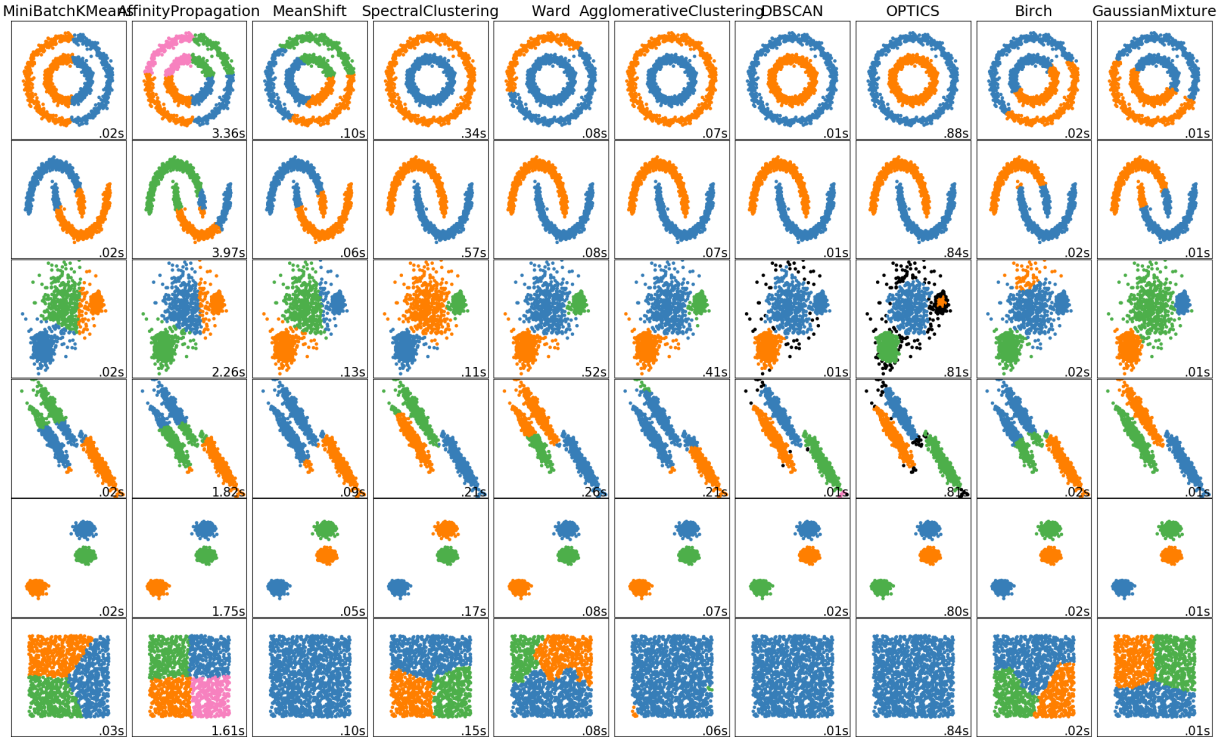
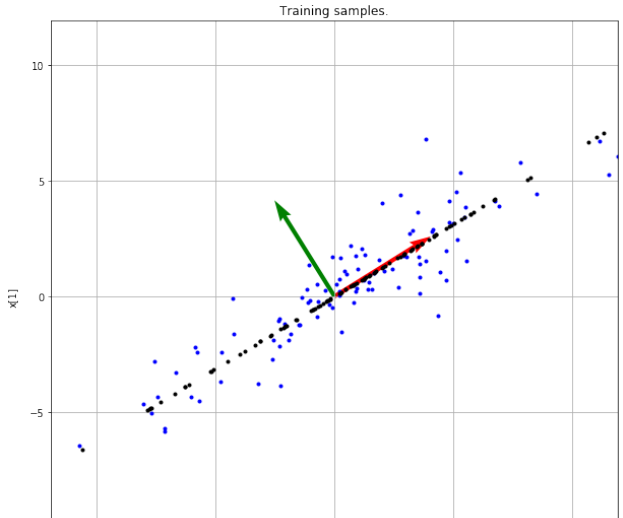
Unsupervised Learning
Clustering and Dimensionality Reduction

Nidhin Koshy Vaidhiyan

Consultant Technologist
Centre for Networked Intelligence, IISc Bangalore
16 January 2020

Find the clusters





Unsupervised Learning

- No labels associated with each datapoint.

