Intro to Convex Clustering with Robust Extensions

AMATH Masters Final Project

Kevin Mueller

March 12 2016

Contents

Why Clustering?

- It is an unsupervised technique, which means that it does not require labeled data.
- Broad set of techniques for identifying subgroups in a data set.
  - K-Means
  - Gaussian Mixture Models (Expectation Maximization)
  - Spectral Clustering
  - Hierarchal Clustering
  - Convex Clustering

K-Means Formulation

- Given Data Points $\{x_j\}_{j=1}^N$
- We seek to find a set of k subgroups (index sets) $\{S_i\}_{i=1}^k$
- Let $\{\theta_i\}_{i=1}^k$ be the cluster centers

$$\min_S \sum_{i=1}^k \sum_{j \in S_i} \|x_j - \theta_i\|^2$$

$$\text{s.t} \quad \theta_i = \frac{1}{\text{card } S_i} \sum_{j \in S_i} x_j$$

$$\bigcup_{i=1}^k S_i = \{1, \ldots, N\}$$

Algorithm 1 k-Means clustering (Lloyd's algorithm [Lloyd, 1982])

Require:

- Data points $\{x_j\}_{j=1}^N$.
- The number of clusters $k \leq N$.
- An initialisation of the centroids $\{\theta_i\}_{i=1}^k$.

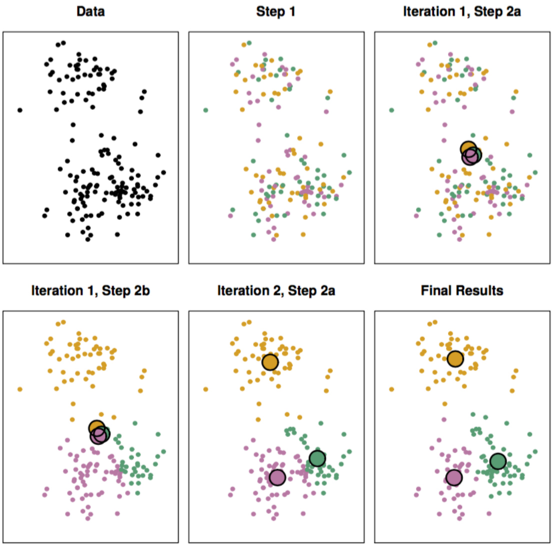
Ensure:

- Index sets $\{S_i\}_{i=1}^k$.

```
loop
    Update index sets: For fixed centroids {θ_i}_{i=1}^k, compute the index sets {S_i}_{i=1}^k,
        S_i ← {j : ‖x_j − θ_i‖ ≤ ‖x_j − θ_l‖, l = 1, ..., k}.
    Update centroids: For fixed index sets {S_i}_{i=1}^k, estimate the centroids {θ_i}_{i=1}^k,
        θ_i ← (1 / card S_i) Σ_{j∈S_i} x_j.
    if No change in assignment since last iteration or maximum number of iterations reached then
        return
    end if
end loop
```

$$S_i \leftarrow \{j : \|x_j - \theta_i\| \leq \|x_j - \theta_l\|, l = 1, \ldots, k\}.$$

$$\theta_i \leftarrow \frac{1}{\operatorname{card} S_i} \sum_{j \in S_i} x_j.$$


Data
Step 1
Iteration 1, Step 2a
Iteration 1, Step 2b
Iteration 2, Step 2a
Final Results

Problems with K-Means

- Requires an initialization of k clusters
- Can yield different clusters for different initial conditions

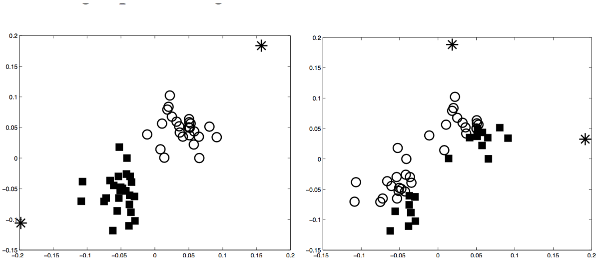


Fig. 1. Illustration of the sensitivity of the k-means clustering algorithm for initial condition.

Convex Relaxation

If we suppose there is a unique optima for the k-means formulation S^*, which is a partitioning of the set $\{1, \ldots, N\}$ into k non-empty disjoint subsets. Then we can rewrite the k-means objective as,

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2$$

$$\text{s.t}\{\mu_1, \ldots, \mu_n\} \text{ contains k unique vectors}$$

Where $\mu \in \mathbb{R}^d$ for $j = 1, \ldots, N$ represents the cluster center for each point x_j. This would still give k clusters assuming that x's belong to the same cluster if their corresponding centroids are the same.

