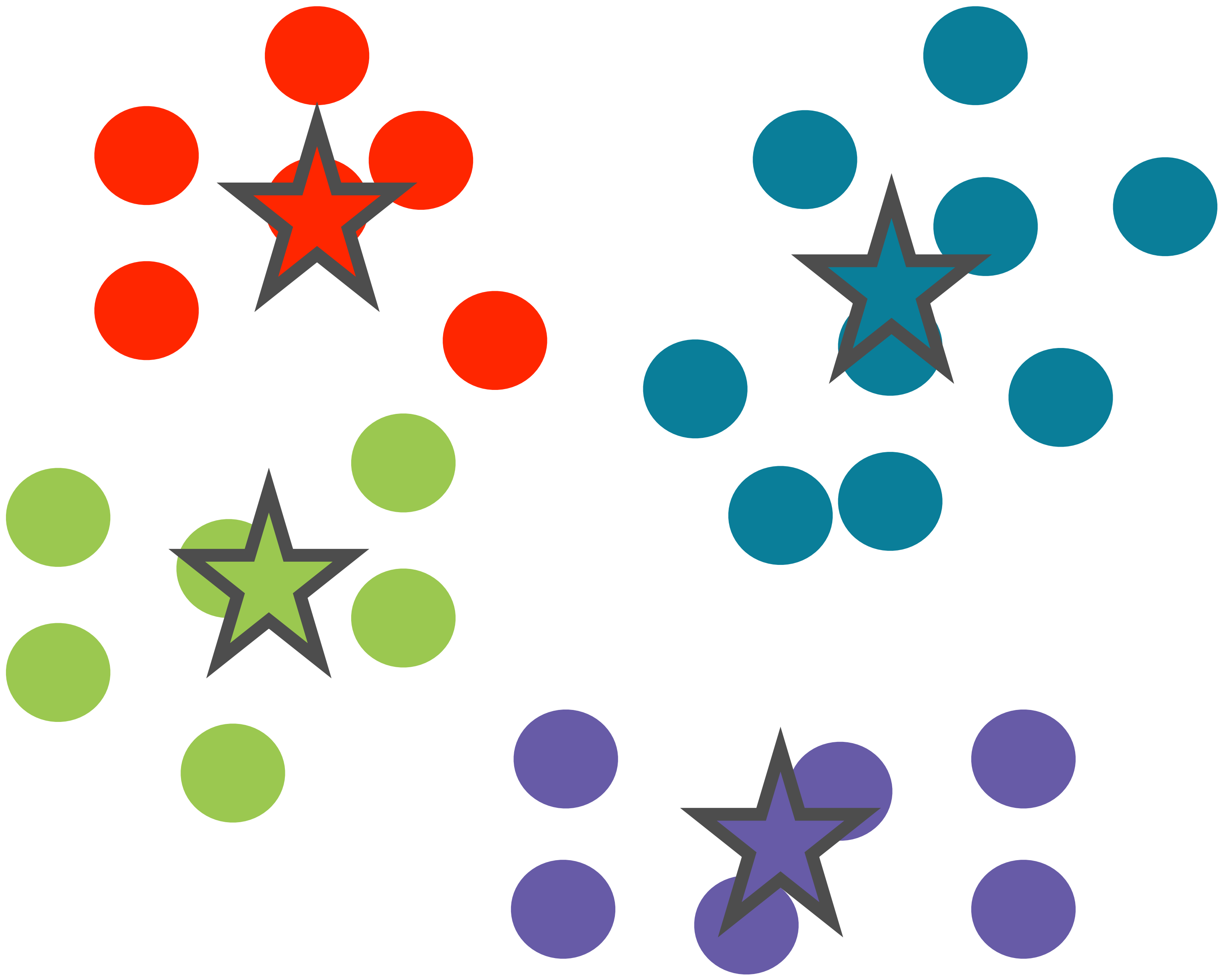
Here K = 4



$(m_1^k, m_2^k \ldots m_N^k)$

Assign Points to Cluster

Each point is associated with exactly one cluster

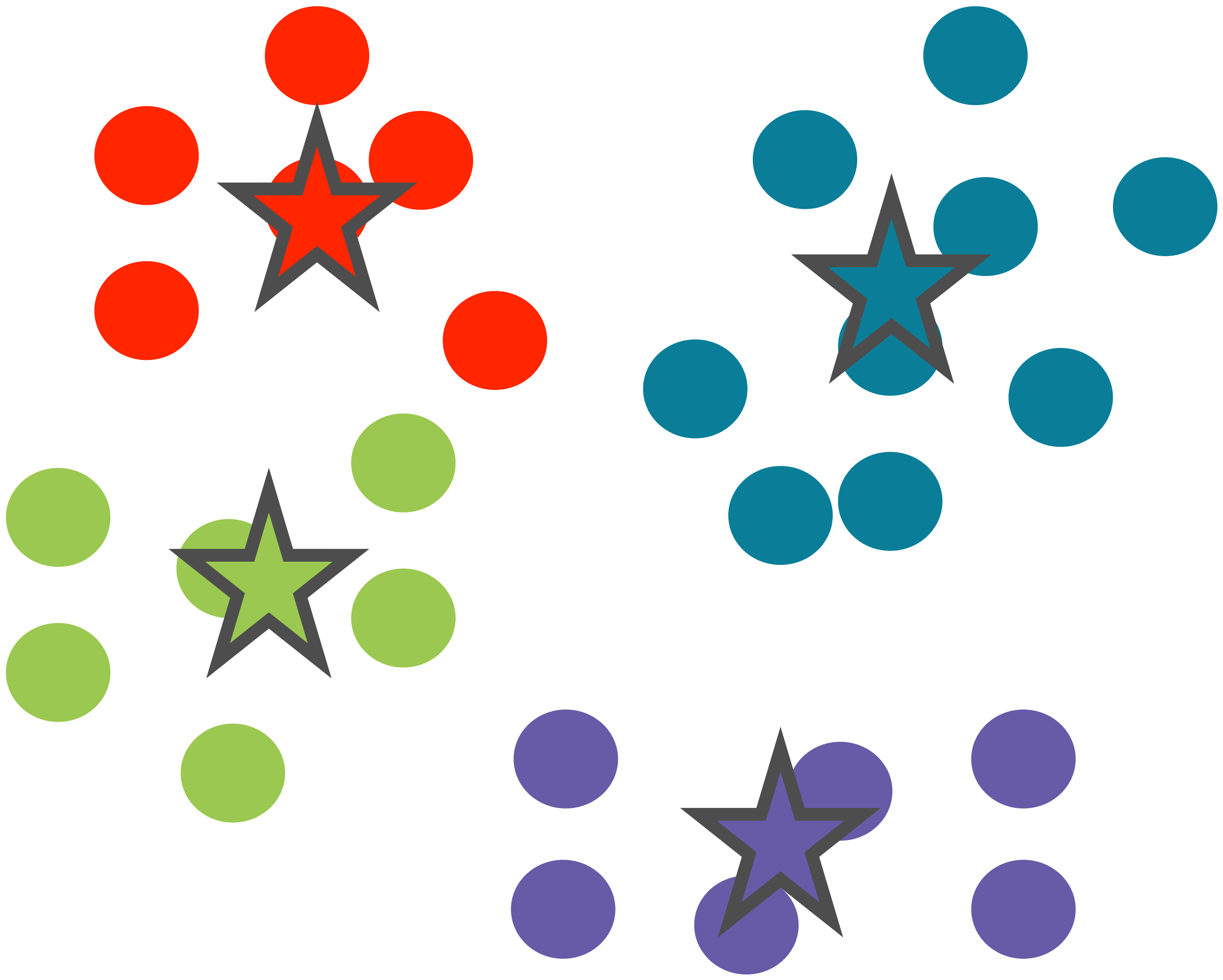


```
Initialise K centroids


Repeat:
  For each data point:
    Assign to "nearest" cluster


  For each centroid:
    Update coordinates

Have centroids converged?
   Yes: Stop, we're done
    No: Keep iterating
```

