Data Stream Sampling With Online Algorithms

Shengdong Zhang
Student I.D. : 301247751
Department of Computing Science, Simon Fraser University, BC, Canada

INTRODUCTION

Clustering can be used to detect the structure hidden in data, discovering the similarity among instances. In fact, we can apply clustering as an add-on step of data stream sampling to improve the quality of samples taken from a data stream. In addition, we are also able to better estimate probability density based on the information from clustering. Since clustering is unsupervised, we can simply add an online version of a clustering algorithm to any data stream processing systems then train it on-the-fly. The reason why we choose online algorithms is that they cost much less storage space than those off-line ones, and usually converge faster and give better solutions. These are big advantages in the context of processing data streams.

A Motivating Example

A search engine receives a large stream of queries from different types of users. When processing this query stream, usually only one pass is allowed. If we first cluster the observed queries to categorize the users first, then apply some machine learning techniques to discover those common interests within users of the same category, we are able to help the search engine improve the quality of retrieved results later. Also, since the distribution of clusters could be very skewed, if we can use clustering to estimate probability density of the query stream, we are able to give estimation of some statistics with lower variances (e.g. what keywords are expected in a query), or maintain representative sample set of a fixed size (e.g. it contains 100 samples and each observation is from any one of the clusters with their prior probabilities).

PROBLEM DEFINITION AND ALGORITHMS

The proposed algorithm has 3 main steps:

- Inverse Weighted Online K-Means Algorithm (IWK)
- Online Stepwise EM Algorithm (sEM) for estimating parameters of mixture Gaussian distribution
- Stratified Sampling and Pseudo-MCMC Sampling on Data Stream

The first step involves a better version of online K-means. This algorithm plays as the role of cluster center locator. The cluster centers found will be used to activate the training of sEM, which is the second component. More details about these two algorithms will be shown in later section. After the first two steps, at any point of time, the algorithm will keep a fixed-size representative set of observations, or be asked to produce an estimate of a statistic given the number of samples allowed.

In this project, two assumptions are made: the number of clusters is known; each observation in data stream is independent and from a mixture of multiple Gaussian distributions. For better visualization, artificial data sets are generated in 2 dimensional space, and one benchmark high dimensional dataset will be used for experiment.

Inverse Weighted Online K-Means Algorithm

Traditional k-means algorithm has one big weakness: final solution is highly sensitive to the initialization of cluster centers. Since the training stage is a local search process, a cluster center (mean) only sees those data nearest to it and simply ignore the others [Barbakh, 08]. IWK is an improved version of k means —— instead of letting a cluster center sees locally, we let it see globally, by weighting all distances between it and all data points. Below is the reconstruction error function of IWK algorithm. Let $X = \{\vec{x}_1, \ldots, \vec{x}_N\}$ and $\vec{m}_i$'s be the means of clusters.

$$E(\{\vec{m}_i\}_{i=1}^K | X) = \sum_{i=1}^N [\sum_{j=1}^K \frac{1}{||\vec{x}_i - \vec{m}_j||^p}] min_{k=1}^K ||\vec{x}_i - \vec{m}_k||^n$$

By taking the partial derivative of this function with respect to the means, we can easily derive the online updating rules for all $\vec{m}_i$'s. The parameters n and p control the correction rate and stability of the updating process. For more details, please read [Barbakh, 08]. Below is the IWK algorithm with $p = -1$ and $n = 1$.

