Accelerating mean shift using tree-based algorithms

Shao-Hsuan Wang

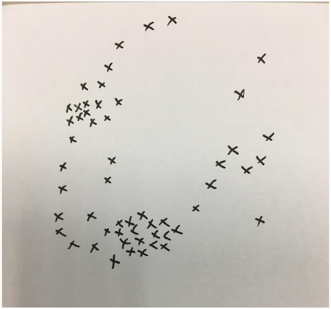
September 21, 2018

Outline

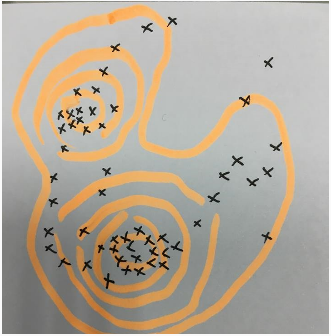
- Clusters
- Kernel density estimates (KDE)
- Mean shift
- High dimensional data problem
- K-dimensional tree (KD tree)
- Fast mean shift procedure
- Some problems

Clusters

Let the data points be a finite sample S of a probability density function. Then, a natural way to characterize the cluster structure of dataset is by finding regions containing a high density of data.



Let the data points be a finite sample S of a probability density function. Then, a natural way to characterize the cluster structure of dataset is by finding regions containing a high density of data.



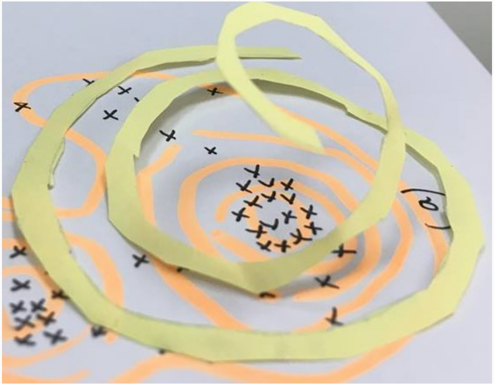
Clusters

1. Parametric model: Gaussian mixture
2. nonparametric: kernel density estimates (KDE)
3. Semi-parametric model

Note: A KDE is a generalization of histograms to define density estimates. (smooth!)

Clusters

We focus on clusters defined by the modes of the KDE. A mode is a local maximum of the density.



kernel density estimates

Let $x \in S \subseteq \mathrm{R}^p$. A function K:$S \mapsto \mathrm{R}$ is said to be a kernel if there exists a profile, $k : [0, \infty] \mapsto \mathrm{R}$, such that

$$K(x) = k(\|x\|^2)$$

and

(i) k is nonnegative

(ii) k is non-increasing: $k(a) \geq k(b)$ if $a < b$

(iii) k is piecewise continuous and $\int_0^\infty k(r)dr < \infty$

For example, $k(x) = \mathrm{I}(x \leq 1)$ (flat kernel), $k(x) = \exp(-x)$ (Gaussian kernel)

kernel density estimates

Some Properties:

(1) Let X_i, $i = 1 \ldots, n$ be iid rv and the corresponding density be $f(\cdot)$ with finite moments. Then, we have kernel density estimate

$$\widehat{f}(x) = \frac{1}{h} \sum_{i=1}^{n} K\left(\frac{X_i - x}{h}\right)$$

, where h is a bandwidth. For example, Let K be Gaussian kernel, we have $\mathrm{E}[\widehat{f}(x)] = f(x) + O(h^2)$ and $\mathrm{var}(\widehat{f}(x)) = O((nh)^{-1})$ and the optimal bandwidth $h = O(n^{-1/5})$ by minimizing aymptotic mean integrated squared error $\mathrm{AMISE}(h) = O((nh)^{-1}) + O(h^4)$.

Some Properties:

(2)

$$\nabla \widehat{f}(x) = \frac{2}{h^2} \sum_{i=1}^{n} \dot{k}\left(\left\| \frac{X_i - x}{h} \right\|^2 \right)(X_i - x) = 0$$

$$\Leftrightarrow$$

$$m(x) - x = 0,$$

where

$$m(x) \equiv \sum_{i=1}^{n} \frac{\dot{k}\left(\left\| \frac{X_i - x}{h} \right\|^2 \right)}{\sum_{i=1}^{n} \dot{k}\left(\left\| \frac{X_i - x}{h} \right\|^2 \right)} X_i.$$

- The sample mean at $x \in \mathrm{R}^p$ is

$$m(x) = \sum_{s \in S} \frac{K(s-x)}{\sum_{s \in S} K(s-x)} s.$$

Then difference $m(x) - x$ is called mean shift in Fukunaga and Hosteler (1975).