- Design choice #1:
  - Distance measure between point, cluster
  - Euclidean distance often used

Reference Vector as Centroid

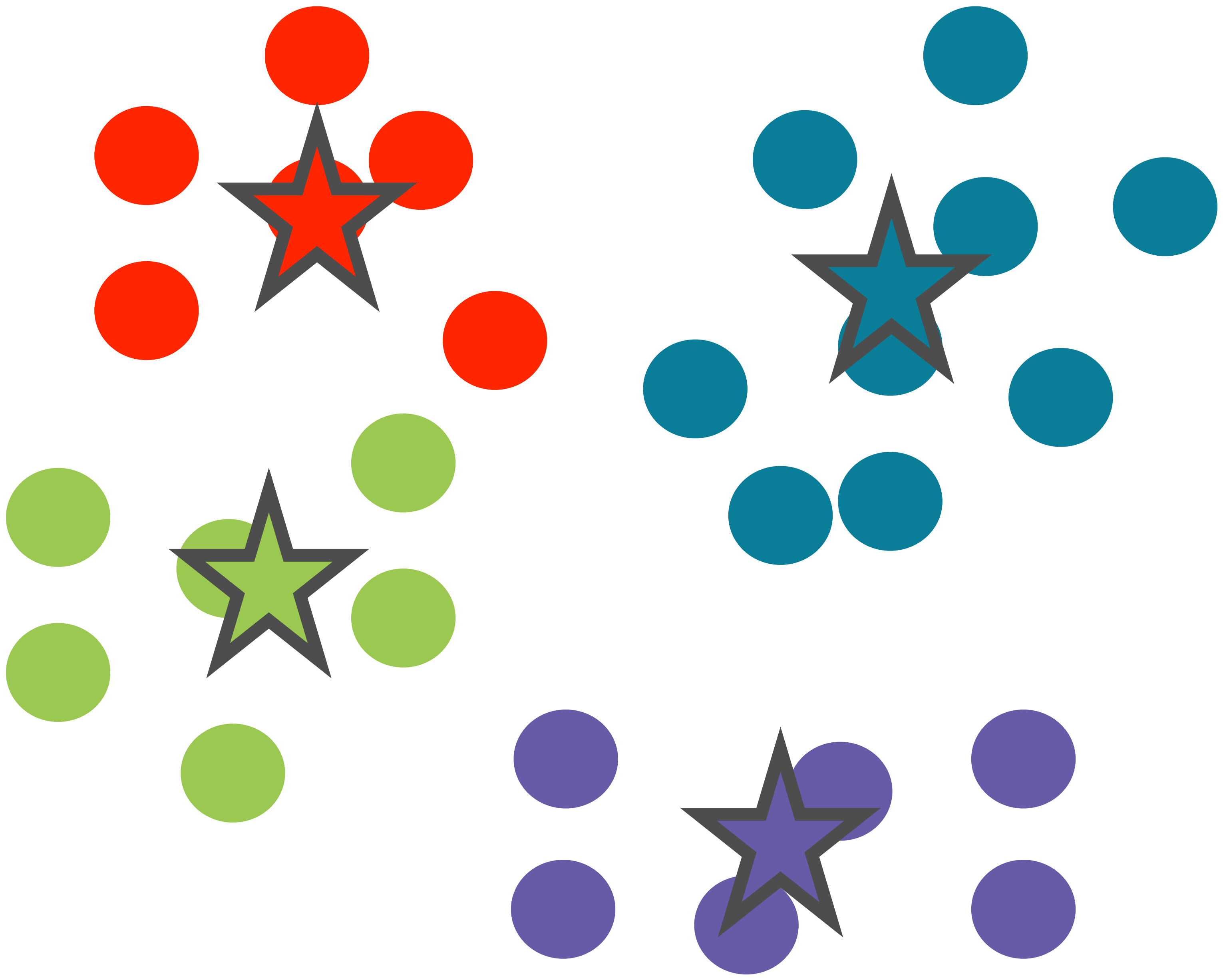
Each reference vector is the average of all points in that cluster



```
Initialise K centroids

Repeat:
  For each data point:
    Assign to "nearest" cluster

  For each centroid:
    Update coordinates

Have centroids converged?
   Yes: Stop, we're done
    No: Keep iterating
```