Unsupervised Learning

- No labels associated with each datapoint.
- Can we infer useful information from available data?

Unsupervised Learning

- No labels associated with each datapoint.
- Can we infer useful information from available data?
- Two main subclasses of problems
  - Clustering
  - Dimensionality Reduction

Outline

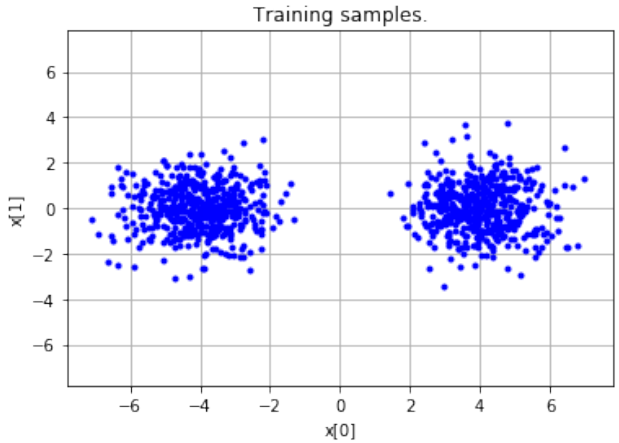
Clustering

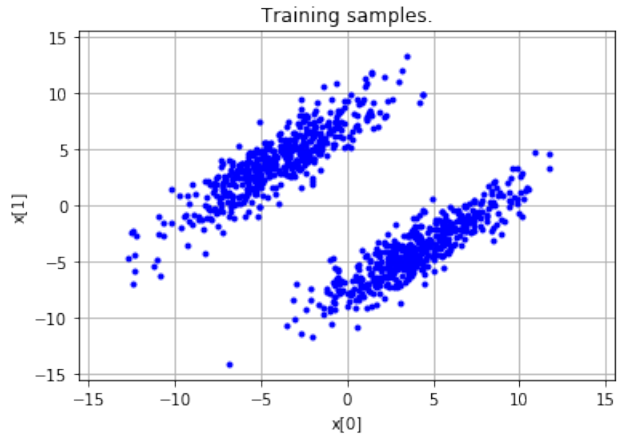
- K-means
- Expectation - Maximisation (EM) algorithm
- Spectral Clustering

