International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:8, 2008

A Monte Carlo Method to Data Stream Analysis

Kittisak Kerdprasop, Nittaya Kerdprasop, and Pairote Sattayatham

***Abstract*—**Data stream analysis is the process of computing various summaries and derived values from large amounts of data which are continuously generated at a rapid rate. The nature of a stream does not allow a revisit on each data element. Furthermore, data processing must be fast to produce timely analysis results. These requirements impose constraints on the design of the algorithms to balance correctness against timely responses. Several techniques have been proposed over the past few years to address these challenges. These techniques can be categorized as either data-oriented or task-oriented. The data-oriented approach analyzes a subset of data or a smaller transformed representation, whereas task-oriented scheme solves the problem directly via approximation techniques. We propose a hybrid approach to tackle the data stream analysis problem. The data stream has been both statistically transformed to a smaller size and computationally approximated its characteristics. We adopt a Monte Carlo method in the approximation step. The data reduction has been performed horizontally and vertically through our EMR sampling method. The proposed method is analyzed by a series of experiments. We apply our algorithm on clustering and classification tasks to evaluate the utility of our approach.

***Keywords*—**Data Stream, Monte Carlo, Sampling, Density Estimation.

I. Introduction

DATA analysis is the process of computing various summaries and derived values from collected data. Data mining can be viewed as an intelligent data analysis aiming at extracting valuable knowledge from large amounts of information stored in data repositories [1], [3]. The techniques used in data mining have been adopted from the areas of machine learning and statistics, but scalable to deal with the problem of huge repositories of information. The recent advances in hardware and software have enabled the rapid generation of continuous stream of information such as customer click streams, telephone records, retail chain transactions, web page visits, and so on. Mining stream data that grow at an unlimited rate poses a new challenge to researchers and practitioners in the area of data mining [1], [9].

Data stream is defined as massive amounts of data continuously generated at a rapid rate, possibly time-varying and unpredictable [2], [9]. Major characteristics of data streams are the continuously online arrival of data elements, uncontrolled order of such elements upon arrival, variable sizes, and a one-time processing of an element before it is discarded or archived due to the massive size of data that far exceeds the storage capacity. The requirements of timely analysis and efficient memory usage constrain most data stream mining algorithms to sacrifice accuracy of the analysis results for the fast and feasible processing.

Development of approximation algorithms [5], [13] is a direct solution to the problem of data stream mining. However, the large volumes of data continuously arriving in a stream could eventually make the algorithms inefficient. A more practical solution is to apply a data reduction technique along with the approximation algorithms. Data summarization techniques, such as wavelet analysis [10] and histogram [2], have been proposed as synopsis data structures to provide a summary presentation of data. The issue of dynamic space allocation as the underlying data distribution changes over time is a fundamental problem of these approaches. Data stream analysis by choosing a subset of the incoming stream is another class of techniques for producing approximate results. Sampling is a statistical-based technique widely used to scale up the mining algorithms [7]. Nevertheless, in the context of data stream in which the data size is unknown, simply applying a sampling method cannot give reliable approximation.

We, therefore, propose a Monte Carlo method to draw representatives from data stream. Monte Carlo simulation is a widely used method to produce a good approximation to the true value or quantity. Our algorithm has been designed to produce data elements from which the approximate analysis is close to the exact one. We perform cluster and classification analyses on several data sets to verify the reliability of the method.

The paper is organized as follows. Section 2 presents the theoretical background of a general Monte Carlo method. Section 3 sketches the draft idea of density estimation from a sample. Our proposed method that is efficiently applicable to data stream analysis is explained in Section 4. Some of the experimental results from cluster and classification analyses over the reduced data stream are shown in Section 5. We conclude in Section 6 with a discussion for future work.

This work was supported by the Thailand Research Fund under Grant MRG4780170, the National Research Council, and Suranaree University of Technology for the sponsorship of Data Engineering and Knowledge Discovery Research Unit.

