K-means Clustering of Proportional Data Using L1 Distance

Hisashi Kashima
IBM Research, Tokyo Research Laboratory

Jianying Hu
Bonnie Ray
Moninder Singh
IBM Research, T.J. Watson Research Center
We propose a new clustering method for proportional data with the L1 distance

- K-means clustering
- K-means clustering of proportional data with the L1-distance
- Motivation of L1-proportional data clustering: Workforce management
- An efficient sequential optimization algorithm
- Experimental results with real world data sets
Review of \( K \)-means clustering

- \( K \)-means clustering
  - partitions \( N \) data points \( \{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\} \) into \( K \) groups
  - obtain \( K \) centers \( \{\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(K)}\} \)

- Iteration:
  1. Assign each data point \( x^{(i)} \) to its closest cluster
     \[
     \pi^{(i)} := \arg\min_{j} D(x^{(i)}, \xi^{(j)})
     \]
  2. Estimate the \( j \)-th new centroid by
     \[
     \xi^{(j)} := \arg\min_{\xi} \sum_{i: \pi^{(i)} = j} D(x^{(i)}, \xi)
     \]
     - where \( D(\cdot, \cdot) \) is a distance measure between two vectors

\( K \)-means Clustering of Proportional Data Using L1 Distance
**K-means clustering of proportional data**

- **K-means clustering of proportional data**
  - partitions $N$ proportional data points $\{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\}$ into $K$ groups
  - obtain $K$ centers $\{\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(K)}\}$

- A proportional data is a $M$-dimensional vector $x^{(i)} = (x_{1}^{(i)}, x_{2}^{(i)}, \ldots, x_{M}^{(i)})$
  - each element is non-negative $x_{d}^{(i)} \geq 0$
  - elements are summed to be one, i.e. $\sum_{i=1}^{M} x_{d}^{(i)} = 1$

- Cluster center also must satisfy the constraints $\xi_{d}^{(j)} \geq 0$ and $\sum_{d=1}^{D} \xi_{d}^{(j)} = 1$

---

**K-means Clustering of Proportional Data Using L1 Distance**
We use the L1 distance, but why?

Application to Skill Allocation-Based Project Clustering

- We concentrate on the L1-distance as the distance measure

\[
D(x^{(i)}, \xi^{(j)}) := \sum_{d=1}^{M} |x_d^{(i)} - \xi_d^{(j)}|
\]

- Our motivation: Workforce management
  - We want staffing templates for assigning skilled people to various projects
  - A template indicates how much % of the whole project time is charged to a particular job role
    - e.g. “software installation” template = (consultant=0.1, engineer=0.9, architect=0.0)
    - used for efficient assignment of appropriate people to appropriate project
    - used as bases of skill demand forecasting
  - The L1 distance from templates can be directly translated into cost differences.
    - Also allows different skills associated with different costs

- L1-distance is known to be robust to noise
Challenge of using L1 distance for $K$-mean clustering of proportional data

- For L2 distance, the closed form solution is obtained as the mean
  - regardless whether the proportional constraints apply
- For L1 distance,
  - the median is the closed form solution for the unconstrained case
  - With proportional constraints, no closed form solution exists
- There are two fast approximations for constrained L1 $K$-means:
  1. Use the mean (just like L2 $K$-means)
  2. Median followed by normalization

- Challenge: can we find an efficient way to compute accurate solutions?
Algorithm for K-means clustering with proportional data

- **K-means clustering of proportional data**
  - partitions N data points \( \{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\} \) into K groups
  - obtain K centers \( \{\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(K)}\} \)

- **Iteration:**
  1. Assign each data point \( x^{(i)} \) to one of K clusters
     \[
     \pi^{(i)} := \arg\min_j \mathcal{D}(x^{(i)}, \xi^{(j)})
     \]
  2. For the \( j \)-th cluster, estimate a new centroid by
     \[
     \xi^{(j)} := \arg\min_{\xi} \sum_{i: \pi^{(i)} = j} \mathcal{D}(x^{(i)}, \xi) \quad \text{s.t.} \quad \xi^{(j)} \geq 0 \quad \text{and} \quad \sum_{d=1}^{M} \xi^{(i)} = 1
     \]
     where \( \mathcal{D}(\cdot, \cdot) \) is the L1-distance: \( \mathcal{D}(x^{(i)}, \xi) := \sum_{d=1}^{M} |x^{(i)}_d - \xi_d| \)
The key point is how to solve step 2 efficiently