- Generalized mean shift (Cheng (1995)):

$$m(x) = \sum_{s \in S} \frac{K(s-x)w(s)}{\sum_{s \in S} K(s-x)w(s)} s,$$

where $w : S \mapsto (0, \infty)$, a weight function.

Let $T \subseteq \mathrm{R}^p$ be a finite set ("the cluster centers"). The evolution of T in the form of iterations $T \leftarrow m(T)$ with $m(T) : \{m(t) : t \in T\}$ is called a mean shift algorithm. For each t, a sequence $m(t), m(m(t)), \ldots$ is called trajectory of t. The algorithm halts when it reaches a fixed point $m(T) = T$ (Cheng (1995)).

- non-blurring: $T \leftarrow m(T)$
- blurring: $S \leftarrow m(S)$

Recall: S is the data points.

High dimensional data problem

Large n, p: high computational costs. For example,

$p = 10^4, n = 10^5$

High dimensional data problem

For example, γ-SUP clustering algorithm (Chen et al. (2014)):

begin

  iter $\leftarrow 0$

  start with $\tilde{\mu}_i \leftarrow x_i/\tau$, $i = 1, \ldots, n$

  repeat

    for $i = 1 : n$

      $w_{ij} \leftarrow \exp_{1-s}(-\|\tilde{u}_i - \tilde{u}_j\|_2^2)$, $j = 1, \ldots, n$

      $z_i \leftarrow \sum_{j=1}^n \frac{w_{ij}}{\sum_{k=1}^n w_{ik}} \tilde{\mu}_j$

    end

  $\tilde{\mu}_i \leftarrow z_i, i = 1, \ldots, n$

  iter $\leftarrow$ **iter**$+1$

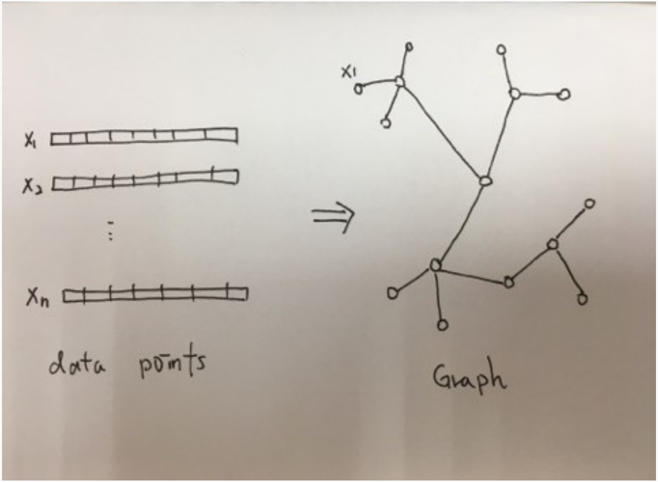
  until convergence

  output distinct cluster centers $\{\tau\tilde{\mu}_i, 1 \leq i \leq n\}$ and cluster membership

end

Computational complexity: $O(pn^2)$

High dimensional data problem



Accelerating

For example, accelerating γ-SUP clustering algorithm using KD tree:

begin

iter $\leftarrow 0$

start with $\tilde{\mu}_i \leftarrow x_i/\tau$, $i = 1, \dots, n$

repeat

kdt = computeKdTree({ $\tilde{\mu}_i$ })

for $i = 1 : n$

NN=kdt.nearestNeighbor($\tilde{\mu}_i$)

$w_{ij} \leftarrow \exp_{1-s}(-\|\tilde{u}_i - \tilde{u}_j\|_2^2)$, $j = 1, \dots, D$

$z_i \leftarrow \sum_{j \in \mathrm{NN}} \frac{w_{ij}}{\sum_{k \in \mathrm{NN}} w_{ik}} \tilde{\mu}_j$

end

$\tilde{\mu}_i \leftarrow z_i, i = 1, \dots, n$

iter $\leftarrow$ **iter**$+1$

until convergence

output distinct cluster centers $\{\tau\tilde{\mu}_i, 1 \le i \le n\}$ and cluster membership

end

Computational complexity $O(pn \log n) + O(pDn)$

Accelerating

For example, accelerating γ-SUP clustering algorithm using Fast Mean Shift & KD tree:

begin

  iter $\leftarrow 0$

  start with $C =$ doCell($\{\tilde{\mu}_i\}$), $i = 1, \ldots, n$

  repeat

  $c_j = \frac{1}{n_j} \sum_{k \in C_j} \tilde{\mu}_k$, $j = 1, \ldots, m$

  kdt = computeKdTree($\{ c_j \}$)

    for $j = 1 : m$

      NN=kdt.nearestNeighbor(c_j)