- Design choice #2:
  - Calculating cluster center from points in cluster
  - Centroid (simple average) often used

Total Reconstruction Error



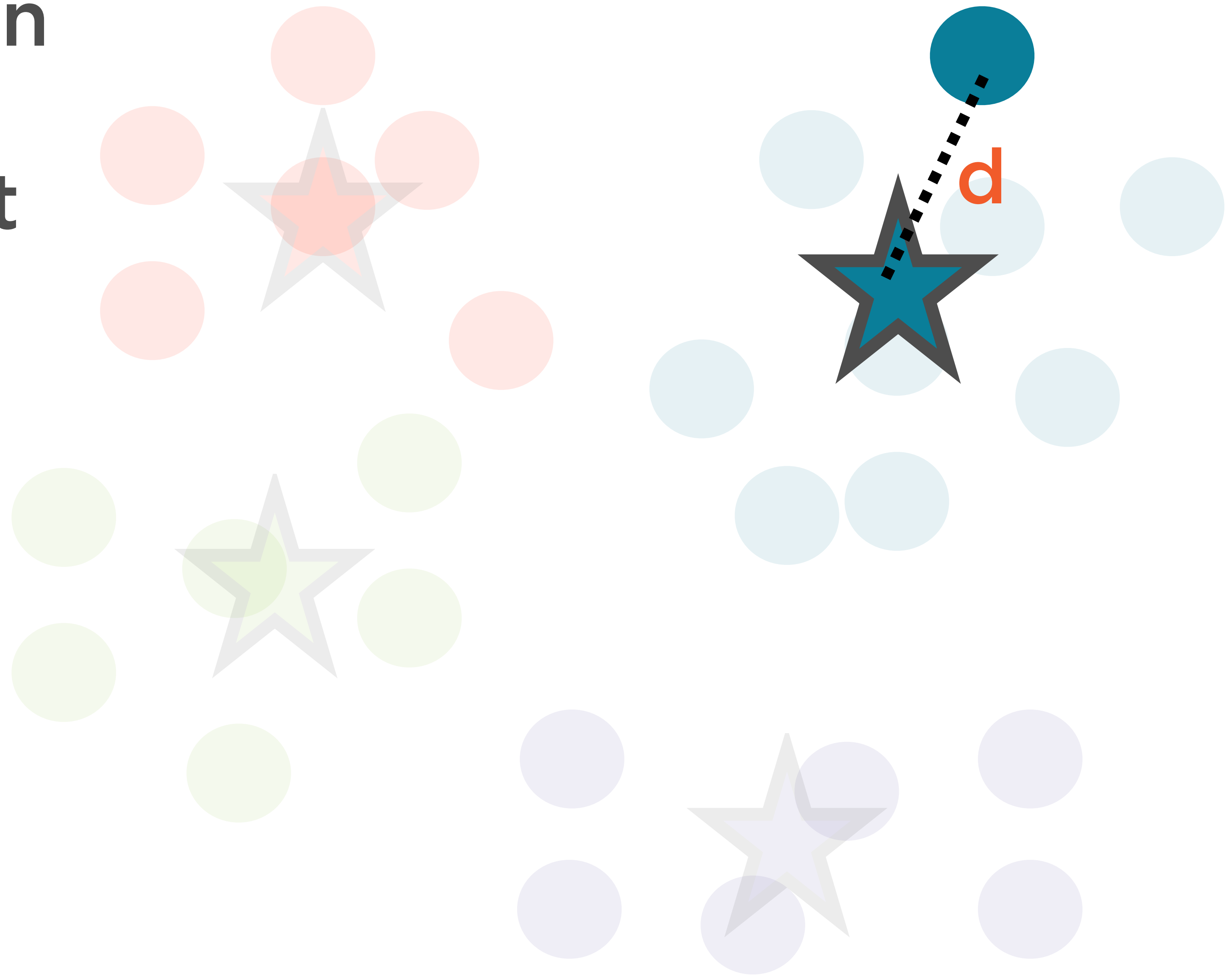
...Minimizing total reconstruction error