- The optimization problem involved in Step 2 is

$$\xi(j) := \arg\min_\xi \sum_{d=1}^M \sum_{i:\pi(i)=j} |x_d^{(i)} - \xi_d| \quad \text{s.t.} \quad \xi_d \geq 0 \quad \text{and} \quad \sum_{d=1}^M \xi_d = 1$$

- The equivalent linear programming problem has $O( \#\text{data points} \times \#\text{dimensions} )$-variables

- But we want a more efficient method tailored for our problem using the equality constraint explicitly
Our approach: sequential optimization w.r.t. 2 variables

- Key observation: we have only one equality constraint

\[
\xi(j) := \arg\min_{\xi} \sum_{d=1}^{M} \sum_{\hat{j} : \pi(i) = \hat{j}} |x_{d}^{(i)} - \xi_{d}| \quad \text{s.t.} \quad \xi_{d} \geq 0 \quad \text{and} \quad \sum_{d=1}^{M} \xi_{d} = 1
\]

- We employ sequential optimization borrowing the idea of SMO algorithm for SVM (QP)
  - Picks up two variables at a time and optimizes w.r.t. the two variables

- Iteration:
  1. find a pair of variables \( \xi_{d} \) and \( \xi_{d'} \) which improves the solution the most
  2. Optimize the objective function with respect only to them (while keeping the equality constraint satisfied)
How to select the two variables?

- Key observation: The objective function is decomposed into piecewise linear convex functions with only one parameter

\[ \xi^{(j)} := \arg\min_{\xi} \sum_{d=1}^{M} f_d(\xi_d), \ f_d(\xi_d) := \sum_{i: \pi(i) = j} |x_d^{(i)} - \xi_d| \]

- If we decrease \( \xi_d \), \( \xi_{d'} \) will be increased
- Find a pair of variables \( \xi_d \) and \( \xi_{d'} \) which has the steepest gradient
  - Found in O(log \( M \)) time by efficient implementations

\[ g(\xi_d, \xi_{d'}) := g^-(\xi_d) - g^+(\xi_{d'}) \]

Gradient wrt the change

Left gradient of \( f_d(\xi_d) \)

Right gradient of \( f_{d'}(\xi_{d'}) \)
How much do we update the two variables?

- Update $\xi_d$ and $\xi_{d'}$ while keeping the constraint.
- If we decrease $\xi_d$ by $\Delta$, $\xi_{d'}$ increases by $\Delta$.
- Move the two variables until either of them reaches a corner point.

$$
\xi_d := \xi_d - \min\{\Delta_d, \Delta_{d'}\}
$$

$$
\xi_{d'} := \xi_{d'} + \min\{\Delta_d, \Delta_{d'}\}
$$

- The number of updates is bounded by the number of corners.

The diagram illustrates the decrease of $\xi_d$ and the increase of $\xi_{d'}$.
Experiments

- Two real world datasets representing skill allocations for past projects in two service areas in IBM
  - 1,604 project with 16 skill categories
  - 302 projects with 67 skill categories
- We compared three algorithms
  - The proposed algorithm
  - “Normalized median”
    1. Computes dimension-wise medians
    2. Normalizes them to make the sum to be one
  - “Mean”
    • Uses sample means as cluster centroids
      (The equality constraint is automatically satisfied)
The proposed method achieves low L1 errors

- 10-fold cross validation × 10 runs with different initial clusters
- Performances are evaluated by sum of L1 distances to nearest clusters
- The proposed algorithm consistently outperforms both alternative approaches at all values of $K$

![Graph showing the performance of the proposed method against alternative approaches for different numbers of clusters.](image-url)
The proposed method produces moderately sparse clusters

- The proposed method leads to more interpretable cluster centroids
- “Normalized Median” produces many cluster centroids with only a single non-zero dimension
Conclusion

- We proposed a new algorithm for clustering proportional data using L1 distance measure
  - The proposed algorithm explicitly uses the equality constraint

- Other applications include
  - document clustering based on topic distributions
  - video analysis based on color distributions