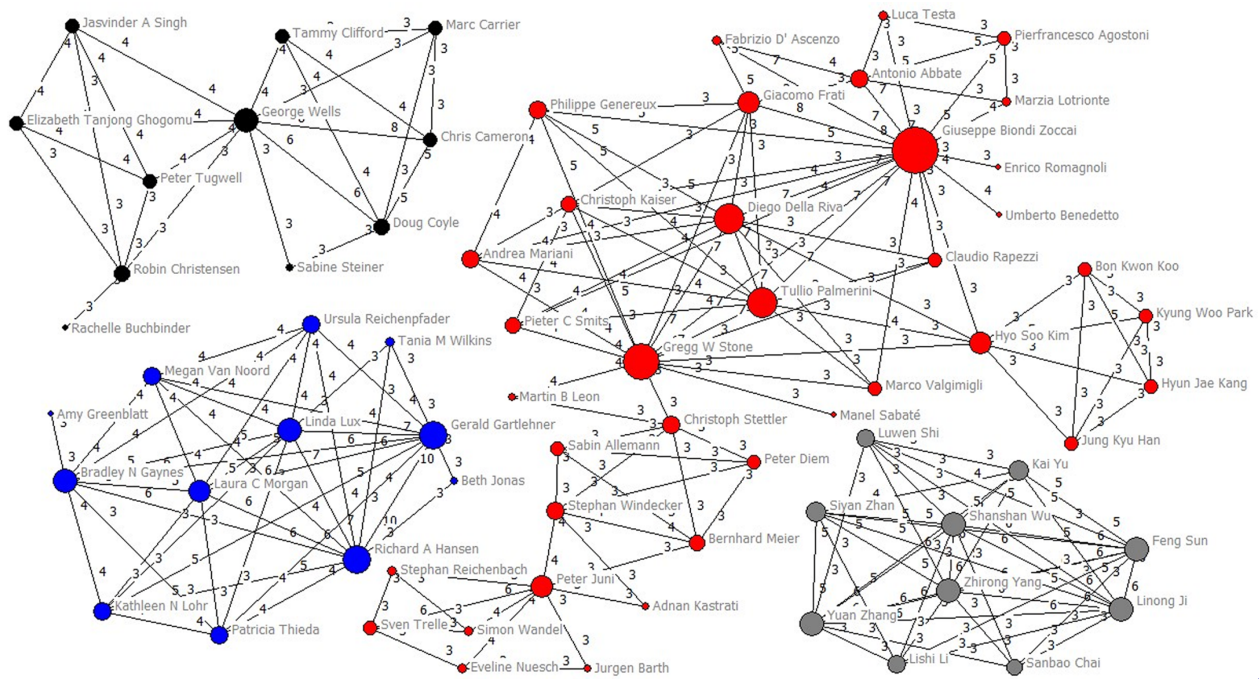
PCA for Dimensionality Reduction

Clustering

- Loose Definition: Given a set of points, can we create groups such that elements in a group are similar in some sense.







Clustering - Applications

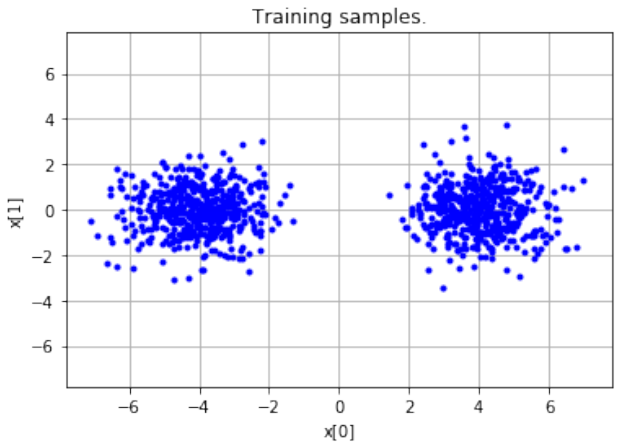
- Community detection in social networks.
- Targeted Marketing.

Clustering algorithm

- K-means algorithm
- Expectation Maximisation - A generalisation of K-means
- Spectral Clustering

K-means algorithm

- A simple algorithm for clustering.
- Quite effective when the clusters are well separated and are isotropic.



Notation

Let us build some essential notation.

- Let n denote the total number of datapoints.

Notation

Let us build some essential notation.

- Let n denote the total number of datapoints.
- Let K denote the number of clusters.

Notation

Let us build some essential notation.

- Let n denote the total number of datapoints.
- Let K denote the number of clusters.
- Let $x_i \in \mathcal{R}^d$ be a datapoint, for $i = 0, 1, \ldots, n-1$.

Notation

Let us build some essential notation.

- Let n denote the total number of datapoints.
- Let K denote the number of clusters.
- Let $x_i \in \mathcal{R}^d$ be a datapoint, for $i = 0, 1, \ldots, n-1$.
- Let *datapoints* denote the set $\{x_0, x_1, \ldots, x_{n-1}\}$

Notation

Let us build some essential notation.

- Let n denote the total number of datapoints.
- Let K denote the number of clusters.
- Let $x_i \in \mathcal{R}^d$ be a datapoint, for $i = 0, 1, \ldots, n-1$.
- Let *datapoints* denote the set $\{x_0, x_1, \ldots, x_{n-1}\}$
- Let $labels = (labels_0, labels_1, \ldots, labels_{(n-1)})$ denote the cluster labels for each datapoint, where $label_i \in \{0, 1, \ldots, (K-1)\}$.

Notation

Let us build some essential notation.

- Let n denote the total number of datapoints.
- Let K denote the number of clusters.
- Let $x_i \in \mathcal{R}^d$ be a datapoint, for $i = 0, 1, \ldots, n-1$.
- Let *datapoints* denote the set $\{x_0, x_1, \ldots, x_{n-1}\}$
- Let $labels = (labels_0, labels_1, \ldots, labels_{(n-1)})$ denote the cluster labels for each datapoint, where $label_i \in \{0, 1, \ldots, (K-1)\}$.
- Let $\mu = (\mu_0, \mu_1, \ldots, \mu_{K-1})$ denote the cluster centroids, where $\mu_i \in \mathcal{R}^d$, for $i = 0, 1, \ldots, (K-1)$

Notation-functions

1 assign_labels($datapoints, \mu, K$)