The lower the total reconstruction error, the better the fit

Individual Reconstruction Error

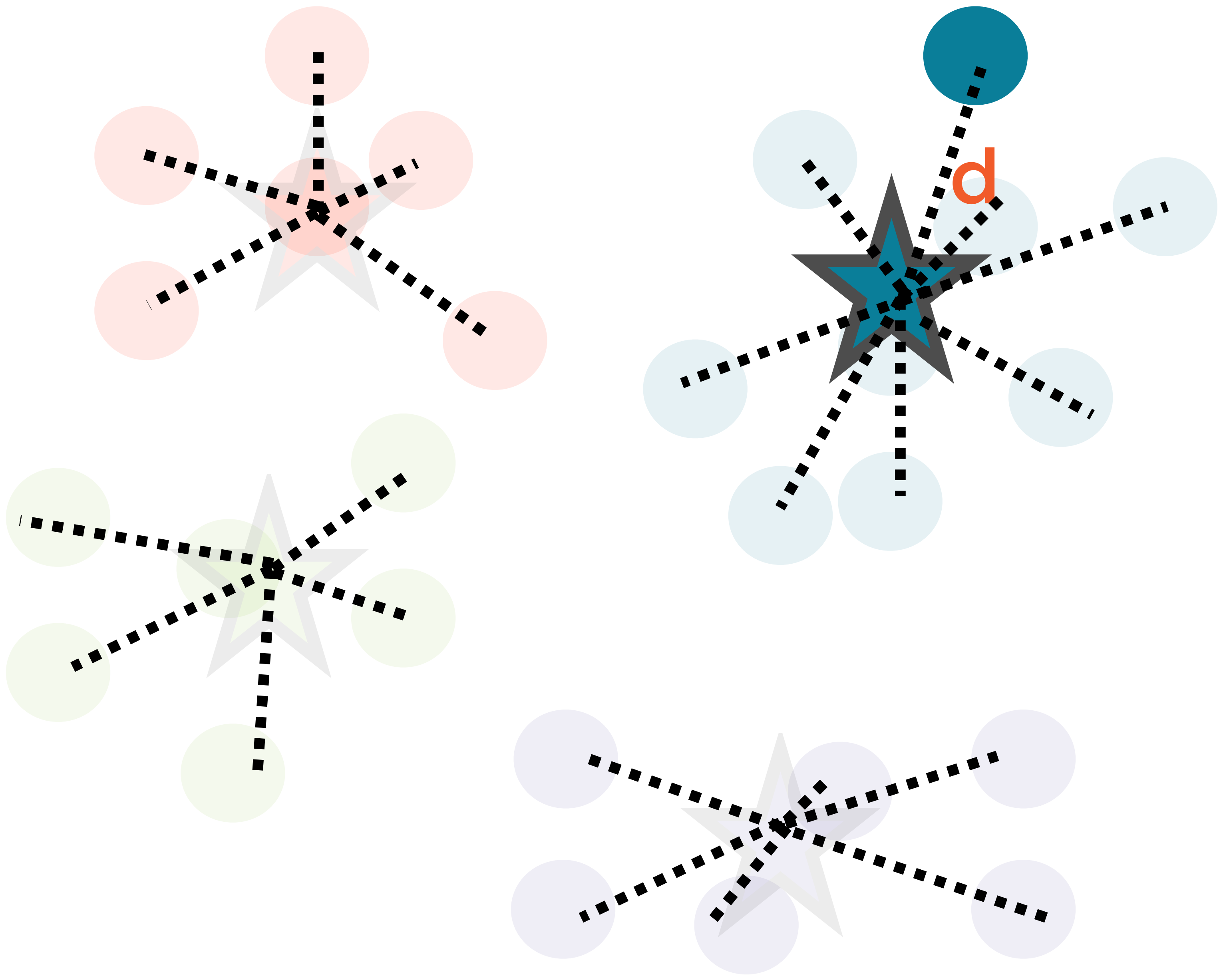
Square of Euclidean distance of each point from nearest reference vector