      $w_{j\ell} \leftarrow \exp_{1-s}(-\|c_j - c_\ell\|_2^2)$, $\ell = 1, \ldots, D$

      $z_j \leftarrow \sum_{\ell \in \mathrm{NN}} \frac{w_{j\ell}}{\sum_{k \in \mathrm{NN}} w_{jk}} c_\ell$

    end

  $c_j \leftarrow z_j, j = 1, \ldots, m$

  assignCluster($\{ c_j \}$,$\{ \tilde{\mu}_i \}$)

  iter $\leftarrow$ **iter**+1

  until convergence

  output distinct cluster centers $\{\tau\tilde{\mu}_i, 1 \le i \le n\}$ and cluster membership

end

Computational complexity $O(pn\log m) + O(pDm)$

KD tree

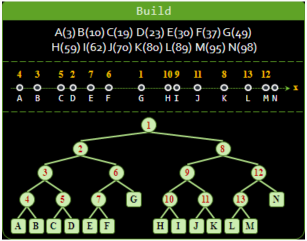


Figure 1: 1-dim KD tree

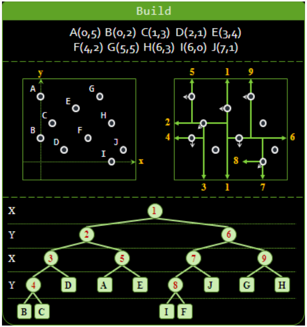


Figure 2: 2-dim KD tree

Computational complexity $O(pn\log n)$

(Source: http://www.csie.ntnu.edu.tw/ u91029/Position.html)

Fast mean shift (Guo et al. (2006))

Make Cells(divide $x_i, i = 1, \ldots, n$ into m local subsets C_j, $j = 1, \ldots, m$)

Let $C = \{C_1, \ldots, C_m\}$.

- S1. Initialize C by randomly selecting a sample as c_1. Let $C = \{c_1\}$, Then, at the ith iteration, $i = 1, \ldots, n$ do S2-S3
- S2. Compute $\|x_i - c_j\|$ $(c_j \in C)$.
- S3. If $\|x_i - c_j\| \leq r$, assign x_i to C_j. Otherwise, add x_i C subset center and assign x_i to this new subset.
- S4. count n_j, the number of samples in C_j and update each $(c_j \in C)$ as $c_j = \sum_{x_i \in C_j} x_i / n_j$.

Computational complexity $O(pmn)$

Fast mean shift (Guo et al. (2006))

First, use a large r_0 to divide the original set into very few local subsets. Then, hierarchically divide the existing local subsets by adopting the re-sampling procedure with gradually decreased r_s until a threshold r_T is reached.

Computational complexity $O(pn \log m)$

Random Projection tree (RP tree)

A simple variant of the KD tree which automatically adapts to intrinsic low dimensional structure in data

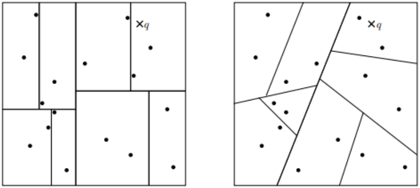


Figure 1: Left: A spatial partitioning of $\mathbb{R}^2$ induced by a k-d tree with three levels. The dots are data points; the cross marks a query point q. Right: Partitioning induced by an RP tree.

(Source: Dasgupata and Freund 2008)

Reduce methods

1. For the sample size n: the random partition, the random sampling, the KNN using tree-based, bin method (cells, blocks).

2. For the image size p: the random partition, the random sampling, principle component, bin method, parameterized (KNN, modelling).

Some Problems

1. Means shift, random partition, and algorithms for the GPU
2. low SNR: False peaks and mean-shift method
3. mean shift for 3D structures
4. Fuzzy entropy: likelihood on graph (similar with LargeVis, UMAP)

References

1. Mean Shift, Mode Seeking, and Clustering (Cheng 1995)

2. A Fast Mean Shift Procedure with New Iteration Strategy and Re-sample (Guo et al. 2006)

3. Random projection trees and low dimensional manifolds (Dasgupata and Freund 2008)

4. Fast Mean Shift by Compact Density Representation (Freedman and Kisilev 2009)

5. γ-SUP: A Clustering Algorithm For Cryo-Election Microscopy Images of Asymmetric Particles. (Chen et al. 2014)

6. Accelerating t-SNE using Tree-Based Algorithms (Van De Maaten 2014)

References

7. A review of mean-shift algorithms for clustering (Carreira-Perpiñán 2015)

8. Accelerated Mean Shift For Static And Streaming Environments (Ende et al. 2015)