Machine Learning for Signal Processing

Clustering

Bhiksha Raj

Class 11. 13 Oct 2016
Statistical Modelling and Latent Structure

• Much of statistical modelling attempts to identify *latent* structure in the data
  – Structure that is not immediately apparent from the observed data
  – But which, if known, helps us explain it better, and make predictions from or about it

• Clustering methods attempt to extract such structure from *proximity*
  – *First-level* structure (as opposed to deep structure)

• We will see other forms of latent structure discovery later in the course
Clustering
How
Clustering

What is clustering

- Clustering is the determination of naturally occurring grouping of data/instances (with low within-group variability and high between-group variability)
Clustering

• What is clustering
  – Clustering is the determination of naturally occurring grouping of data/instances (with low within-group variability and high between-group variability)
Clustering

• What is clustering
  – Clustering is the determination of naturally occurring grouping of data/instances (with low within-group variability and high between-group variability)

• How is it done
  – Find groupings of data such that the groups optimize a “within-group-variability” objective function of some kind
Clustering

• **What is clustering**
  – Clustering is the determination of naturally occurring grouping of data/instances *(with low within-group variability and high between-group variability)*

• **How is it done**
  – Find groupings of data such that the groups optimize a “within-group-variability” objective function of some kind
    – The objective function used affects the nature of the discovered clusters
      • E.g. Euclidean distance vs. •
Clustering

• What is clustering
  – Clustering is the determination of naturally occurring grouping of data/instances (with low within-group variability and high between-group variability)

• How is it done
  – Find groupings of data such that the groups optimize a “within-group-variability” objective function of some kind
  – The objective function used affects the nature of the discovered clusters
    • E.g. Euclidean distance vs.
    • Distance from center
Why Clustering

• Automatic grouping into “Classes”
  – Different clusters may show different behavior

• Quantization
  – All data within a cluster are represented by a single point

• Preprocessing step for other algorithms
  – Indexing, categorization, etc.
Finding natural structure in data

• Find natural groupings in data for further analysis
• Discover *latent* structure in data
Some Applications of Clustering

• Image segmentation
Representation: Quantization

• Quantize every vector to one of $K$ (vector) values
• What are the optimal $K$ vectors? How do we find them? How do we perform the quantization?
• LBG algorithm
• How to retrieve all music videos by this guy?
• Build a classifier
  – But how do you *represent* the video?
Representation: BOW

• Bag of words representations of video/audio/data

Training: Each point is a video frame

Representation: Each number is the #frames assigned to the codeword
Obtaining “Meaningful” Clusters

• Two key aspects:
  – 1. The feature representation used to characterize your data
  – 2. The “clustering criteria” employed
Clustering Criterion

• The “Clustering criterion” actually has two aspects

• Cluster compactness criterion
  – Measure that shows how “good” clusters are
    • The objective function

• Distance of a point from a cluster
  – To determine the cluster a data vector belongs to
“Compactness” criteria for clustering

• Distance based measures
  – Total distance between each element in the cluster and every other element in the cluster
“Compactness” criteria for clustering

• Distance based measures
  – Total distance between each element in the cluster and every other element in the cluster
“Compactness” criteria for clustering

- Distance based measures
  - Total distance between each element in the cluster and every other element in the cluster
  - Distance between the two farthest points in the cluster
“Compactness” criteria for clustering

• Distance based measures
  – Total distance between each element in the cluster and every other element in the cluster
  – Distance between the two farthest points in the cluster
  – Total distance of every element in the cluster from the centroid of the cluster
“Compactness” criteria for clustering

• Distance based measures
  – Total distance between each element in the cluster and every other element in the cluster
  – Distance between the two farthest points in the cluster
  – Total distance of every element in the cluster from the centroid of the cluster
“Compactness” criteria for clustering

- Distance based measures
  - Total distance between each element in the cluster and every other element in the cluster
  - Distance between the two farthest points in the cluster
  - Total distance of every element in the cluster from the centroid of the cluster
“Compactness” criteria for clustering

• Distance based measures
  – Total distance between each element in the cluster and every other element in the cluster
  – Distance between the two farthest points in the cluster
  – Total distance of every element in the cluster from the centroid of the cluster
  – Distance measures are often weighted Minkowski metrics

\[ dist = \sqrt[n]{w_1|a_1 - b_1|^n + w_2|a_2 - b_2|^n + \ldots + w_M|a_M - b_M|^n} \]
Clustering: Distance from cluster

