Machine Learning

Deniz Gündüz and Krystian Mikolajczyk

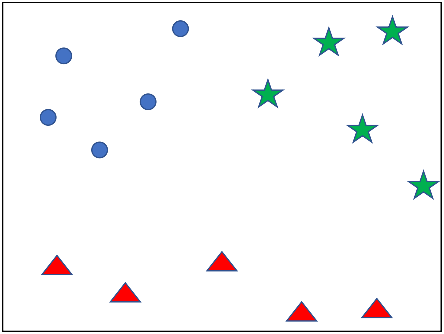
Department of Electrical and Electronic Engineering
Imperial College London

Machine Learning

- Supervised learning: Given data samples with labels $(\boldsymbol{x}, y)$, we want to learn a function $y = f(\boldsymbol{x})$ to predict labels of new samples
  - Classification: y is discrete
  - Regression: y is continuous
- Unsupervised learning: We are given only samples of data X, we want to compute a function $y = f(\boldsymbol{x})$ that provides a simpler representation
  - y is discrete: Clustering
  - y is continuous: Matrix factorization, autoencoders, Kalman filtering

Clustering

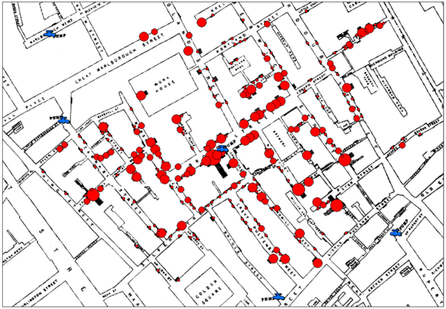
- Goal is to group ‘similar’ items into clusters



- We want
  - the items in the same cluster to be similar
  - items in different cluster to be dissimilar

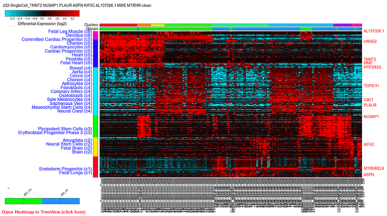
Early Application

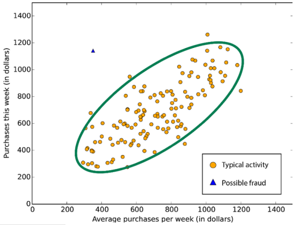
- John Snow (1813-1858), a London physician, created a map showing the deaths caused by a cholera outbreak in Soho and the locations of water pumps in the area.
- Observed that deaths were clustered around certain pumps
- Removing the handles stopped the deaths

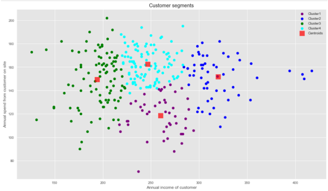


Applications of Clustering

- Anomaly detection (identifying fake news, spam detection, etc.)
- Marketing (cluster customers, products, etc.)
- Gene clustering



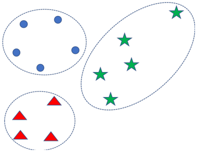




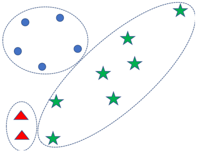
How to measure 'similarity' ?

- We want a function that assigns a real number to every pair of two samples from the space
- Function value should increase with dissimilarity of the objects
- For example:
  - Eucledian distance: $d(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{i=1}(x_i - y_i)^2}$
  - Correlation coefficient: $d(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=1}(x_i - \mu_x)(y_i - \mu_y)}{\sigma_x \sigma_y}$

How to evaluate clusters?



(a) good cluster



(b) bad cluster

- Intra-cluster cohesion (compactness)
  - How close the samples in a cluster to the cluster center
- Inter-cluster separation (isolation):
  - How far (dissimilar) different cluster centroids from one another.
- In most case we depend on expert judgement