Kittisak Kerdprasop is with the School of Computer Engineering, and the director of Data Engineering and Knowledge Discovery Research Unit Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand (e-mail: kerdpras@ sut.ac.th).

Nittaya Kerdprasop is with the School of Computer Engineering, and the member of Data Engineering and Knowledge Discovery Research Unit Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand (e-mail: nittaya@ sut.ac.th).

Pairote Sattayatham is with the School of Mathematics, Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand (e-mail: pairote@ sut.ac.th).

II. Principles of Monte Carlo

Monte Carlo method is a class of stochastic algorithm for simulating the behavior of physical or mathematical systems [11], [14]. The term stochastic implies that the methods are non-deterministic in which they are based on the use of random numbers and probability statistics to investigate problem. To understand the method of Monte Carlo, it is useful to think of it as a general technique of numerical integration. Suppose we need to evaluate the *d*-dimensional integral of a function *f* over the unit interval

$$\Phi = \int_0^1 \int_0^1 \dots \int_0^1 f(x_1, x_2, \dots, x_n)\, dx_1\, dx_2 \dots dx_n = \int_{(0,1)^d} f(x)\, dx. \quad (1)$$

The integral is a non-random problem, but the Monte Carlo method represents the integral as an approximation problem by introducing a random vector *U* that is uniformly distributed between 0 and 1. Applying the function *f* to *U*, we obtain a random variable $f(U)$ with expectation

$$E[f(U)] = \int_{(0,1)^d} f(x)\phi(x)\, dx \quad (2)$$

where ϕ is the probability density function of *U*. Since the value of ϕ on the region of integration is 1, equation (2) becomes

$$E[f(U)] = \int_{(0,1)^d} f(x)\, dx \quad (3)$$

Equations (1) and (3) allow us to represent the integral Φ as a probabilistic expression as follow:

$$\Phi = E[f(U)] \quad (4)$$

To estimate Φ, we need a mechanism for drawing points $U_1, U_2, \dots, U_n$. Applying function *f* to each of these *n* random points yields *n* independent and identically distributed (*iid*) random variables $f(U_1), f(U_2), \dots, f(U_n)$, each with expectation Φ and standard deviation σ. Averaging the results produces the *Monte Carlo estimator*

$$\hat{\Phi} = \frac{1}{n}\sum_{i=1}^{n} f(U_i) \quad (5)$$

which is an unbiased estimator for Φ with the error $\hat{\Phi}$ - Φ approximately normally distributed with mean 0 and standard deviation $\sigma/\sqrt{n}$.

The form of the standard error $\sigma/\sqrt{n}$ is an important property of Monte Carlo methods. First, it tells us that if we increase the number of our samples by a factor of four, we will half the standard error. Second, standard error does not depend on the dimensionality of the integral. A Monte Carlo estimator based on *n* draws from the domain $[0,1]^d$ still have the form $\sigma/\sqrt{n}$ for all dimensions *d*. Most techniques of numerical integration such as the trapezoidal rule degrade in convergence rate with increasing dimensions.

We consider a Monte Carlo method to be useful in the domain of data stream analysis in which the number of data is overwhelming and the exact data distribution is unknown. The focus of our study is to generate samples from a stream data which is a prior step to data modeling and analysis. Once the samples have been successfully drawn, the characteristics of stream can be estimated. We concentrate on the sampling problem because it can provide a satisfactory estimation which will be proven through experimentations on cluster and classification analyses.

III. Sampling Method and Density Estimation

Basically the Monte Carlo method employs any technique of statistical sampling to approximate solutions to quantitative problems. With Monte Carlo method, a large system can be sampled in a number of random configurations, and that data can be used to describe the system as a whole. The efficiency of the method depends largely on the ability to draw samples effectively. For a particular domain of stream data, we consider the rejection sampling method. Rejection sampling, or acceptance-rejection sampling, is a sampling method first introduce by Von Neumann [16]. This method is used in cases where a target distribution, *f(x)*, is too complicate for us to sample from it directly.