d^2



Total Reconstruction Error

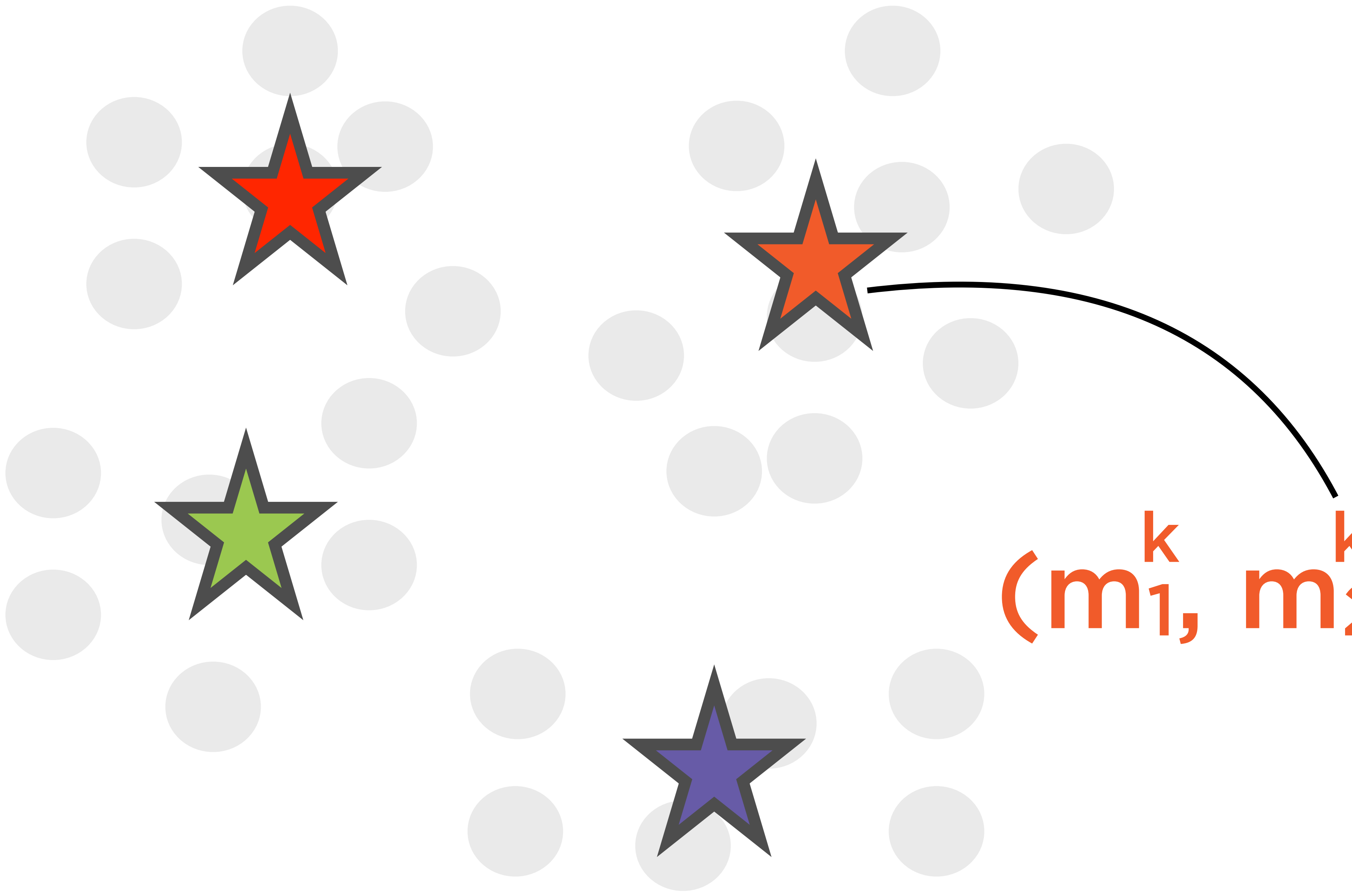
Sum over all points and reference vectors