Complexity of Clustering

- Consider n data points and k clusters
- Associate each data point with one cluster
- Brute-force method:
  - Write down all possible clusterings
  - Associate a score for each
  - Choose the best scoring one
- Not more than k^n clusterings
- Permutations must be discounted: $O(k^n/k!)$
- Number of ways n objects can be partitioned into k non-empty and disjoint parts is given by Stirling numbers of second kind:

$$\frac{1}{k!}\sum_{t=0}^{k}(-1)^t\binom{k}{t}(k-t)^n$$

- Not practically feasible

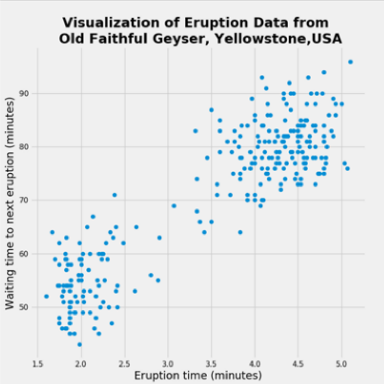
k-Means Clustering

- n data points: $\mathcal{D} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}, \ \boldsymbol{x}_i \in \mathcal{X}$
- k clusters: $\mathcal{C} = \{\mathcal{C}_1, \dots, \mathcal{C}_k\}$
- Assign a representative to each cluster: $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_k, \ \boldsymbol{\mu}_j \in \mathcal{X}$
- Score function

$$J(\mathcal{C}) = \sum_{i=1}^{k} \sum_{\boldsymbol{x}_j \in \mathcal{C}_i} \|\boldsymbol{x}_j - \boldsymbol{\mu}_k\|^2$$

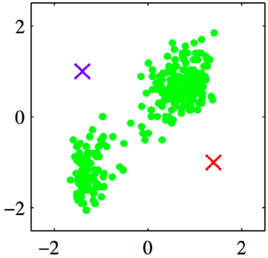
k-Means Clustering

A greedy iterative algorithm to decide the clusters and cluster centers



k-Means Clustering

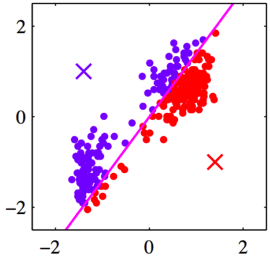
- Standardise data (s.t. each variable has zero-mean and unit standard deviation)
- Choose two cluster centers



C. Bishop, Pattern Recognition and Machine Learning, Figure 9.1.

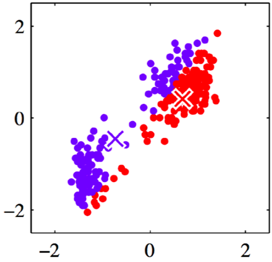
k-Means Clustering

- Assign each data point to one of the two clusters
- Equivalent to classification of data samples with the perpendicular bisector of the two cluster centers



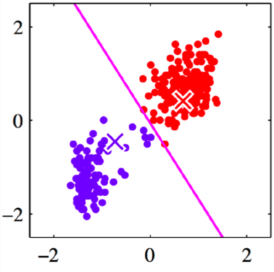
k-Means Clustering

- Recompute cluster centers



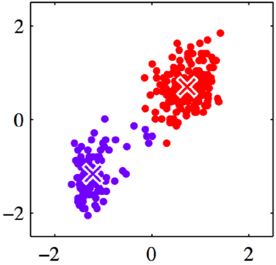
k-Means Clustering

- Assign points to the closest center



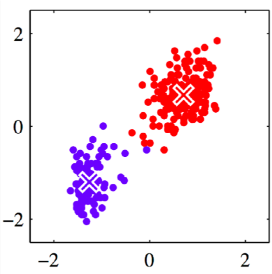
k-Means Clustering

- Compute cluster centers



k-Means Clustering

- Follow the same steps until convergence



k-Means Algorithm: Formal Description

- Let $\mathcal{X}$ be a space with some distance metric d.
  (e.g., $\mathcal{X} = \mathcal{R}^d$ and $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|$)
- Dataset: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\},\ \boldsymbol{x}_i \in \mathcal{X}$
- We want to create k clusters $\mathcal{C}_1, \ldots, \mathcal{C}_k$
- The cluster center of $\mathcal{C}_i$ is given by

$$\boldsymbol{\mu}_i = \boldsymbol{\mu}(\mathcal{C}_i) = \underset{\boldsymbol{\mu} \in \mathcal{X}}{\operatorname{argmin}} \sum_{\boldsymbol{x} \in \mathcal{C}_i} d(\boldsymbol{x}, \boldsymbol{\mu})$$

- For Euclidean distance, $\boldsymbol{\mu}_i$ is simply the average of the samples in $\mathcal{C}_i$

k-Means Algorithm: Formal Description

Consider $\mathcal{X} = \mathbb{R}^d$ and $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2$