Suppose we have a simpler distribution, *g(x)*, which we can evaluate and generate samples from, then the difficult sampling problem can be avoided by sampling from *g(x)* instead. By generating a uniform random variable *u* from the interval [0,1], we accept *x* if the condition $u \le f(x) / Cg(x)$ holds; otherwise reject the value of *x* and repeat the sampling step. Posing the restriction $Cg(x) \ge f(x)$ for some $C > 1$, we say that *Cg* envelopes *f*. The validation of this method is the envelope principle. When simulating the point (x, v) where $v = u*Cg(x)$, we produce a uniform simulation over the subgraph of $Cg(x)$. Accepting only points such that $u \le f(x) / Cg(x)$ then produces points (x, v) uniformly distributed over the subgraph of *f(x)* and thus, marginally, a simulation from *f(x)*.

Rejection sampling will work best if *g* is a good approximation to *f*. However, in a high-dimensional problem the value of *C* needs to be chosen very large to ensure the requirement $Cg(x) > f(x)$, for all *x*. The result is an enormous rejection rate.

The difficulty of applying rejection sampling method directly to the problem of data stream analysis is that we do not know beforehand where the modes of *f* are located or how high they are. In other words, we do not know the exact characteristics of the target density. We thus propose to apply the EM (Expectation-Maximization) technique [6] to approximate the density *f(x)*.

We consider multi-dimensional stream data as mixtures of Gaussian, or normal, probability density functions (pdf). Gaussian mixtures [8], [12] are combinations of Gaussian distributions written as

$$g(x) = \sum_{i=1}^{K} p_i f(x \mid \theta_i) \quad (6)$$

A random variable *x* denotes independent observation in *K* mixture components. The p_i's are the mixing proportions, $0 < p_i < 1$ for all $i = 1, \dots, K$, and $p_1 + \dots + p_K = 1$. The $f(x|\theta_i)$ denotes the density of a *d*-dimensional Gaussian distribution

with mean vector μ and covariance matrix Σ, that is θ = (μ,Σ), and the Gaussian pdf is given by [4], [15]

$$g_{(\mu,\Sigma)}(x) = \frac{1}{(2\pi)^{d/2}\sqrt{\det(\Sigma)}} \exp\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\} \quad (7)$$

By varying the number of Gaussians *K*, the mixing proportions p_i, and the parameter θ_i of each Gaussian density function, Gaussian mixtures can be used to describe any complex pdfs (Fig. 1).

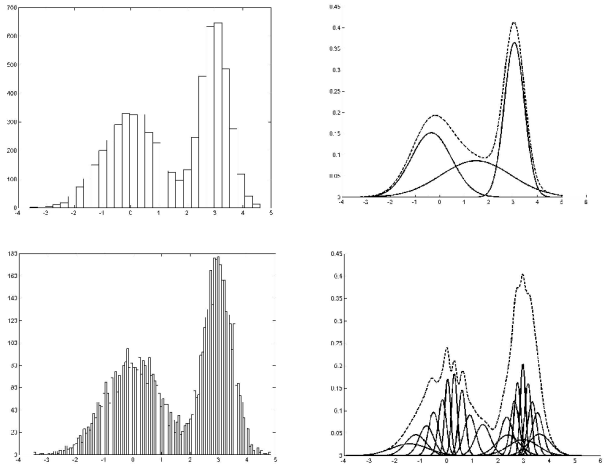


Fig. 1 one dimensional Gaussian mixture densities for *K* = 3 (first row) and *K* = 30 (second row). The left column shows the histogram of Gaussian density, the right column gives the corresponding Gaussian mixture pdf