1: **for** i in $range(n)$ **do**
2: $\quad labels_i = \arg\min_{j=0}^{K-1} \|x_i - \mu_j\|^2$
3: **end for**
4: **return** labels

2 update_centroids($datapoints, labels, K$)

1: **for** j in $range(K)$ **do**
2: $\quad \mu_j = \frac{\sum_{i=0}^{n-1} x_i . 1_{\{label_i == j\}}}{\sum_{i=0}^{n-1} 1_{\{label_i == j\}}}$
3: **end for**
4: **return** μ

K-means algorithm

3 K-means($datapoints, K$)

1: $converged$ = FALSE
2: μ = random_initialisation_centroids(K, $datapoints$)
3: **while** (not $converged$) **do**
4: $labels$ = assign_labels($datapoints$, μ)
5: μ_{new} = update_centroids($datapoints$, $labels$)
6: **if** ($\mu_{new} == \mu$) **then**
7: $converged$ = TRUE
8: **end if**
9: $\mu = \mu_{new}$
10: **end while**
11: **return** $labels$, μ

K-means algorithm - HANDS-ON 01

- Exercise - Try K-means for $K = 2$ clusters
- Exercise - Keeping the actual number of clusters to 2, increase K while calling K-means
- Qn - If prior knowledge of K is not available, how to choose K?

K-means algorithm as a solution to an Optimisation Problem

- K-means algorithm tries to minimise the intra-cluster variances.
- Let

$$J(datapoints, centroids, labels) := \sum_{i=0}^{n} \|x_i - \mu_{labels_i}\|^2 \quad (1)$$

- K-means tries to minimise:

$$\min_{centroids, labels} J(datapoints, centroids, labels) \quad (2)$$

- The above optimisation problem is a non-convex optimisation problem (why?). K-means is a co-ordinate descent procedure to solve the above optimisation problem.
  - Keeping the centroids fixed, it finds the optimal labels.
  - Keeping the labels fixed, it finds the optimal centroids. (Prove that centroid minimises the sum of squared distance for each cluster.)
- After every iteration, the cost fuction $J(.)$ decreases. Hence, $J(.)$ is guaranteed to converge to atleast a local minima.

Probabilistic Interpretation of K-means

- Assume that datapoints in all clusters are generated from a Gaussian distribution having a scaled Identity matrix as the covariance matrix, i.e., $\Sigma_j = \sigma^2 I, \quad j = 0, 1, \ldots, K-1$.
- However, the mean of the Gaussian distribution is different for different clusters.
- The log likelihood ratio in this scenario can be obtained as:

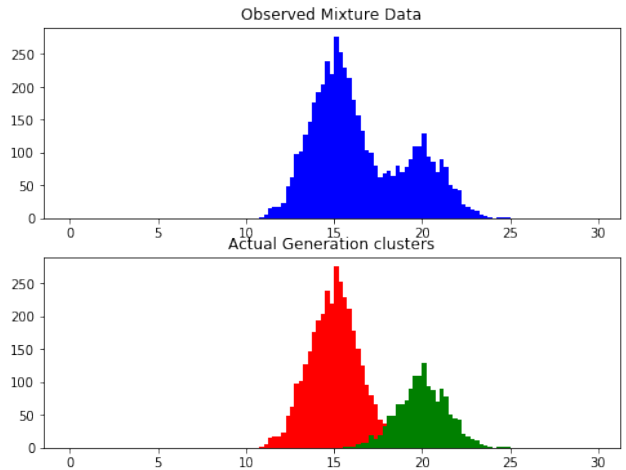
$$
\begin{aligned}
& L(datapoints | labels, centroids) \\
& := \sum_{i=0}^{n-1} \sum_{j=0}^{K-1} 1_{\{label_i = j\}} \log \left(\frac{1}{(2\pi\sigma^2)^{d/2}} exp^{\frac{-\|x_i - \mu_j\|^2}{2\sigma^2}} \right) \\
& = \sum_{i=0}^{n-1} \sum_{j=0}^{K-1} 1_{\{label_i = j\}} \left(\frac{-\|x_i - \mu_j\|^2}{2\sigma^2} - \frac{d}{2} \log(2\pi\sigma^2) \right) \\
& = -\frac{J(datapoints, centroids, labels)}{2\sigma^2} - constant
\end{aligned}
$$

Probabilistic Interpretation of K-means

- Thus, minimising $J(\textit{datapoints}, \textit{centroids}, \textit{labels})$ is same as maximising the likelihood function under the above mentioned assumptions.
- This now clearly brings about the disadvantages of K-means.
- K-means is a good algorithm only in the restricted regime mentioned above.
- Exercise: HANDS-ON. Generate samples violating the above assumptions and show that K-means does not perform well.