1. Initiate all cluster centers $\vec{m}_i$'s
2. For each data sample $\vec{x}_i$ that arrives:
   - Let $k^* = arg\, min_{k=1}^K (||\vec{x}_i - m_k||)$
   - Update all $\vec{m}_i$'s as follows

$$\vec{m}_{k^*}^{new} = \vec{m}_{k^*}^{old} - \zeta a_{ik^*}(\vec{x}_i - \vec{m}_{k^*}^{old})$$
$$\vec{m}_k^{new} = \vec{m}_k^{old} - \zeta a_{ik}(\vec{x}_i - \vec{m}_k^{old})$$

   where

$$\begin{aligned} a_{ik^*} &= -(n+1)||\vec{x}_i - \vec{m}_{k^*}||^{n-1} - n||\vec{x}_i - \vec{m}_{k^*}||^{n-2} \sum_{j \neq k^*} ||\vec{x}_i - \vec{m}_j|| \\ a_{ik} &= \frac{-||\vec{x}_i - \vec{m}_{k^*}||^n}{||\vec{x}_i - \vec{m}_k||} \end{aligned} \tag{1}$$

In the updating formulas, ζ is the learning rate. Setting ζ is an important issue and we will get back to it in the Experiments session later. This algorithm performs very well on the task of fast locating cluster centers (means) in terms of space cost and time cost. As we can see, the main computational cost for one data point is finding the nearest cluster center for it and computing a_{ik^*}. For one iteration, the time complexity for this two parts is simply $O(K)$, where K is the number of clusters. Also, the space complexity of the algorithm is $O(K)$ as well. This algorithm is much more robust than usual online k-means algorithms, for it is less sensitive to poorly initialized cluster centers.

Online Expectation Maximization Algorithm

Online EM algorithms have two variants: incremental EM and stepwise EM. In this model, the latter one is chosen, for it does not need to store any sufficient statistics like the first variant, which is again one big advantage for processing data streams. The sEM algorithm is a stochastic approximation type EM. It makes approximation of distribution parameters by one or a few samples, and then interpolate between the existing parameters and the approximated ones [Liang, 2009]. Below is the framework of sEM.

1. Initiate all cluster centers $\vec{m}_i$'s
2. For M observation(s) $\vec{x}_i$ that arrives:
   - E Step: Compute cluster membership probability p_{ik}, which is posterior probability, for each cluster center k
   - M Step: Update all cluster centers $\vec{m}_i$'s and their sample covariance matrices S_i with the membership probabilities. Also, update all prior cluster probability estimation p_k.

Because we assume Gaussian mixture distribution, we can derive the following updating rules.

E Step:

$$p_{ik} = \frac{p_i det(S_k)^{-1/2} exp[-(1/2)(\vec{x}_i - \vec{m}_k)^T S_k^{-1}(\vec{x}_i - \vec{m}_k)]}{\sum_{j=1}^{K} p_j det(S_j)^{-1/2} exp[-(1/2)(\vec{x}_i - \vec{m}_j)^T S_j^{-1}(\vec{x}_i - \vec{m}_j)]} \tag{2}$$

M Step:

$$\begin{aligned} \vec{m}_k^{new} &= (1-\gamma_N)\vec{m}_k^{old} + \gamma_N \frac{\sum_{i=1}^{M} p_{ik}\vec{x}_i}{\sum_{i=1}^{M} p_{ik}} \\ S_k^{new} &= (1-\gamma_N)S_k^{old} + \gamma_N \frac{\sum_{i=1}^{M} p_{ik}(\vec{x}_i - \vec{m}_k^{new})(\vec{x}_i - \vec{m}_k^{new})^T}{\sum_{i=1}^{M} p_{ik}} \\ p_k^{new} &= \frac{N p_k^{old} + \sum_{i=1}^{M} p_{ik}}{N+M} \end{aligned} \tag{3}$$

N is the number of total observations before the update, and M is the number of observations used for the update. N does not include the M observations. γ_N is the momentum for the update. It is also called step size in some literature. When $M = 1$, sEM will be updated by just one sample, which will cause unstability in the training. To avoid this issue, M is usually set to some number greater than 1 so the quality of estimation can be improved. Thus, M is called mini-batch size. For efficiency of this algorithm, M should be a small number.