In stream data a mixture density $p_i f(x|\theta_i)$ has been observed with unknown parameters θ_i and p_i. To find these parameters to optimally fit a mixture model for a given set of data, the EM algorithm [6], [12], [15] can be used. The EM algorithm is a broadly applicable approach to the iterative computation of maximum likelihood estimates. For a set of *iid* samples $X = \{ x_1, ..., x_N \}$ drawn from a data generation model $f_{(\mu,\Sigma)}(x_i)$, thus the resulting density for the samples is

$$\prod_{i=1}^{N} f_{(\mu,\Sigma)}(x_i) = L(\theta \mid x). \quad (8)$$

The likelihood function $L(\theta \mid x)$ is the likelihood of the parameters given the data. In the maximum likelihood problem, the goal is to find θ that maximizes *L*, that is $\arg\max_\theta L(\theta \mid X)$. In the Gaussian case, the computation of the exponential can be avoided by maximizing *log* ($L(\theta \mid x)$) instead of $L(\theta \mid x)$.

The EM algorithm is an approach to find the maximum of likelihood functions in incomplete data problems. Let X be observed data, Z be unobserved data, and Y = X∪Z be full data set. The probability distribution of Z depends on X and the unknown parameter θ. Given an initial parameter $\theta^{(0)}$, The EM algorithm produces a sequence $\{\theta^{(0)}, \theta^{(1)}, \theta^{(2)}, ... \}$ that converges to a stationary point of the likelihood function.

IV. EMR Sampling

In our particular case of data stream analysis, we assume that the observed data have a normal distribution. Given a specific number of models, the EM algorithm is applied to estimate the mean of each model. These mean values have been scaled up to produce an upper bound for the underlying partially observed target density. The idea of the proposed method is illustrated in Fig. 2. The target function is represented as a one-dimensional 3-Gaussian mixtures (the three solid lines at the bottom of Fig. 2) from which we want to draw samples. The density *E(x)* is estimated with the upper bound requirement that $E(x) > f(x)$ for all *x*. $E'(x)$ is the approximation (shown as a thick dash line in Fig. 2) of the unknown target density. A broad distance of *E* and *E'* (e.g., at *x* = 1) represents a rejecting area, whereas a narrow distance (e.g., at *x* = 6.5) is an acceptance one.

It should be noted that EM requires a pre-specified number of *K* components to be incorporated into the mixture models. According to our proposed method, a suitable number should be selected by a user. To cope with multi-dimensional problem, we propose to use a statistical method – principal component analysis (PCA) – to reduce the complicated problem to a simpler two-dimensional problem. That is, we take into account only the first and second major components of the data set. The two-dimensional data are used to train the EM algorithm to estimate parameters μ and Σ of the Gaussian mixture models. The estimated Gaussian pdf is a distribution E (as shown in Fig. 2). To sample from the estimated density we scale up this distribution to obtain an approximate E', which is a simpler distribution that we can evaluate and generate samples from. The outline of our EMR sampling algorithm is illustrated in Fig. 3. The subroutine *Density_Estimator* to approximate the density function has been shown in Fig. 4.

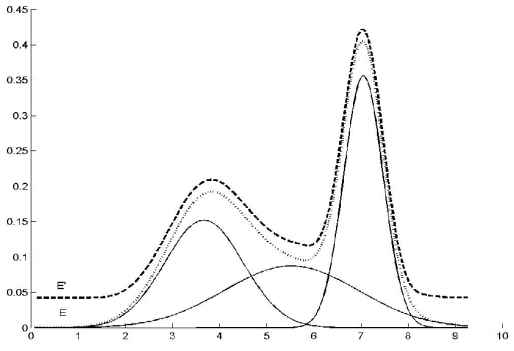


Fig. 2 EM-based rejection (EMR) sampling

From the estimated density E and the rough approximate E', we perform rejection sampling with the decision criteria $\{E(x)/(\sqrt{d} * E'(x))\} \geq u$, when *u* is a uniform variable distributed between 0 and 1, and *d* is a dimensionality of the data. The input from stream data has been taken one by one. The data item that satisfies the criteria will be included in the

sample until the specified sample size is completely filled up. Then the sampled data set is a representative of the whole stream. Any analysis methods can now be performed on this set.

