SPR Day 12: K-Means

Read: PRML textbook 9.1 K-Means

Goals: (1) Understand problem setup
cost function
optimization problem

(2) Understand coord. descent approach

(3) Look ahead
- scaling to large data
- handling other distances
- towards probabilistic approach

K-Means Clustering

Problem Statement

Given: N observed data vectors, each D-dim.

$\{x_n\}_{n=1}^{N}$ where $x_n \in \mathbb{R}^D \ \forall n$

Desired number of clusters K (integer, $K>0$)

Goal: Determine locations of K cluster centers

$\{\mu_k\}_{k=1}^{K}$ where $\mu_k \in \mathbb{R}^D \ \forall k$

such that across all datapoints, the distance between each point and its nearest center is minimized

We want a "low cost" clustering!

Cost function:

$$\text{cost}(x,\mu) = \sum_{n=1}^{N} \min_{k \in \{1,2,\dots K\}} \text{dist}(x_n, \mu_k)$$

$$= \sum_{n=1}^{N} \min_{k} \|x_n - \mu_k\|_2^2 \quad \text{Euclidean distance}$$

$$= \sum_{n=1}^{N} \min_{k} \sum_{d} (x_{nd} - \mu_{kd})^2$$

Alternative: Can write in a way to make solving easier

Let r_n be a length K "one hot" vector indicating the closest cluster to example x_n

Can write $\text{cost}(x, r, \mu) = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|_2^2$

KMeans Coordinate Descent

e.g. if K=4, r_n could be:

1	2	3	4
0	0	0	1
0	0	1	0
0	1	0	0
1	0	0	0

Goal: Find $r = \{r_n\}_{n=1}^N$, $r_n \in \text{onehot}(K)$

and $\mu = \{\mu_k\}_{k=1}^K$, $\mu_k \in \mathbb{R}^D$

that minimize cost

$$\text{cost}(x, r, \mu) = J(x, r, \mu) = \sum_n \sum_k r_{nk} \|x_n - \mu_k\|_2^2$$

Optimization Problem:

$$\min_{\substack{\{r_n\}_{n=1}^N, r_n \in \text{onehot}(K) \\ \mu \in \mathbb{R}^{K \times D}}} J(x, r, \mu)$$

How to solve? What algorithm?
Can we just do gradient descent? No! r is not continuous

Look at coordinate descent.
Iteration t has two steps.

Step 1: Fix $\mu = \mu^{t-1}$. Find optimal r given μ^{t-1}.

$$r^t = \min_{\{r_n\}_{n=1}^N} J(x, r, \underbrace{\mu^{t-1}}_{\text{fixed}})$$

Step 2: Fix $r = r^t$. Find optimal μ given r^t.

$$\mu^t = \min_{\mu \in \mathbb{R}^{K \times D}} J(x, r^t, \mu)$$

Solving Assignment Step

Step 1: $$\min_{\{r_n\}_{n=1}^N} \sum_n \underbrace{\sum_k r_{nk} \|x_n - \mu_k\|_2^2}_{f^n(r_n)}$$

Because objective is a sum over N separate terms this looks like

$$\min_{\{r_n\}_{n=1}^N} f^1(r_1) + f^2(r_2) + \cdots f^N(r_N)$$

and we can solve each term separately!

for each n: $$\min_{r_n \in \text{onehot}(K)} \sum_k r_{nk} \|x_n - \mu_k\|_2^2$$

We can show that optimal update is:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \underset{k \in \{1,2,\ldots K\}}{\text{argmin}} \|x_n - \mu_k\|_2^2 \\ 0 \end{cases}$$

Proof by contradiction. Suppose we set $r_{nj} = 1$, but there exists $k \neq j$ where $\|x_n - \mu_k\| < \|x_n - \mu_j\|$.