- How far is a data point from a cluster?
  - Euclidean or Minkowski distance from the centroid of the cluster
Clustering: Distance from cluster

• How far is a data point from a cluster?
  – Euclidean or Minkowski distance from the centroid of the cluster
  – Distance from the closest point in the cluster
Clustering: Distance from cluster

• How far is a data point from a cluster?
  – Euclidean or Minkowski distance from the centroid of the cluster
  – Distance from the closest point in the cluster
  – Distance from the farthest point in the cluster
Clustering: Distance from cluster

- How far is a data point from a cluster?
  - Euclidean or Minkowski distance from the centroid of the cluster
  - Distance from the closest point in the cluster
  - Distance from the farthest point in the cluster
  - Probability of data measured on cluster distribution
Clustering: Distance from cluster

• How far is a data point from a cluster?
  – Euclidean or Minkowski distance from the centroid of the cluster
  – Distance from the closest point in the cluster
  – Distance from the farthest point in the cluster
  – Probability of data measured on cluster distribution
  – Fit of data to cluster-based regression
Optimal clustering: Exhaustive enumeration

• All possible combinations of data must be evaluated
  – If there are M data points, and we desire N clusters, the number of ways of separating M instances into N clusters is

  \[
  \frac{1}{M!} \sum_{i=0}^{N} (-1)^i\binom{N}{i} (N-i)^M
  \]

  – Exhaustive enumeration based clustering requires that the objective function (the “Goodness measure”) be evaluated for every one of these, and the best one chosen

• This is the only correct way of optimal clustering
  – Unfortunately, it is also computationally unrealistic
Not-quite non sequitur: Quantization

- Linear quantization (uniform quantization):
  - Each digital value represents an equally wide range of analog values
  - Regardless of distribution of data
  - Digital-to-analog conversion represented by a “uniform” table

<table>
<thead>
<tr>
<th>Signal Value</th>
<th>Bits</th>
<th>Mapped to</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \geq 3.75v$</td>
<td>11</td>
<td>$3 \times \text{const}$</td>
</tr>
<tr>
<td>$3.75v &gt; S \geq 2.5v$</td>
<td>10</td>
<td>$2 \times \text{const}$</td>
</tr>
<tr>
<td>$2.5v &gt; S \geq 1.25v$</td>
<td>01</td>
<td>$1 \times \text{const}$</td>
</tr>
<tr>
<td>$1.25v &gt; S \geq 0v$</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Not-quite non sequitur: Quantization

- Non-Linear quantization:
  - Each digital value represents a different range of analog values
    - Finer resolution in high-density areas
    - Mu-law / A-law assumes a Gaussian-like distribution of data
  - Digital-to-analog conversion represented by a “non-uniform” table

<table>
<thead>
<tr>
<th>Signal Value</th>
<th>Bits</th>
<th>Mapped to</th>
</tr>
</thead>
<tbody>
<tr>
<td>S &gt;= 4v</td>
<td>11</td>
<td>4.5</td>
</tr>
<tr>
<td>4v &gt; S &gt;= 2.5v</td>
<td>10</td>
<td>3.25</td>
</tr>
<tr>
<td>2.5v &gt; S &gt;= 1v</td>
<td>01</td>
<td>1.25</td>
</tr>
<tr>
<td>1.0v &gt; S &gt;= 0v</td>
<td>0</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Analog value (arrows are quantization levels)
Non-uniform quantization

• If data distribution is not Gaussian-ish?
  – Mu-law / A-law are not optimal
  – How to compute the optimal ranges for quantization?
    • Or the optimal table
The Lloyd Quantizer

- Lloyd quantizer: An iterative algorithm for computing optimal quantization tables for non-uniformly distributed data
- Learned from “training” data
Lloyd Quantizer

- Randomly initialize quantization points
  - Right column entries of quantization table
Lloyd Quantizer

- Randomly initialize quantization points
  - Right column entries of quantization table

- Assign all training points to the nearest quantization point
  - Draw boundaries
Lloyd Quantizer

- Randomly initialize quantization points
  - Right column entries of quantization table

- Assign all training points to the nearest quantization point
  - Draw boundaries

- Reestimate quantization points
Lloyd Quantizer

- Randomly initialize quantization points
  - Right column entries of quantization table

- Assign all training points to the nearest quantization point
  - Draw boundaries