Input: - a *d*-dimensional data set *D* with *N* points
- an integer *K* to specify the number of models, and
- a sample size *SS*

Output: - a sample set *S* drawn from the mixture models

// Data preprocessing steps //

1. If $d > 0$ then Apply PCA to obtain 1st and 2nd components
2. Transform *D* to a two-dimensional data set *X*

// Density estimation with EM and getting a rough pdf $E'(X)$ //

3. Set max_iteration = max{50, *d*K*}
4. ($E(X)$, $E'(X)$) = *Density_Estimator* (*X*, *K*, max_iteration)
5. Set count = 0
6. While count < *SS* *// Sampling steps //*
7. Sample *x* from $E(X)$
8. Generate *u* from U(0,1)
9. If $u \le E(x)/(\sqrt{d} * E'(x))$ then Accept *x*, add it to *S*, and increment count
10. Return *S*

Fig. 3 EMR sampling algorithm

Density_Estimator (*X*, *K*, *max_iteration*)

1. Initialize parameter $\theta = (\mu, \Sigma)$ for each of *K* Gaussian models by running K-means
2. Initialize the prior probabilities $P(m_k)$ of each model *m* to $1/K$, $k = 1,..., K$
3. Repeat
4. Compute the probability

$$P(m_k^{(i)} \mid x_n, \theta^{(i)}) = \frac{P(m_k^{(i)} \mid \theta^{(i)}) \bullet p(x_n \mid \mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_j P(m_j^{(i)} \mid \theta^{(i)}) \bullet p(x_n \mid \mu_j^{(i)}, \Sigma_j^{(i)})}$$

5. Update means μ_k, variances Σ_k, and priors *P*

$$\mu_k^{(i+1)} = \frac{\sum_{n=1}^{N} x_n P(m_k^{(i)} \mid x_n, \theta^{(i)})}{\sum_{n=1}^{N} P(m_k^{(i)} \mid x_n, \theta^{(i)})}$$

$$\sum_k^{(i+1)} = \frac{\sum_{n=1}^{N} P(m_k^{(i)} \mid x_n, \theta^{(i)}) (x_n - \mu_k^{(i+1)})(x_n - \mu_k^{(i+1)})^T}{\sum_{n=1}^{N} P(m_k^{(i)} \mid x_n, \theta^{(i)})}$$

$$P(m_k^{(i+1)} \mid \theta^{(i+1)}) = \frac{1}{N} \sum_{n=1}^{N} P(m_k^{(i)} \mid x_n, \theta^{(i)})$$

6. Until the *max_iteration* has been reached or the joint likelihood of all data with respect to all the models is greater than the lower boundary criterion $CL(\theta)$

$$L(\theta) \ge CL(\theta) = \sum_{k=1}^{K} \sum_{n=1}^{N} P(m_k \mid x_n, \theta) \log p(x_n \mid \theta)$$

7. Return $\theta_i = (\mu_k, \Sigma_k)$ for $k = 1, ..., K$, and a rough $\theta_i' = (\mu^r{}_k, \Sigma^r{}_k)$ from *r* iterations, $r < 10$

Fig. 4 Density-Estimator algorithm

V. Experimentations

A. Evaluation of Density Estimator

The objective of our initial experiments is to empirically evaluate the closeness of the estimated density to the real one. The closeness is determined by comparing the Euclidean distance of the estimated mean vector $\hat{\mu}$ to the original mean vector μ, and comparing the estimated covariance matrix $\hat{\Sigma}$ to the original covariance matrix Σ. We use a synthetic data generator to produce two-dimensional Gaussian mixtures. The number of mixture models, number of points in each model, original mean vector and covariance matrix are input parameters.