Nothing needs to be computed for the initialization of cluster centers. They are the output of the previous IWK algorithm. However, we need the sample covariance matrices for these clusters before we run sEM. For edge-cut connection between these two algorithm, when training IWK, for each cluster, the latest 20 samples nearest to it will be stored for computing the covariance matrix of this cluster. In addition, it is crucial for the selection of M and γ_N. We will discuss it in the Experiments session.

Both time and space complexity should be $O(K)$. However, when the dimensionality of data is large, the essential computational burden would be the calculation of the determinant and the inverse of S_k.

Stratified Sampling and Pseudo MCMC Sampling

Stratified Sampling

Clustering can be regarded as one way of stratification. As long as data points are well clustered, we can expected variance within a cluster relatively smaller than the variance of population. There is a big advantage of using stratified sampling after clustering. Below is the theoretical justification.

Suppose we are estimating the mean of a random variable X, which is $E[X]$ and each observation of this random variable is i.i.d.. There is also a discrete random variable C with K possible outcomes (clusters) $c_1, \ldots, c_K$, and corresponding probabilities $p_1 = p(C = c_1), \cdots, p_2 = p(C = c_K)$. If we estimate the mean by taking average of samples, say, $\bar{X}$, the variance will be $Var(\bar{X}) = \frac{1}{N} Var(X)$, where N is the number of samples. On the other hand, $E[X]$ can be estimated by conditioning on C such that

$$E[X] = \sum_{i=1}^{k} E[X|C = c_i]p(C = ci)$$

So the estimator will be $\bar{X}_{SS} = \sum_{i=1}^{K} \bar{X}_i p_i$, where $\bar{X}_i$ is the estimator of mean conditioned on $C = c_i$. It is unbiased. Therefore, the conditional variance of $\bar{X}_i$ will be $Var(\bar{X}_i) = \frac{Var(X|C=c_i)}{Np_i}$. Since all samples are i.i.d., we have

$$Var(\bar{X}_{SS}) = \sum_{i=1}^{K} p_i^2 Var(\bar{X}_i) = \frac{1}{N}\sum_{i=1}^{K} p_i Var(X|C = c_i) = \frac{1}{N}E[Var(X|C)].$$

Because for any random variable X, $Var(X) = E[Var(X|C)] + Var[E(X|C)]$, we can see that

$$NVar(\bar{X}) = NVar(\bar{X}_{SS}) + Var[E(X|C)]$$

$$Var(\bar{X}) - Var(\bar{X}_{SS}) = \frac{1}{N} Var[E[X|C]] \geq 0$$

Therefore, if the random variable C has large effect on the random variable X, the stratifying estimator X_{SS} will have considerably smaller variance than the one of $\bar{X}$. Even in the worst case that C does not have any effect on X, introducing C will not introduce additional variance. After EM clustering, we will know the cluster centers in the data space and the cluster probabilities, so we can compute the stratifying estimator of population mean based on a given number of samples and the cluster membership of them. Below is the algorithm, assuming $p_1, \ldots, p_k$ are the cluster prior probabilities estimated from the previous EM clustering step, and the allowed sample size is N.

1. Store the latest N observations $y_1, \cdots, y_N$;
2. For each observation, find which cluster is nearest to it;
3. For each cluster i, compute the mean of observations belonging to it, denoted by $\bar{X}_i$;
4. Compute the stratifying estimator $\bar{X}_{SS} = \sum_{i=1}^{K} \bar{X}_i p_i$.

Also, we can directly get the population center by computing the weighted sum of cluster centers with prior cluster probabilities as weights, which is the same as stratified sampling.