- Reestimate quantization points

- Iterate until convergence
Generalized Lloyd Algorithm: K–means clustering

• K means is an iterative algorithm for clustering vector data

• General procedure:
  – Initially group data into the required number of clusters somehow (initialization)
  – Assign each data point to the closest cluster
  – Once all data points are assigned to clusters, redefine clusters
  – Iterate
K–means

• Problem: Given a set of data vectors, find natural clusters

• Clustering criterion is **scatter**: distance from the centroid
  – Every cluster has a centroid
  – The centroid represents the cluster

• **Definition**: The **centroid** is the weighted mean of the cluster
  – Weight = 1 for basic scheme

\[
m_{\text{cluster}} = \frac{1}{\sum_{i \in \text{cluster}} w_i} \sum_{i \in \text{cluster}} w_i x_i
\]
K–means

1. Initialize a set of centroids randomly
**K-means**

1. Initialize a set of centroids randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$
1. Initialize a set of centroids randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{cluster} = \text{distance}(x, m_{cluster})$

3. Put data point in the cluster of the closest centroid
   - Cluster for which $d_{cluster}$ is minimum

K–means
K–means

1. Initialize a set of centroids randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   • $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   • Cluster for which $d_{\text{cluster}}$ is minimum
K–means

1. Initialize a set of centroids randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   • $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   • Cluster for which $d_{\text{cluster}}$ is minimum
K–means

1. Initialize a set of centroids randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   • $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   • Cluster for which $d_{\text{cluster}}$ is minimum
K–means

1. Initialize a set of centroids randomly

2. For each data point \( x \), find the distance from the centroid for each cluster
   - \( d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}}) \)

3. Put data point in the cluster of the closest centroid
   - Cluster for which \( d_{\text{cluster}} \) is minimum
K–means

1. Initialize a set of centroids randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   - Cluster for which $d_{\text{cluster}}$ is minimum
K–means

1. Initialize a set of centroids randomly

2. For each data point \( x \), find the distance from the centroid for each cluster
   - \( d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}}) \)

3. Put data point in the cluster of the closest centroid
   - Cluster for which \( d_{\text{cluster}} \) is minimum
**K–means**

1. Initialize a set of centroids randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   - Cluster for which $d_{\text{cluster}}$ is minimum

4. When all data points are clustered, recompute centroids
   
   $$m_{\text{cluster}} = \frac{1}{\sum_{i \in \text{cluster}} w_i} \sum_{i \in \text{cluster}} w_i x_i$$
K–means

1. Initialize a set of centroids randomly
2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$
3. Put data point in the cluster of the closest centroid
   - Cluster for which $d_{\text{cluster}}$ is minimum
4. When all data points are clustered, recompute centroids

\[
m_{\text{cluster}} = \frac{1}{\sum_{i \in \text{cluster}} w_i} \sum_{i \in \text{cluster}} w_i x_i
\]
5. If not converged, go back to 2
K-Means comments

• The distance metric determines the clusters
  – In the original formulation, the distance is $L_2$ distance
    • Euclidean norm, $w_i = 1$

$$\text{distance}_{\text{cluster}}(x, m_{\text{cluster}}) = \| x - m_{\text{cluster}} \|_2$$

– If we replace every $x$ by $m_{\text{cluster}}(x)$, we get Vector Quantization

• K-means is an instance of generalized EM

• Not guaranteed to converge for all distance metrics
Initialization

• Random initialization

• Top-down clustering
  – Initially partition the data into two (or a small number of) clusters using K means
  – Partition each of the resulting clusters into two (or a small number of) clusters, also using K means
  – Terminate when the desired number of clusters is obtained
K-Means for Top–Down clustering

1. Start with one cluster
K-Means for Top–Down clustering

1. Start with one cluster
K-Means for Top–Down clustering

1. Start with one cluster

2. Split each cluster into two:
   - Perturb centroid of cluster slightly (by < 5%) to generate two centroids
K-Means for Top–Down clustering

1. Start with one cluster

2. Split each cluster into two:
   - Perturb centroid of cluster slightly (by < 5%) to generate two centroids
K-Means for Top–Down clustering