Convex Relaxation

We can reformulate the constraint by counting the unique vectors in the set $\{\mu_1, \ldots, \mu_N\}$. Define an $\mathbb{R} \in N^2$ matrix as, $\Delta_{ij} = \kappa_(\mu_i, \mu_j)$ where $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ has the symmetric property,

$$\kappa(\mu_i,\, \mu_j) = 0 \iff \mu_i = \mu_j$$

An easy example of a κ with these properties are the well known difference norms. In these case

$$\Delta = \begin{pmatrix} 0 & \times & \cdots & \times \\ & \ddots & \ddots & \vdots \\ & & \ddots & \times \\ & & & 0 \end{pmatrix}$$

Convex Relaxation

The number of vectors in the set $\{\mu_j\}_{j=n+1}^{N}$ that are equal to μ_n, is the number of the zeros in the nth row of the upper upper triangle. Therefore to count the number of duplicates we can,

- Count the number of zeros in the first row of the upper triangle
- Count the number of zeros in the second row, unless there is a zero in the same column as the first row.
- Do this for all $N - 1$ rows.

Which is the equivalent of counting the number of columns in the upper triangle, containing at least one zero.

By using the indicator function $I(x) = \begin{cases} 1, & \text{if } x \neq 0 \\ 0, & \text{if } x = 0 \end{cases}$,
the number of zeros in the jth column of Δ is

$$\sum_{i<j} (1 - I(\Delta_{ij})) .$$

The unique number of vectors in the set $\{\mu_j\}_{j=1}^N$ is

$$N - \sum_{j=2}^{N} I \left(\sum_{i<j} (1 - I(\kappa(\mu_i, \mu_j)) \right) .$$

or by using the l_0-norm, which is defined as the number of non-zero elements of a vector the above expression can be written as

$$N - ||\delta||_0$$

Where the vector $\delta = [\delta_2 \dots \delta_N]^T$ is defined as

$$\delta_j = \sum_{i<j} \left(1 - I\left(\kappa(\mu_i, \mu_j)\right)\right)$$
$$= j - 1 - \sum_{i<j} I\left(\kappa(\mu_i, \mu_j)\right) = j - 1 - \|\gamma\|_0.$$

Which implies the vectors $\gamma^j = \left[\gamma_1^j \dots \gamma_{j-1}^j\right]^T$ for $j = 2, \dots, N$ are given by $\gamma_i^j = \kappa\left(\mu_i, \mu_j\right)$ giving the non-convex reformulation of the original k-means,

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2$$

$$\text{s.t. } k = N - \|\delta\|_0$$

Since the ℓ_0-norm is not convex we can make our formulation convex by approximating it with the ℓ_1-norm. This is a popular trick used in compressed sensing, lasso, and other methods. Relaxing our constraint we get,

$$k = N - \sum_{j=2} \left| j - 1 - \|\gamma^j\|_0 \right| \implies \sum_{j=2}^{N} \|\gamma^j\|_0 = \frac{3N - N^2}{2} - k$$

we can again relax the ℓ_0-norm and get

$$\sum_{j=2}^{N} \sum_{i<j} \kappa(\mu_i, \mu_j) = \frac{3N^2 - N^2}{2} - k$$

Finally we can apply a Lagrange multiplier and rewrite the objective function in an unconstrained form as

$$\min_{U \in R^{n \times d}} F_\lambda(U) := \sum_{j=1}^{N} \|x_j - \mu_j\|^2 + \lambda \sum_{j=2}^{N} \sum_{i<j} w_{ij} \kappa(\mu_i, \mu_j).$$

This expression is equivalent for some $\lambda > 0$, where μ_i is the ith column of U, d is the dimension of the data, and $w_{ij} = \exp\left(-\gamma \|x_i - x_j\|^2\right)$ are fixed weights. This expression is convex for any $\kappa(x, y) = \|x - y\|_p$ for any convex norm, which yields the sum-of-norms (SON) clustering method.

One of the main advantages of this convex form is it allows us to tweak the objective function in order to exploit structure.

General Objective Function

$$\min_{U \in \mathbb{R}^{n \times d}} F_\lambda(U) := \sum_{j=1}^{N} f(x_j - \mu_j) + \lambda \sum_{j=2}^{N} \sum_{i<j} w_{ij} \kappa(\mu_i, \mu_j)$$

Fidelity Terms

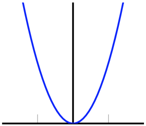
$$f = \begin{cases} \|\cdot\|_2^2 \\ h_\alpha(\cdot) \end{cases}$$

Regularization Terms

$$\kappa = \begin{cases} \|\cdot\|_1 \\ \|\cdot\|_2 \end{cases}$$

Fidelity Terms

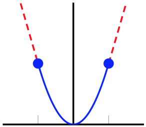
Quadratic Penalty



Penalizes the outliers quadratically

$$f(z) = \frac{1}{2}\|z\|^2$$

Huber Penalty



Huber penalty preferred since it is more robust to outliers

$$h_\alpha(z) = \begin{cases} |z|^2 & \text{if } |z| \leq \alpha \\ 2\alpha|z| - \alpha^2 & \text{if } |z| > \alpha \end{cases}$$

For computational purposes its helpful to rewrite the nested sum with a linear operator,