Semi-pseudo Metropolis-Hastings Sampling

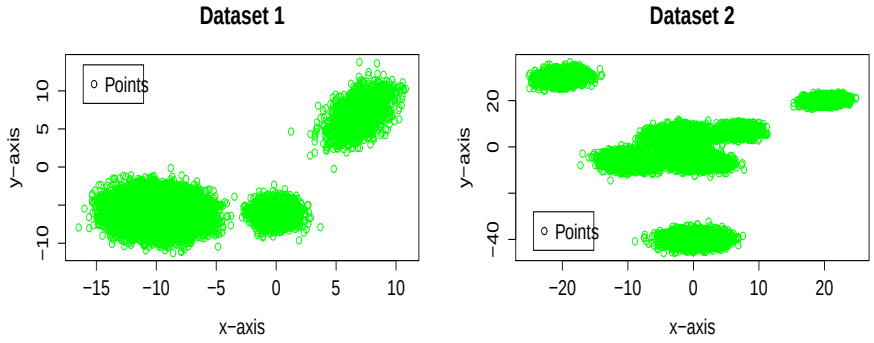
This sampling method is also based on the the cluster prior probabilities $p_1, \ldots, p_K$ from the previous EM step. It maintains a sample set of size N such that each observation is from any one of the clusters proportional to the cluster prior probabilities. Below is the algorithm:

1. Collect N latest observations as a sample set
2. Repeat Until be asked to output the sample set
   - Read next observation x and find c_{new} = cluster of x
   - Generate a random number $U \sim Uniform(0, 1)$
   - If $U < \frac{p_{c_{old}}}{p_{c_{new}}}$
     - Select a cluster i with a probability $p =$ (cluster's number of occurrences in the sample set)/(number of clusters). If $p \leq p_i$, re-select a cluster
     - Select an observation belonging to the selected cluster, replace it with x
     - $c_{old} = c_{new}$

This algorithm intimates the Metropolis-Hastings algorithm with independence chain given by the cluster prior probabilities. We use uniform target distribution of clusters, so the terms of target distribution density got canceled out in the acceptance probability. By doing so, the sampling algorithm usually accepts observations from rarely seen clusters, and selectively reject observations from frequently occurring clusters. The reason why we call it "semi-pseudo" is because it does not generate a random variable from the proposal distribution. Instead, it considers the new observation as the generated random variable from that distribution. It would be a reasonable assumption when the estimated prior cluster distribution is close to the true one.