Initial Values Matter

Pick initial values as smartly as possible

$(m^1, \ldots m^K)$



$(m_1^k, m_2^k \ldots m_N^k)$

Evaluating Clustering Models

Evaluating Clustering Models

Homogeneity	Completeness	V-measure
Adjusted Rand Index (ARI)	Adjusted Mutual Info	Silhouette

Evaluating Clustering Models

Homogeneity	Completeness	V-measure
Adjusted Rand Index (ARI)	Adjusted Mutual Info	Silhouette

Desirable Properties of Clusters

Homogeneity

Each cluster should contain members of the same class

Completeness

All members of a class should lie in the same cluster

Homogeneity vs. Completeness

Homogeneity and completeness are inversely related

Each lies between 0 and 1

Similar to precision and recall

Need a metric to optimize trade-off

V-measure

$$2 \times \frac{\text{Homogeneity} \times \text{Completeness}}{\text{Homogeneity} \times \text{Completeness}}$$

Harmonic mean of homogeneity and completeness

Closer to lower of two

Favors even weightage to both metrics

Homogeneity, Completeness, V-measure

Related set of metrics

Bounded scores between 0 and 1

Easy to interpret - higher is better

Apply to any algorithm

However, require labeled data

Evaluating Clustering Models

Homogeneity	Completeness	V-measure
Adjusted Rand Index (ARI)	Adjusted Mutual Info	Silhouette

Adjusted Rand Index

Measure of similarity between labels and assigned clusters

Also needs labeled data

Adjusts for probability of correct labeling by chance

Named after William Rand

Adjusted Rand Index

Value between -1 and 1 in sklearn

1 indicates that labels and calculated clusters agree perfectly for all points

0 or negative values are bad

Indicate that labels and calculated clusters are independent

Evaluating Clustering Models

Homogeneity	Completeness	V-measure
Adjusted Rand Index (ARI)	Adjusted Mutual Info	Silhouette

Adjusted Mutual Info

Measures mutual information in overlap between cluster assignments

Also needs labeled data

1 indicates highest mutual information, i.e. best clustering

0 or negative values are bad

Indicate that labels and calculated clusters are independent

Evaluating Clustering Models

Homogeneity	Completeness	V-measure
Adjusted Rand Index (ARI)	Adjusted Mutual Info	Silhouette

Important advantage of Silhouette scoring: Does not require labeled data

Silhouette Score

Defines Silhouette coefficient for each sample

Measure of how similar an object is to objects in its own cluster

And how different it is from objects in other clusters

Overall Silhouette score averages Silhouette coefficient of each sample

No need for labeled data

Demo

Implementing K-means clustering

Demo

Implementing K-means clustering on the Iris dataset

Summary

Clustering as a classic ML problem

Solving clustering using unsupervised learning

Setting up the clustering problem

Solving the clustering problem using K-means clustering