Expectation Maximisation (EM) as a generalisation of K-means

- EM is an algorithm that tries to find the maximum likelihood estimates of parameters (cluster centroids and covariance matrices in our case) from observations (datapoints in our case) in the presence of unobserved latent variables (labels in our case).
- Thus, EM can be seen as a generalisation of K-means. It tries to find the maximum likelihood estimates without putting restricting assumptions on the parameters.



Expectation Maximisation (EM)

Let us now take a closer look at the EM algorithm.
Let $Y^n = (Y_1, Y_2, ..., Y_n)$ be a set of observations.
Let θ denote the parameter we want to optimize to maximize the likelihood.
The maximum likelihood estimate θ^* is defind as:

$$\theta^* = \arg\max_\theta p(Y^n|\theta) = \arg\max_\theta \log(P(Y^n|\theta)) =: \arg\max_\theta L(\theta).$$

In many cases it might not be easy to find θ^* directly from the above expression. This is particularly the case when there are unobservable random variables that influence the sample generation.

Expectation Maximisation (EM)

Let us try to derive the EM algorithm.

$$L(\theta) := \log(P(Y^n|\theta)) \tag{3}$$

$$= \log\left(\sum_{z\in\mathcal{Z}} P(Y^n, Z|\theta)\right) \tag{4}$$

$$= \log\left(\sum_{z\in\mathcal{Z}} Q(Z, Y^n, \theta)\frac{P(Y^n, Z|\theta)}{Q(Z, Y^n, \theta)}\right) \tag{5}$$

$$\geq \sum_{z\in\mathcal{Z}} Q(Z, Y^n, \theta)\log\left(\frac{P(Y^n, Z|\theta)}{Q(Z, Y^n, \theta)}\right), \tag{6}$$

where Z is the latent variable or the hidden variable. $Q(Z, Y^n, \theta)$ is a pmf on $\mathcal{Z}$. In particular let us set $Q(Z, Y^n, \theta) = P(Z|Y^n, \theta)$.

Expectation Maximisation (EM)

Let us define

$$I(\theta, \theta') := \sum_{z \in \mathcal{Z}} P(Z|Y^n, \theta') \log \left(\frac{P(Y^n, Z|\theta)}{P(Z|Y^n, \theta')} \right).$$

Exercise: Show that $I(\theta, \theta) = L(\theta)$.

---

4 EM(Y^n, K)

1: Initialize θ_0 to a random value. Initialise $n = 0$.
2: **while** $I(\theta_n, \theta_{n+1})$ not converged **do**
3: E-Step : compute $P(Z|Y^n, \theta_{n-1})$
4: M-Step : $\theta_n := \arg\max_\theta I(\theta, \theta_{n-1})$
5: $n = n + 1$
6: **end while**
7: **return** θ_n

Expectation Maximisation (EM)

Properties:
$L(\theta_n) = l(\theta_n, \theta_n)$ is a monotonically increasing sequence and hence convereges. This is true because the following chain of inequalities hold:

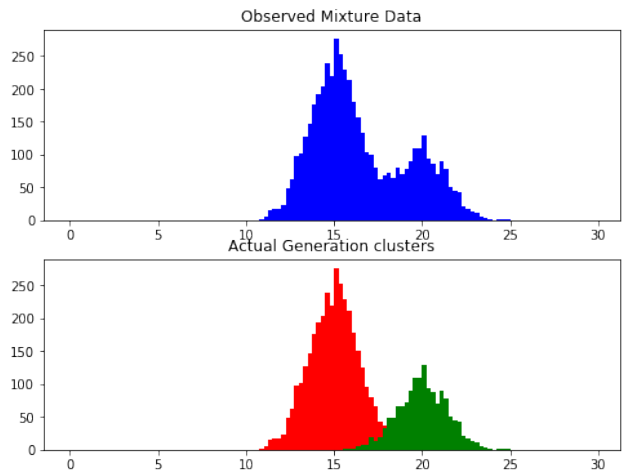
$$L(\theta_n) = l(\theta_n, \theta_n) \geq l(\theta_n, \theta_{n-1}) \geq l(\theta_{n-1}, \theta_{n-1}) = L(\theta_{n-1}).$$

Example - Mixture of Univariate Gaussian with unknown means and known variance.