EXPERIMENTS

Experimental Setup

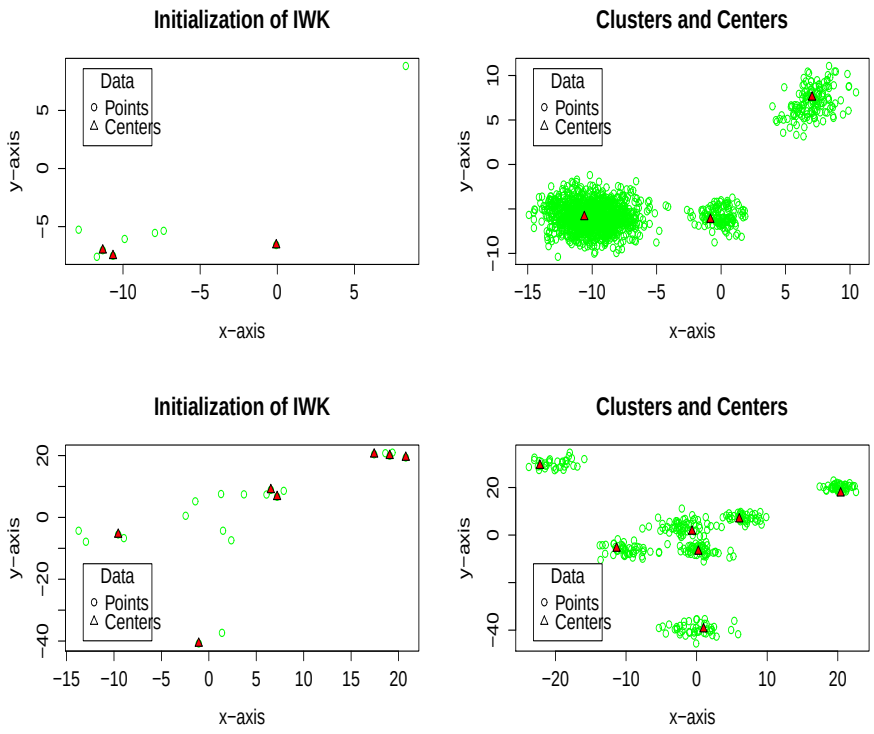


Each dataset has 20000 data points. The first dataset has 3 clusters, and it is designed to be very imbalanced. About $3/4$ of its points belong to the leftmost cluster, and the other two clusters almost equally share the rest. The second dataset has 7 clusters and each data point is generated to be in one of the clusters with probabilities $[1/8, 1/6, 1/6, 1/6, 1/6, 1/12, 1/8]$. For each data set, we will use first 10000 data points for training the online clustering algorithms, and use the rest 10000 points to test the performance of stratifying mean estimator and samples taken by pseudo-MCMC.

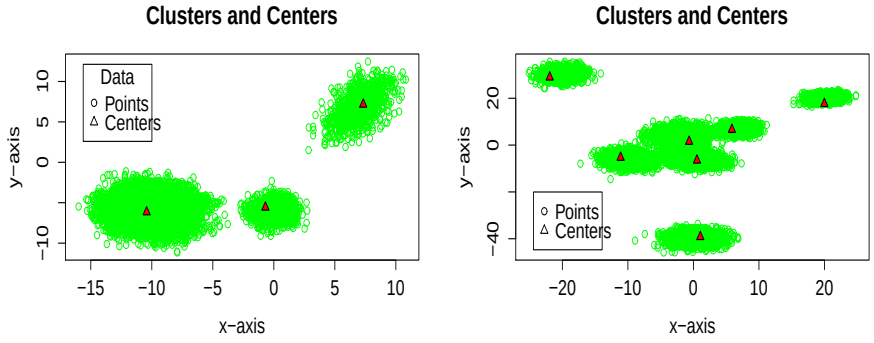
To simulate data streaming, only one data point is read and processed by the online algorithms. Since usually initiated cluster centers are poorly placed, and we use the online IWK algorithm as the initialization step for EM clustering. The learning rate ζ should be set to a relatively large number in the beginning and then decrease fast, for the centers move to the desired positions quickly and then stay. In the experiment, ζ is set to $\zeta_N = (N+2)^{-0.55}$, where N is the number of training iterations. The setting of momentum γ for online EM follows the same strategy. It is set to $\gamma_N = (N+2)^{-1}$. As to mini-batch size, for adding stability to sEM, M is set to 100.

Online IWK Algorithm

The IWK algorithm is more robust to bad initialization of centers than the k-means algorithm. However, the number of iterations needed for the centers moving to the desired positions depends on the initialization, the distribution of data points, and the dimensionality. For the first data set, it takes 1000 iterations from the initial positions (left figure below) to the good ones (right figure below). For the second data set, only after 300 iterations, all cluster centers move to good positions (bottom left and bottom right figures).