We vary the number of models from 2 to 20 with 50 to 1,000 data points in each model. To properly initialize the component means for the θ-parameter learning, we find the approximate mean points by running max{50, *d*K*} iterations of k-means algorithm [17]. Component elements and main diagonal covariance matrix elements are also initialized accordingly, and off-diagonal matrix elements are constrained to zero. Some of our experimental results on the accuracy of our density estimator compared with the simple uniform sampling are illustrated in Table 1. The EMR sampling results are compared against the uniform sampling which always assumes a single Gaussian model. The efficiency of the sampling methods is evaluated on the basis of the closeness of the estimated $\theta_i = (\mu_i, \Sigma_i)$ to the original means and covariance matrices of the generative models. The μ-differences and Σ-differences are averaged from *K* models. The experimental results confirm the applicability of the EMR approach toward the problem of θ-parameter approximation. The estimated means and variances are very close to the original parameter values.

TABLE I
Experimental Results of EMR Sampling from Various Mixing of Gaussian Models

Number of Mixture Models	EMR Sampling		Uniform Sampling	
	μ-difference	Σ-difference	μ-difference	Σ-difference
2	0.000113	0.000901	0.088772	0.144793
6	0.000425	0.001527	0.090213	0.231109
8	0.000961	0.001599	0.098055	0.271645
12	0.000987	0.001938	0.200137	0.430098
14	0.001017	0.001991	0.299873	0.456131
16	0.001025	0.002007	0.300159	0.513772
18	0.001328	0.002031	0.330011	0.720001
20	0.001414	0.002508	0.460101	0.935644

B. Cluster and Classification Analyses

To verify the utility of the proposed method on the real-world data we run the k-means clustering algorithm [17] on various sampled data from the UCI repository [http://www.ics.uci.edu/~mlearn/ MLRepository.html]. We test our algorithm on four data sets: Wisconsin diagnostic breast cancer (466 data points, 2 classes), diabetes (512 data points, 2 classes), DNA (2000 data points, 3 classes), and satellite image (4435 data points, 6 classes). In each data set, we assume that the class labels are correct clusters to be found by the k-means algorithm. By assuming prior knowledge about known clusters, we can evaluate the error rate of the cluster learning.

On evaluation the efficiency of the Monte Carlo approach we simulate a data stream by generating several samples for each data set. In our experiments we observe the performance of cluster learning on increasing samples varied from 1%, 5%, 10%, 15%, ... ,50%, and the complete data set. The experimental results are shown in Fig. 5. The clustering results reveal the efficiency of the proposed method that only around 10-25% sampling size is sufficient for the accurate learning of data clusters.

The classification task has been performed on the same experimental setting with the C4.5 algorithm [17]. the results are shown in Fig. 6.

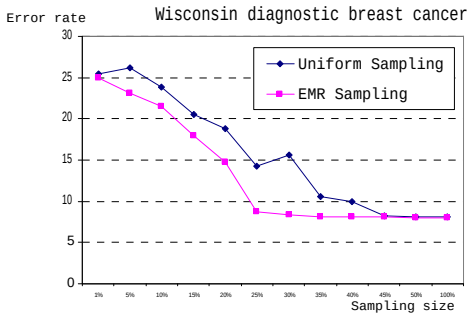
VI. Conclusion

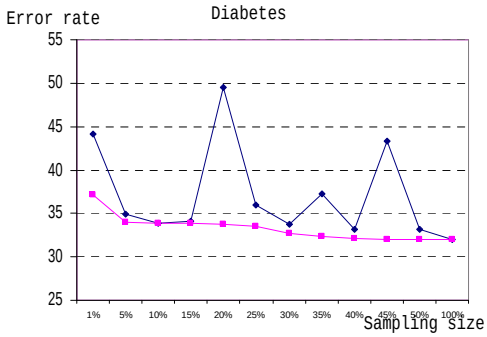
In this paper we propose a technique of Monte Carlo estimation to analyze major characteristics of data stream. At the sampling phase of the Monte Carlo method we propose the EMR sampling algorithm to efficiently draw representative samples from data containing mixture models. We propose to apply the expectation-maximization technique to estimate the means and variances of the mixture models. The algorithm *Density_Estimator* produces two density functions, E and E'. The distance of E and E' at each sampling point is a decision criteria for either sample acceptance or rejection. A narrow distance among the two estimated densities tends to the acceptance case if the distance ratio is greater than the generated uniform random variable from the interval [0, 1].