Observation Model:

$$Y = 1_{\{C=0\}} X_1 + 1_{\{C=1\}} X_2,$$

where $C \sim Ber(p)$, X_1, X_2 are univariate Gaussian random variables with unknown means μ_1 and μ_2 and common variance σ^2.
To begin, let us assume that the variance is known.
Given samples from Y, the aim is to find the maximum likelihood estimates of μ_1 and μ_2. Here C is the latent variable or the hidden variable.



Example - Mixture of Univariate Gaussian with unknown means and known variance.

E-Step : Compute

$$w_i^0 := \frac{e^{\frac{-(Y_i-\mu_1)^2}{2\sigma^2}}(1-p)}{e^{\frac{-(Y_i-\mu_1)^2}{2\sigma^2}}(1-p) + e^{\frac{-(Y_i-\mu_2)^2}{2\sigma^2}}(p)} \tag{7}$$

$$w_i^1 := \frac{e^{\frac{-(Y_i-\mu_2)^2}{2\sigma^2}}(p)}{e^{\frac{-(Y_i-\mu_1)^2}{2\sigma^2}}(1-p) + e^{\frac{-(Y_i-\mu_2)^2}{2\sigma^2}}(p)} \tag{8}$$

$$\tag{9}$$

M-Step : Update μ_1 and μ_2 as

$$\mu_1 = \frac{\sum_{i=1}^n w_i^0 Y_i}{\sum_{i=1}^n w_i^0}, \text{ and} \tag{10}$$

$$\mu_2 = \frac{\sum_{i=1}^n w_i^1 Y_i}{\sum_{i=1}^n w_i^1} \tag{11}$$

Spectral Clustering

- Spectral clustering is a proxy or relaxation to finding clusters on a graph that minimizes inter cluster interactions.
- Consider a graph $G(V, E)$, where V denote the vertices of the graph and E denotes the edges on the graph. We consider undirected graphs.

Spectral Clustering

- Let S denote the similarity matrix of the graph, such that $S_{ij} = S_{ji} \geq 0$.
- Let $C_k, k = 1, 2, \ldots, K$ denote sub-graphs or clusters of the graph, such that

$$C_k \cap C_l = \emptyset \quad \forall k \neq \text{and} \cup_k C_k = V.$$

- Let

$$RatioCut(C_1, C_2, \ldots, C_K) := \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i \in C_k, j \notin C_k} S_{ij}$$

- Let the obective be to

$$minimize \quad RatioCut(C_1, C_2, \ldots, C_K)$$

Graph Laplacian

- Let D be a diagonal matrix such that $D_{ii} = \sum_j S_{ij}$.

Graph Laplacian

- Let D be a diagonal matrix such that $D_{ii} = \sum_j S_{ij}$.
- The un-normalised Laplacian $L = D - S$.

Graph Laplacian

- Let D be a diagonal matrix such that $D_{ii} = \sum_j S_{ij}$.
- The un-normalised Laplacian $L = D - S$.
- Properties of the Laplacian matrix L:
  - For every $f \in \mathcal{R}^n$ we have

$$f^T L f = \frac{1}{2} \sum_{i,j} S_{ij}(f_i - f_j)^2 \geq 0$$

  - L is symmetric and positive semi-definite
  - The smallest eignvalue of L is zero, and the corresponding eigenvector is the all one vector $\mathbf{1}$.
  - L has n non-negative, real-valued eigenvalues = $0 \leq \lambda_1 \leq \lambda_2 \ldots \leq \lambda_n$.

RatioCut via Graph Laplacian

- It can be shown that

$$RatioCut(C_1, C_2, \ldots, C_K) = Tr(H^T L H),$$

where $H \in \mathcal{R}^{n,K}$ is such that

$$H_{ik} = \frac{1}{\sqrt{|C_k|}} 1_{\{i \in C_k\}}$$

and further the columns of H are orthonormal.

- Minimising RatioCut then boils to finding the appropriate H.
- However, finding an exact solution is difficult as the problem is an integer programming problem.
- Hence, we try to find an H with orthonormal columns that minimizes $Tr(H^T L H)$.
- The columns are the eigenvectors corresponding to the lower K eigenvalues of L.