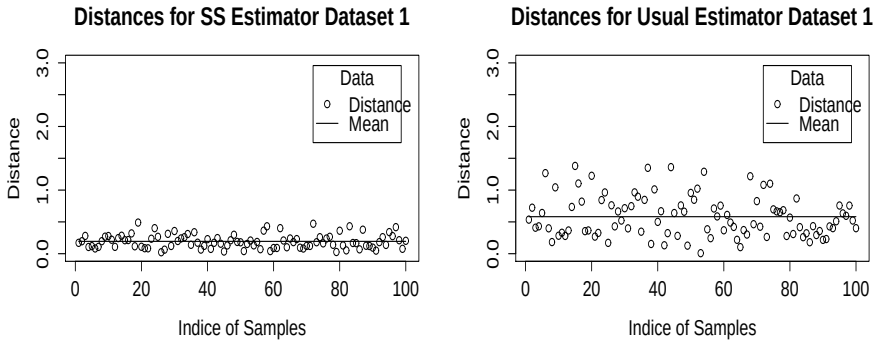
Online EM Algorithm



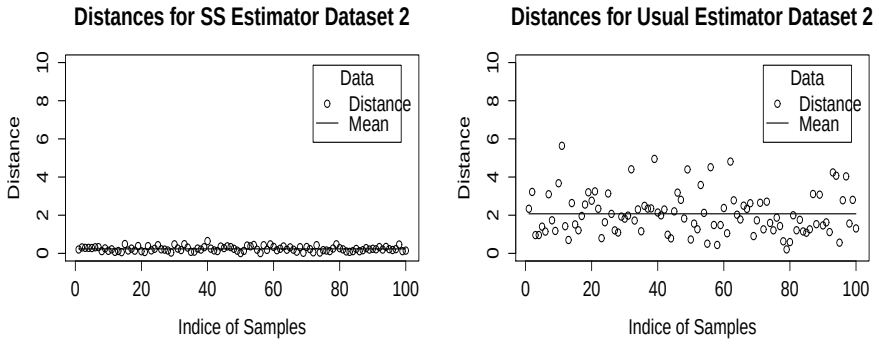

```
Data Set 1 Cluster Centers Before and After EM Clustering
           [,1]       [,2]     [,3]
[1,] -0.5964179 -10.464749 7.494664
[2,] -5.5597278  -6.180047 7.296496
           [,1]       [,2]     [,3]
[1,] -0.6904085 -10.423217 7.325995
[2,] -5.5686057  -6.154012 7.164673

Data Set 2 Cluster Centers Before and After EM Clustering
[1,]   0.9835724 20.40787  0.2423496 -22.19789 -0.653697 6.043061 -11.336300
[2,] -39.5462875 17.60861 -6.9020233  29.22050  1.529052 6.706684  -5.643664
           [,1]     [,2]       [,3]      [,4]       [,5]     [,6]       [,7]
[1,]   1.039223 19.99097  0.4343224 -21.83864 -0.6836589 5.899039 -11.068473
[2,] -39.091255 17.53132 -6.5781494  28.80810  1.4903072 6.493660  -5.443397
```

Above are the cluster centers found by online sEM for the first 10000 points of each data set. Because sEM directly takes cluster centers from IWK which are well placed already, the final centers don't change too much, which is shown by their coordinates above.

Stratified Sampling

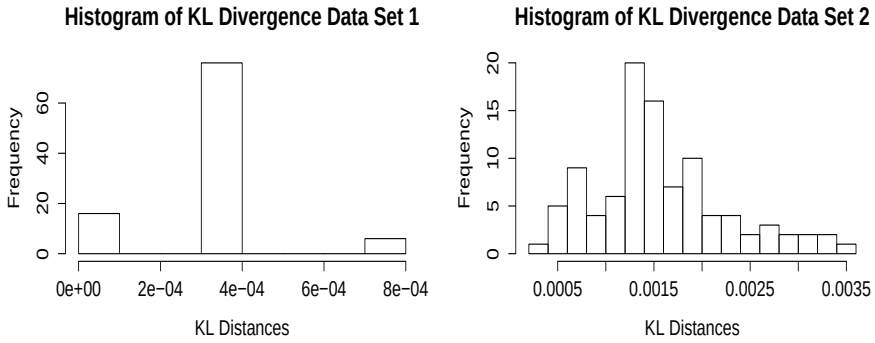


To test the performance of the stratifying sampling estimator, every 100 points after the first 10000 points are collected as a sample from the data stream. First, usual mean estimator $\bar{X} = \frac{1}{100}\sum_{i=1}^{100} X_i$ and stratifying sampling estimator $\bar{X}_{SS} = \sum_{i=1}^{K} \bar{X}_i p_i$ will be computed. Next, the distances between these two estimated means and the population mean will be computed. The plots in this session show the distance for each sample. We can see that estimated mean by stratifying sampling estimator is usually closer to the true mean. The variances for these estimators are:



Data Set 1: $Var(\bar{X}) = 0.587726$, $Var(\bar{X}_{SS}) = 0.1958925$.
Data Set 2: $Var(\bar{X}) = 2.095142$, $Var(\bar{X}_{SS}) = 0.2373381$.