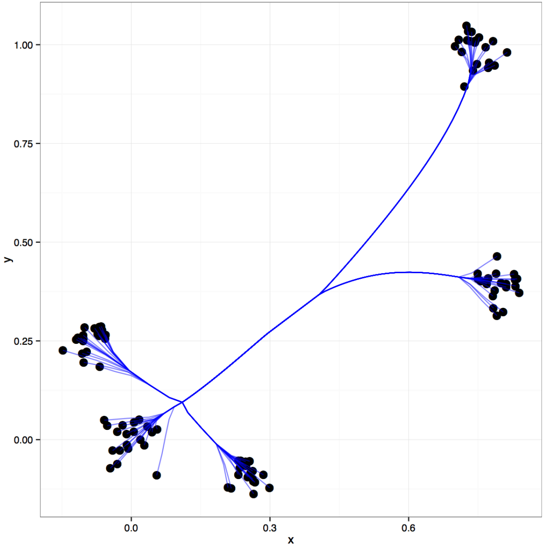
$$Q = \begin{pmatrix} 1 & -1 & 0 & \cdots & \cdots & 0 \\ 1 & 0 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 1 & 0 & \cdots & \cdots & -1 & 0 \\ 1 & 0 & \cdots & \cdots & 0 & -1 \\ 0 & 1 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & 1 & -1 \end{pmatrix}$$

CVX Implementation (Quadratic)

```
%%%%%%%%%%%%% solve convex cluster formulation
  %%%%%%%%%%%%%
cvx_begin
variable mu1(d,N)
minimize(sum(sum((x-mu1).*(x-mu1))) ...
         +lambda*dot(weights_Q,norms(Q*mu1',2,p)))
cvx_end
```

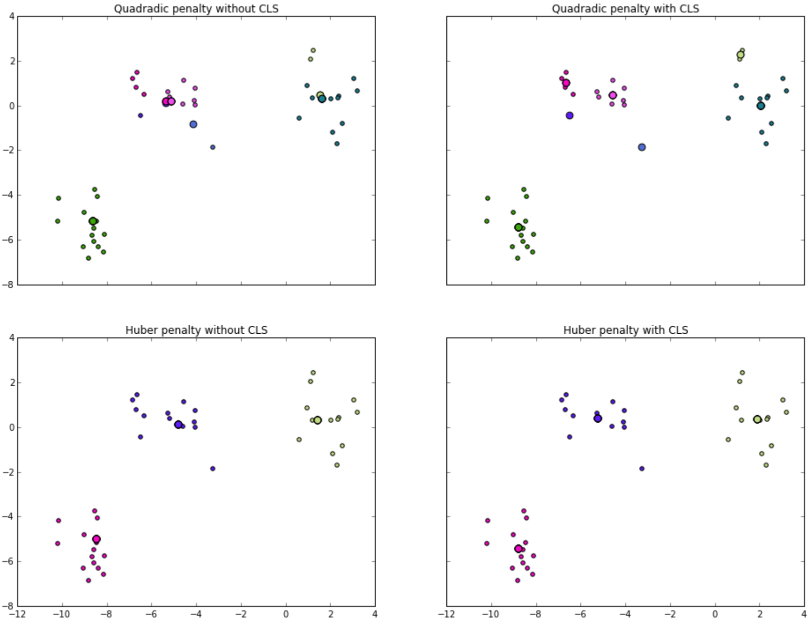
CVX Implementation (Huber)

```
%%%%%%%%%%%%% solve convex cluster formulation
  %%%%%%%%%%%%%
cvx_begin
variable mu1(d,N)
for i =1:N
    g = g + sum(huber(x(:,i)-mu1(:,i)))
end
h = lambda*dot(weights_Q,norms(Q*mu1',2,p))
g = g+h
minimize(g)
cvx_end
```


1.00
0.75
0.50
0.25
0.00
y
0.0
0.3
0.6
x

CVX Implementation (CLS)

```
%%%%% constrained least squares %%%%%%
cvx_begin
variable mu2(d,N)
minimize(sum(sum((x-mu2).*(x-mu2))))
subject to
Q(find(norms(Q*mu1',2,p)<eps),:)*mu2'==0
cvx_end
```


Quadradic penalty without CLS
Quadradic penalty with CLS
Huber penalty without CLS
Huber penalty with CLS
4
2
0
−2
−4
−6
−8
−12
−10
−8
−6
−4
−2
0
2
4

Conclusions

Kmeans

- It is fast.
- Requires the number of clusters as input.
- Sensitive to initial conditions.

Convex Formulation

- Only one global minimum.
- Requires a parameter to tune the number of clusters.
- Many ways to exploit structure.
- Has many other parameters that need tuning.

Challenges

- How can we quantitatively evaluate performance?
- How can we optimize parameters without performance metric?
- How can we build scalable algorithms?

Questions?

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problem. *SIAM Journal of Imaging Sciences*, 2(1):183-202, 2009.

F. Lindsten and H. Ohlsson. Just Relax and Come Clustering! A Convexification of k-Means. *Tech. rep., Linko pings universitet*