1. Start with one cluster

2. Split each cluster into two:
   - Perturb centroid of cluster slightly (by < 5%) to generate two centroids

3. Initialize K means with new set of centroids
K-Means for Top–Down clustering

1. Start with one cluster

2. Split each cluster into two:
   - Perturb centroid of cluster slightly (by < 5%) to generate two centroids

3. Initialize K means with new set of centroids

4. Iterate Kmeans until convergence
K-Means for Top–Down clustering

1. Start with one cluster

2. Split each cluster into two:
   - Perturb centroid of cluster slightly (by < 5%) to generate two centroids

3. Initialize K means with new set of centroids

4. Iterate Kmeans until convergence
K-Means for Top–Down clustering

1. Start with one cluster

2. Split each cluster into two:
   - Perturb centroid of cluster slightly (by < 5%) to generate two centroids

3. Initialize K means with new set of centroids

4. Iterate Kmeans until convergence

5. If the desired number of clusters is not obtained, return to 2
Non-Euclidean clusters

- Basic K-means results in good clusters in Euclidean spaces
  - Alternately stated, will only find clusters that are “good” in terms of Euclidean distances
- Will not find other types of clusters
Non-Euclidean clusters

- For other forms of clusters we must modify the distance measure
  - E.g. distance from a circle
- May be viewed as a distance in a higher dimensional space
  - I.e. *Kernel* distances
  - *Kernel* K-means
- Other related clustering mechanisms:
  - Spectral clustering
    - Non-linear weighting of adjacency
  - Normalized cuts..
The Kernel Trick

- Transform the data into a synthetic higher-dimensional space where the desired patterns become natural clusters
  - E.g. the quadratic transform above

- Problem: What is the function/space?

- Problem: Distances in higher dimensional-space are more expensive to compute
  - Yet only carry the same information in the lower-dimensional space

\[ f([x,y]) \rightarrow [x,y,z] \]
\[ x = x \]
\[ y = y \]
\[ z = \alpha(x^2 + y^2) \]
Distance in higher-dimensional space

- Transform data $x$ through a *possibly unknown* function $\Phi(x)$ into a higher (potentially infinite) dimensional space
  - $z = \Phi(x)$

- The distance between two points is computed in the higher-dimensional space
  - $d(x_1, x_2) = \| z_1 - z_2 \|^2 = \| \Phi(x_1) - \Phi(x_2) \|^2$

- $d(x_1, x_2)$ can be computed without computing $z$
  - Since it is a direct function of $x_1$ and $x_2$
Distance in higher-dimensional space

- Distance in lower-dimensional space: A combination of dot products
  \[ ||z_1 - z_2||^2 = (z_1 - z_2)^T(z_1 - z_2) = z_1.z_1 + z_2.z_2 - 2z_1.z_2 \]

- Distance in higher-dimensional space
  \[ d(x_1, x_2) = ||\Phi(x_1) - \Phi(x_2)||^2 \]
  \[ = \Phi(x_1).\Phi(x_1) + \Phi(x_2).\Phi(x_2) - 2\Phi(x_1).\Phi(x_2) \]

- \( d(x_1, x_2) \) can be computed without knowing \( \Phi(x) \) if:
  - \( \Phi(x_1).\Phi(x_2) \) can be computed for any \( x_1 \) and \( x_2 \) without knowing \( \Phi(.) \)
The Kernel function

• A kernel function $K(x_1,x_2)$ is a function such that:
  $- K(x_1,x_2) = \Phi(x_1) \cdot \Phi(x_2)$

• Once such a kernel function is found, the distance in higher-dimensional space can be found in terms of the kernels
  $- d(x_1, x_2) = ||\Phi(x_1) - \Phi(x_2)||^2$
  $= \Phi(x_1) \cdot \Phi(x_1) + \Phi(x_2) \cdot \Phi(x_2) - 2 \Phi(x_1) \cdot \Phi(x_2)$
  $= K(x_1,x_1) + K(x_2,x_2) - 2K(x_1,x_2)$

• But what is $K(x_1,x_2)$?
A property of the dot product

- For any vector $\mathbf{v}$, $\mathbf{v}^T \mathbf{v} = ||\mathbf{v}||^2 \geq 0$
  - This is just the length of $\mathbf{v}$ and is therefore non-negative

- For any vector $\mathbf{u} = \sum_i a_i \mathbf{v}_i$, $||\mathbf{u}||^2 \geq 0$
  => $(\sum_i a_i \mathbf{v}_i)^T (\sum_i a_i \mathbf{v}_i) \geq 0$
  => $\sum_i \sum_j a_i a_j \mathbf{v}_i \cdot \mathbf{v}_j \geq 0$