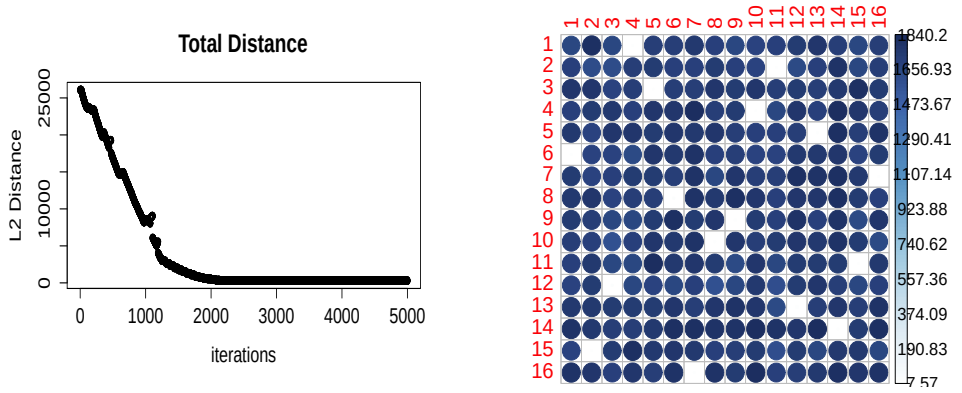
Pseudo MCMC Sampling



The plot above shows the KL distances between the prior cluster probability distribution and the sampling distributions of 100 outputs randomly taken from the algorithm for each data set. We can see that the sample set maintained by the algorithm usually has very small KL distance from the prior one.

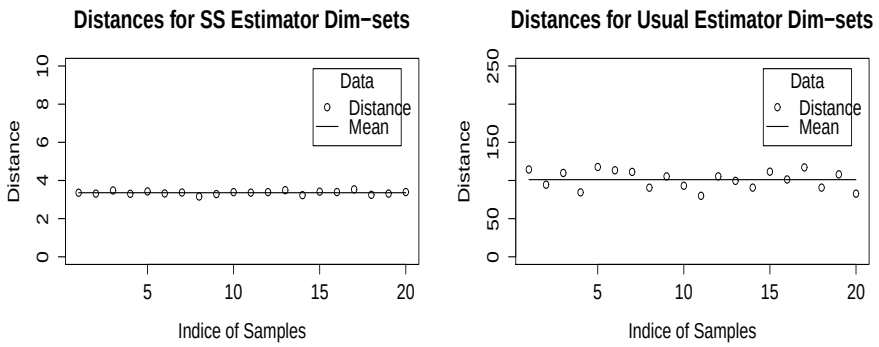
Online IWK Algorithm on High Dimensional Data Set

A benchmark high dimensional data set called DIM-sets (high) is used to test the ability of this algorithm. It has 1024 data points and the number of dimensions equals 512. It has 16 Gaussian clusters. Below are the results.



We use the total distance between estimated cluster centers and the true cluster centers to measure how good the estimation is. The left plot above shows that online IWK converges after 2000 iterations, which indicates approximately only 2 scans of the data set. Since we can not visualize 512 dimensional space, to determine how close are the estimated centers to the true centers, we visualize the Euclidean distance matrix, which is the right plot. Every entry is represented by a circle. Deeper the color, large the value. Row indexes correspond estimated centers, column indexes correspond to true centers. We can see that in each row and column, there is exactly only one hole. It is because the values corresponding to these holes are too small compared to the other, say less than 50, the shapes of circles fade out. This indicates that the algorithm successfully locate all clusters and estimated centers are all close to the true ones. Also, stratified sampling still works in this case. For this data set, only 20 sample sets are used.