Spectral Clustering Algorithm

5 Spectral Clustering(S, K)

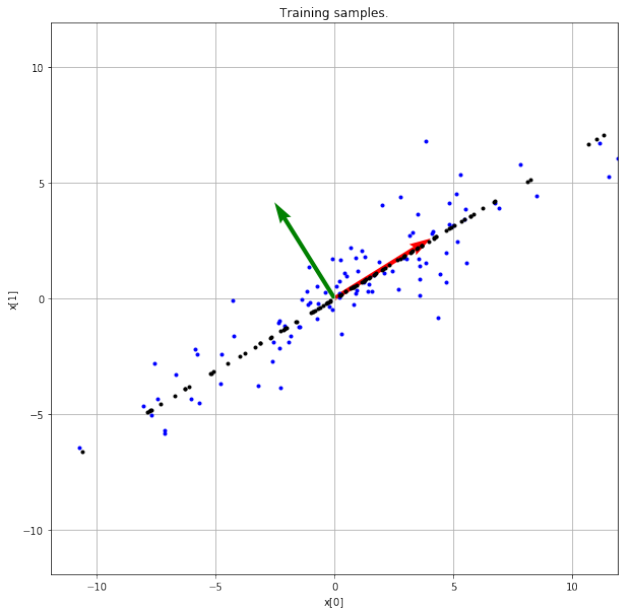
1: Compute the Graph Laplacian L.
2: Let $U \in \mathcal{R}^{n,K}$ be the matrix whose columns correspond to the lowest K eigenvalues of L.
3: Let $v_1, v_2, , \ldots, v_n$ be the rows of U.
4: Cluster $v_1, v_2, , \ldots, v_n$ using any clustering technique (K-means will do).
5: **return** Clusters given by K-means

HANDS-ON - Spectral clustering for earlier problem.

- One needs to create an appropriate similarity matrix.
- Defining the appropriate similarity matrix S has large impact on the performance of the algorithm.
- Try different similarity matrices.

PCA for Dimensionality Reduction

- Given a set of datapoints, what is the best linear transformation to comrpess the data?
- Objective will be to minimize the MSE between the original data and the recovered version after compression.
- The solution is Principal Component Analysis (PCA).



PCA Derivation

- To begin, let us look at the linear projection of the data points to a single dimension.
- Let w be the projection vector. A datapoint x_i is transformed to $\hat{x}_i = (w^T x_i)w$.
- The MSE across all datapoints is obtained as:

$$MSE(w) = \frac{1}{M} \sum_{i=1}^{M} \|x_i - (w^T x_i)w\|^2. \tag{12}$$

PCA Derivation

The aim is to find the optimum w^* that minimises the MSE

$$w^* = \arg \min_{w:\|w\|=1} MSE(w) \tag{13}$$

$$= \arg \min_{w:\|w\|=1} \frac{1}{M} \sum_{i=1}^{M} \|x_i - (w^T x_i) w\|^2 \tag{14}$$

$$= \arg \min_{w:\|w\|=1} \frac{1}{M} \sum_{i=1}^{M} (x_i - (w^T x_i) w)^T (x_i - (w^T x_i) w) \tag{15}$$

$$= \arg \min_{w:\|w\|=1} \frac{1}{M} \sum_{i=1}^{M} x_i^T x_i - (w^T x_i)(w^T x_i) \tag{16}$$

$$- (w^T x_i)(x_i^T w) + (w^T x_i)^2 w^T w \tag{17}$$

$$= \arg \min_{w:\|w\|=1} \frac{1}{M} \sum_{i=1}^{M} x_i^T x_i - (w^T x_i)^2 \tag{18}$$

$$= \arg \max_{w:\|w\|=1} \frac{1}{M} \sum_{i=1}^{M} w^T x_i x_i^T w \tag{19}$$

$$= \arg \max_{w:\|w\|=1} w^T \left(\frac{1}{M} \sum_{i=1}^{M} x_i x_i^T \right) w. \tag{20}$$

PCA Derivation

We know that the solution to the previous problem is the eigenvector of the covariance matrix $\Sigma = \frac{1}{M} \sum_{i=1}^{M} x_i x_i^T$ corresponding to the largest eigenvalue. Similarly, we can show that the projection matrix for projecting to a k dimensional subspace consists of the eigenvectors corresponding to the largest k eigenvalues of the covariance matrix.