- This holds for ANY real $\{a_1, a_2, \ldots\}$
The Mercer Condition

• If \( z = \Phi(x) \) is a high-dimensional vector derived from \( x \) then for all real \( \{a_1, a_2, \ldots\} \) and any set \( \{z_1, z_2, \ldots\} = \{\Phi(x_1), \Phi(x_2), \ldots\} \)
  
  \[
  \Sigma_i \Sigma_j a_i a_j z_i . z_j \geq 0
  \]
  
  \[
  \Sigma_i \Sigma_j a_i a_j \Phi(x_i) . \Phi(x_j) \geq 0
  \]

• If \( K(x_1, x_2) = \Phi(x_1) . \Phi(x_2) \)
  
  \[
  \Rightarrow \Sigma_i \Sigma_j a_i a_j K(x_i, x_j) \geq 0
  \]

• Any function \( K() \) that satisfies the above condition is a valid kernel function
The Mercer Condition

• $K(x_1,x_2) = \Phi(x_1) \cdot \Phi(x_2)$
  $=> \sum_i \sum_j a_i a_j K(x_i,x_j) >= 0$

• A corollary: If any kernel $K(.)$ satisfies the Mercer condition

  $d(x_1, x_2) = K(x_1,x_1) + K(x_2,x_2) - 2K(x_1,x_2)$

  satisfies the following requirements for a “distance”

  – $d(x,x) = 0$
  – $d(x,y) >= 0$
  – $d(x,w) + d(w,y) >= d(x,y)$
Typical Kernel Functions

• Linear: \( K(x, y) = x^T y + c \)

• Polynomial \( K(x, y) = (ax^T y + c)^n \)

• Gaussian: \( K(x, y) = \exp(-||x-y||^2/\sigma^2) \)

• Exponential: \( K(x, y) = \exp(-||x-y||/\lambda) \)

• Several others
  – Choosing the right Kernel with the right parameters for your problem is an artform
Kernel K-means

- Perform the K-mean in the Kernel space
  - The space of $z = \Phi(x)$

- The algorithm..
The mean of a cluster

- The average value of the points in the cluster *computed in the high-dimensional space*

\[
m_{\text{cluster}} = \frac{1}{N_{\text{cluster}}} \sum_{i \in \text{cluster}} \Phi(x_i)
\]

- Alternately the weighted average

\[
m_{\text{cluster}} = \frac{1}{\sum_{i \in \text{cluster}} w_i} \sum_{i \in \text{cluster}} w_i \Phi(x_i) = C \sum_{i \in \text{cluster}} w_i \Phi(x_i)
\]
The mean of a cluster

- The average value of the points in the cluster *computed in the high-dimensional space*

\[ m_{\text{cluster}} = \frac{1}{N_{\text{cluster}}} \sum_{i \in \text{cluster}} \Phi(x_i) \]

**RECALL:** We may never actually be able to compute this mean because \( \Phi(x) \) is not known.

- Alternately the weighted average

\[ m_{\text{cluster}} = \frac{1}{\sum_{i \in \text{cluster}} w_i} \sum_{i \in \text{cluster}} w_i \Phi(x_i) = C \sum_{i \in \text{cluster}} w_i \Phi(x_i) \]
**K–means**

- Initialize the clusters with a random set of K points
  - Cluster has 1 point

- For each data point $x$, find the closest cluster

\[
\text{cluster}(x) = \min_{\text{cluster}} d(x, \text{cluster}) = \min_{\text{cluster}} \| \Phi(x) - m_{\text{cluster}} \|^2
\]

\[
d(x, \text{cluster}) = \| \Phi(x) - m_{\text{cluster}} \|^2 = \left( \Phi(x) - C \sum_{i \in \text{cluster}} w_i \Phi(x_i) \right)^T \left( \Phi(x) - C \sum_{i \in \text{cluster}} w_i \Phi(x_i) \right)
\]

\[
= \left( \Phi(x)^T \Phi(x) - 2C \sum_{i \in \text{cluster}} w_i \Phi(x)^T \Phi(x_i) + C^2 \sum_{i \in \text{cluster}} \sum_{j \in \text{cluster}} w_i w_j \Phi(x_i)^T \Phi(x_j) \right)
\]

\[
= K(x, x) - 2C \sum_{i \in \text{cluster}} w_i K(x, x_i) + C^2 \sum_{i \in \text{cluster}} \sum_{j \in \text{cluster}} w_i w_j K(x_i, x_j)
\]