DIM-sets: $Var(\bar{X}) = 106.3895$, $Var(\bar{X}_{SS}) = 3.535508$.



Conclusion and Discussion

In this project, we first applied online stepwise EM algorithm to estimate parameters of mixture Gaussian distribution with online inverse weighted k means algorithm as preparing step to cluster observed data from data streams, then use stratified sampling technique to estimate the population mean with much smaller variance, and finally use the idea from Metropolis-Hastings algorithm to keep a representative sample set of data streams.

In fact, online EM with mixture Gaussian distribution seems redundant because online IWK algorithm is able to find cluster centers nicely. In the experiments, simply using online IWK could even have better prior cluster distribution estimation, because it is hard to get very accurate estimated covariance matrix for Gaussian distribution of each cluster, and poorly estimated covariance matrices will seriously influence the estimation of prior cluster distribution. However, the point I try to make here is that online EM algorithm has the potential to estimate parameters of a distribution in an online fashion, not just for mixture Gaussian, we can do it for mixture Possions, or even for distributions from the exponential families. Once the estimation is done, surely it will enable us to do a lot of useful things, such as what I implemented in the project, or analyze data of each cluster. Also, in my opinion, using online IWK algorithm as a preparing step help online EM algorithm converge faster.

As to stratified sampling, because of the inequality derived previously, as long as the online algorithms get not-too-poor estimation of prior cluster distribution, it is able to help reduce variance of the estimator and never turn things to be worse. Not just for computing the mean of population, we can do it for computing any function of observations. Thus, clustering could always be a good preprocessing technique to use before applying stratified sampling on data stream.

The idea of semi-pseudo MCMC sampling was inspired by the idea of the Metropolis-Hastings algorithm. The reason why I called the sample set collected in this way representative is that if we randomly take all observations in a sliding window on the data stream, due to the skewness, data from rarely-observed cluster have large chance not to be included. The proposed method guarantees data points from each cluster are kept proportional to estimated prior cluster distribution.

Last thing is the change-point detection. When dealing with data streams, the distribution of observations is rarely stationary. However, as long as the distribution does not vary extremely often, it is reasonable to assume it is a stationary distribution for a short period of time then transit to other stationary distribution for other short period of them, and this process keeps repeating. One way of detecting change point is to keep a representative sample set, periodically use stratified sampling on it to find the estimated population center, and compare it with the

population center computed by estimated cluster centers. If these two centers are far away from each other, i.e. the distance is greater than a preset threshold, a change point may occur and the whole proposed algorithm should be re-run to make adjustment for the new distribution. The period of this updating process should depend on the application. Also, there are many literatures introducing a lot of sophisticated models such as [Saatci, ICML 2010].

Due to time constraint, I apologize that I was not able to implement online EM for parametric distributions other than Gaussian and compare the proposed algorithm's performance on them. Also, because computing the determinant and the inverse matrix of a matrix of high dimension is computationally expensive and time consuming, I did not test the performance of online EM for Gaussian mixture on the DIM-sets dataset.

References

- Barbakh, W., & Fyfe, C. Online clustering algorithms. International Journal of Neural Systems.
- Roche, A. Em algorithm and variants: An informal tutorial. ArXiv.
- Liang, P., & Klein, D. Online EM for unsupervised models. The 2009 Annual Conference of the North American Chapter of the ACL.
- Givens, Geof H., and Jennifer A. Hoeting. Computational Statistics, Second Edition. Wiley, Print.
- Ross, Sheldon M. Simulation, Fifth Edition. China Machine Press, Print.
- Saatci, Y., Turner, R., & Rasmussen, C. E. Gaussian process change point models. Proceedings of the 27th International Conference on Machine Learning.
- Clustering datasets. Retrieved from http://cs.joensuu.fi/sipu/datasets/