Then we can improve cost, and thus r_{nj} not optimal (by setting $r_{nk} = 1$ instead)

Solving Mean Update Step

Step 2: $\min_{\mu \in \mathbb{R}^{K \times D}} \sum_{k=1}^{K} \sum_{n=1}^{N} r_{nk} \|x_n - \mu_k\|_2^2$

Again, can view as K separate functions each involving only $\mu_k \in \mathbb{R}^D$

For each k: $\min_{\mu \in \mathbb{R}^D} \sum_n r_{nk} \|x_n - \mu_k\|_2^2 = \underbrace{\sum_n r_{nk} (x_n - \mu_k)^T (x_n - \mu_k)}_{\ell(\mu_k)}$

Solve by: take derivs, set to zero, find optimal μ_k

$$0 = \nabla_{\mu_k} \ell(\mu_k) = \nabla_{\mu_k} \sum_n r_{nk} \left[\cancel{x_n^T x_n} - 2 x_n^T \mu_k + \mu_k^T \mu_k \right] \quad \text{(const so } \emptyset\text{)}$$

$$0 = \text{"} = \sum_n r_{nk} \left[\emptyset - \nabla_{\mu_k} (2 x_n^T \mu_k) + \nabla_{\mu_k} (\mu_k^T \mu_k) \right]$$

$$0 = \sum_n r_{nk} \left[-2 x_n + 2 \mu_k \right]$$

$$\sum_n r_{nk} x_n = \left(\sum_n r_{nk} \right) \mu_k$$

$$\boxed{\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}}$$

Set location of cluster k to the mean of all examples assigned to k

Edge case: avoid divide by zero when $\sum_n r_{nk} = \emptyset$

Iterative algorithm of K-Means

Coordinate descent algo.

Input: $\{x_n\}_{n=1}^N$ (examples)

K (num clusters)

$\mu^0 = \{\mu_k\}_{k=1}^K$ (initial cluster locations)

Procedure:

Repeat for iters $t = 1, 2, \dots T$

step 1 [for each n in $1, 2, \dots N$:

Update $r_{nk}^t \leftarrow \begin{cases} 1 & \text{if } k = \operatorname{argmin} \|x_n - \mu_k^{t-1}\|_2^2 \\ 0 & \text{o.w.} \end{cases}$

step 2 [for each k in $1, 2, \dots K$:

Update $\mu_k^t \leftarrow \dfrac{\sum_n r_{nk}^t x_n}{\sum_n r_{nk}^t}$

Big Idea: Each step guaranteed to lower cost

$$J(x, r^1, \mu^0) \geq J(x, r^1, \mu^1) \geq J(x, r^2, \mu^1) \geq \cdots J(x, r^T, \mu^T)$$

after iter 1 step 1 — after iter 1 step 2 — after iter 2 step 1 — ... — after iter T step T

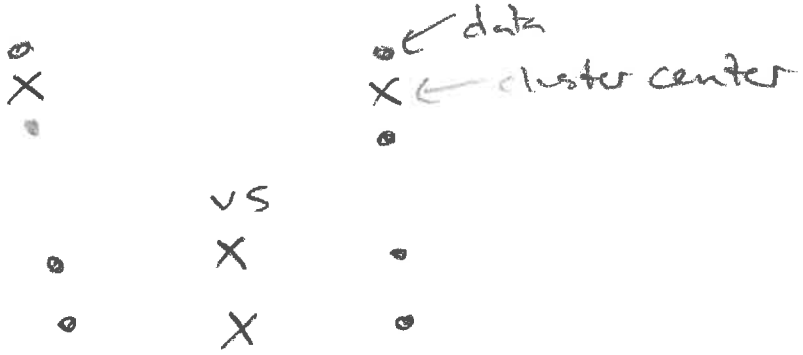
Will eventually converge (stop changing: $r^{t+1} = r^t$ for some t).

However, will it yield a global optima? Not always

Demos of KMeans

See slides

- illustration that local optima exist



Scaling to Large Datasets

Can parallelize many steps

e.g. solve for r_1, r_2, ..., r_N each on separate machines

Can process a few examples at a time

option 1: SGD approach

for each minibatch X^b:

E) find optimal assignments r_n^b given μ^{t-1}

M) take gradient step

$$\mu^t \leftarrow \mu^{t-1} - \epsilon \nabla J(x^b, r^b, \mu^t)$$

option 2: incremental or memoized approach

Store assignment statistics for each batch b

$N_{bk} = \sum_n r_{nk}$

$S_{bk} = \sum_n r_{nk} x_n$

for each minibatch x^b:

update local statistics

N_{bk}^{new}

S_{bk}^{new}

increment global stats

$N_{0K} = N_{0K} - N_{bk}^{old} + N_{bk}^{new}$

update $\mu_k = S_{0k} / N_{0K}$