Computed entirely using only the kernel function!
**K–means**

1. Initialize a set of *clusters* randomly
K–means

1. Initialize a set of clusters randomly

The centroids are virtual: we don’t actually compute them explicitly!

\[ m_{\text{cluster}} = \frac{1}{\sum_{i \in \text{cluster}} w_i} \sum_{i \in \text{cluster}} w_i x_i \]
**K-means**

1. Initialize a set of clusters randomly

2. For each data point \( x \), find the distance from the centroid for each cluster

   \[
   d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})
   \]

   \[
   d_{\text{cluster}} = K(x, x) - 2C \sum_{i \in \text{cluster}} w_i K(x, x_i) + C^2 \sum_{i \in \text{cluster}} \sum_{j \in \text{cluster}} w_i w_j K(x_i, x_j)
   \]
**K–means**

1. Initialize a set of clusters randomly

2. For each data point \( x \), find the distance from the centroid for each cluster
   - \( d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}}) \)

3. Put data point in the cluster of the closest centroid
   - Cluster for which \( d_{\text{cluster}} \) is minimum
K–means

1. Initialize a set of clusters randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   - Cluster for which $d_{\text{cluster}}$ is minimum
K–means

1. Initialize a set of clusters randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   - Cluster for which $d_{\text{cluster}}$ is minimum
**K–means**

1. Initialize a set of clusters randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   - Cluster for which $d_{\text{cluster}}$ is minimum
K–means

1. Initialize a set of clusters randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   - Cluster for which $d_{\text{cluster}}$ is minimum
**K–means**

1. Initialize a set of clusters randomly

2. For each data point \( x \), find the distance from the centroid for each cluster
   - \( d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}}) \)

3. Put data point in the cluster of the closest centroid
   - Cluster for which \( d_{\text{cluster}} \) is minimum
K–means

1. Initialize a set of clusters randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   - $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   - Cluster for which $d_{\text{cluster}}$ is minimum
K–means

1. Initialize a set of clusters randomly

2. For each data point $x$, find the distance from the centroid for each cluster
   • $d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}})$

3. Put data point in the cluster of the closest centroid
   • Cluster for which $d_{\text{cluster}}$ is minimum

4. When all data points are clustered, recompute centroids
   $$m_{\text{cluster}} = \frac{1}{\sum_{i \in \text{cluster}} w_i} \sum_{i \in \text{cluster}} w_i x_i$$

   • We do not explicitly compute the means
   • May be impossible – we do not know the high-dimensional space
   • We only know how to compute inner products in it
Kernel K–means

1. Initialize a set of clusters randomly

2. For each data point \( x \), find the distance from the centroid for each cluster
   
   \[ d_{\text{cluster}} = \text{distance}(x, m_{\text{cluster}}) \]

3. Put data point in the cluster of the closest centroid
   
   • Cluster for which \( d_{\text{cluster}} \) is minimum

4. When all data points are clustered, recompute centroids

   \[ m_{\text{cluster}} = \frac{1}{\sum_{i \in \text{cluster}} w_i} \sum_{i \in \text{cluster}} w_i x_i \]

5. If not converged, go back to 2

• We do not explicitly compute the means
• May be impossible – we do not know the high-dimensional space
• We only know how to compute inner products in it
How many clusters?

• Assumptions:
  – Dimensionality of kernel space > no. of clusters
  – Clusters represent separate directions in Kernel spaces

• Kernel correlation matrix $\mathbf{K}$
  – $K_{ij} = K(x_i, x_j)$

• Find Eigen values $\Lambda$ and Eigen vectors $\mathbf{e}$ of kernel matrix
  – No. of clusters = no. of dominant $\lambda_i (1^T \mathbf{e}_i)$ terms
Spectral Methods

• “Spectral” methods attempt to find “principal” subspaces of the high-dimensional kernel space

• Clustering is performed in the principal subspaces
  – Normalized cuts
  – Spectral clustering

• Involves finding Eigenvectors and Eigen values of Kernel matrix

• Fortunately, provably analogous to Kernel K-means
Other clustering methods

• Regression based clustering
• Find a regression representing each cluster
• Associate each point to the cluster with the best regression
  – Related to kernel methods
Clustering..

• Many many other variants
• Many applications..

• Important: Appropriate choice of feature
  – Appropriate choice of feature may eliminate need for kernel trick..

  – Google is your friend.