1. Initialize cluster centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k \in \mathbb{R}^d$ randomly
2. Repeat until convergence
   - For $i = 1, \ldots, n$, set

$$c_i = \operatorname*{argmin}_j d(\boldsymbol{x}_i, \boldsymbol{\mu}_j)$$

   - For $j = 1, \ldots, k$, set

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^n \mathbb{1}\{c_i = j\}\boldsymbol{x}_i}{\sum_{i=1}^n \mathbb{1}\{c_i = j\}}$$

Convergence of k-Means Algorithm

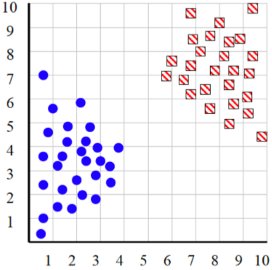
Objective of k-means algorithm:

$$J(\mathcal{C}_1, \ldots, \mathcal{C}_k) = \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in \mathcal{C}_i} d(\boldsymbol{x}, \boldsymbol{\mu}_i) = \sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{\mu}_{c_i}\|^2$$

- In each iteration, we keep $\boldsymbol{\mu}_i$'s fixed and minimize J with respect to c_i's, then fix c_i's and minimize J with respect to $\boldsymbol{\mu}_i$'s
- J monotonically decreases, hence must converge
- In theory, it can oscillate between two or more clusterings (with the same J value (almost never happens in practice)
- J is non-convex, so not guaranteed to converge to a global minimum
- Recommendation: run several times with different initialization, and pick the result with the smallest objective function

How to Choose the Right Number of Clusters

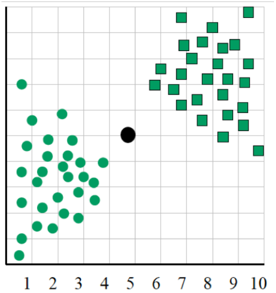
- In general, we don't know the answer
- Consider the following dataset



Example from T. Mitchell, 10601, Fall 2012.

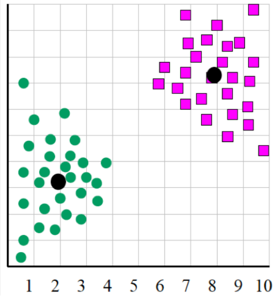
How to Choose the Right Number of Clusters

- For $k = 1$, the objective function is 873



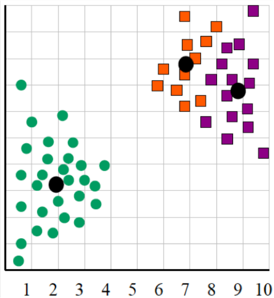
How to Choose the Right Number of Clusters

- For $k = 2$, the objective function is 173.1



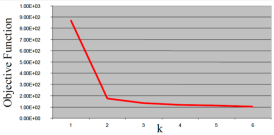
How to Choose the Right Number of Clusters

- For $k = 3$, the objective function is 133.6



How to Choose the Right Number of Clusters

- If we plot the objective function for $k = 1, 2, \ldots, 6$



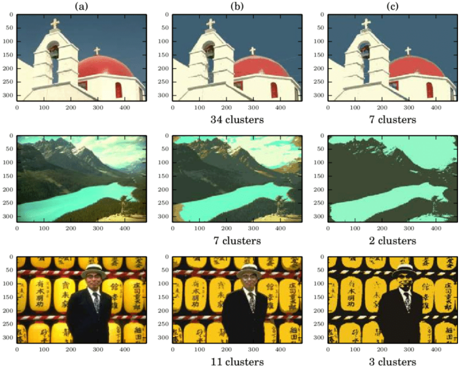
- Abrupt change at $k = 2$ suggests presence of two clusters in the data
- This technique for determining the number of clusters is known as "knee finding" or "elbow finding"
- Not always as clear as this example!

Applications of k-Means Algorithm

- Clustering can be considered as lossy data compression: *vector quantization*
- All points in a cluster are represented by the cluster center, introducing distortion
- Cluster centers represent the compression codebook
- The objective function corresponds to the reconstruction error
- We need $\log_2(k)$ bits to represent all the clusters
- We obtain a *rate-distortion function*

Image Segmentation

- Partition image into region such that each region corresponds to a distinct object or parts of an object
- Treat each pixel as a data point



T. Santos, SciPy and OpenCV as an interactive computing environment for computer vision.