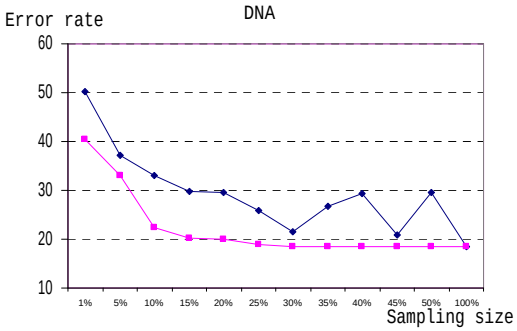
The experimental results verify the utility of the proposed *Density_Estimator* algorithm and the EMR sampling method. The clustering and classification experimentations on real-world data also confirm the efficiency of our method. We plan to further our study on skewed data in which the distributions are not uniformly distributed.

References

[1] C. Aggarwal, J. Han, J. Wang, and P. Yu, "A framework for clustering evolving data streams," in *Pro. Very Large Data Bases*, 2003.
[2] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Model and issues in data stream systems," in *Pro. ACM PODS*, 2002.
[3] M. Berthold and D.J. Hand, *Intelligent Data Analysis: An Introduction*. Springer-Verlag, 2003.
[4] J. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," Dept. Electrical Engineering and Computer Science, University of California Berkeley, Technical Report TR-97-021, 1998.
[5] G. Coremode and S. Muthukrishnan, "What's hot and what's not: Tracking most frequent items dynamically," in *Pro. ACM PODS*, 2003.
[6] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society B*, vol. 39, pp. 1-22, 1977.
[7] P. Domingos and G. Hulten, "A general method to scaling up machine learning algorithms and its application to clustering," in *Pro. ICML*, 2001.
[8] M. A. T. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, pp. 381–396, 2002.
[9] M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Mining data stream: A review," *SIGMOD Record*, vol. 34, pp. 18–26, 2005.
[10] A. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. Strauss, "One-pass wavelet decompositions of data streams," *IEEE Trans. Knowledge and Data Engineering*, vol. 15, pp. 541–554, 2003.
[11] D. Mackay, "Introduction to Monte Carlo," in *Learning in Graphical Models*, M. Jordan, Ed. MIT Press, 1996, pp. 175-204.
[12] J. M. Marin, K. Mengersen, and C. Robert, "Bayesian modelling and inference on mixtures of distributions," in *Handbook of Statistics*, vol. 25, Elsevier-Science, 2005.
[13] S. Muthukrishnan, "Data streams: Algorithms and applications," in *Proc. ACM-SIAM Symposium on Discrete Algorithm*, 2003.
[14] R. Neal, "Probabilistic inference using Markov chain Monte Carlo methods," Dept. Computer Science, University of Toronto, Technical Report CRG-TR93-1, 1993.
[15] B. Resch, "A tutorial for the course computational intelligence," Available: http://www.igi.tugraz.at/lehre/CI
[16] J. von Neumann, "Various techniques used in connection with random digits," *Applied Mathematics Series*, vol. 12, National Bureau of Standards, Washington, D.C., 1951.
[17] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, 2000.







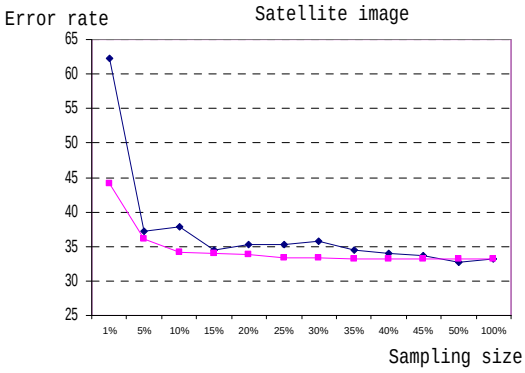
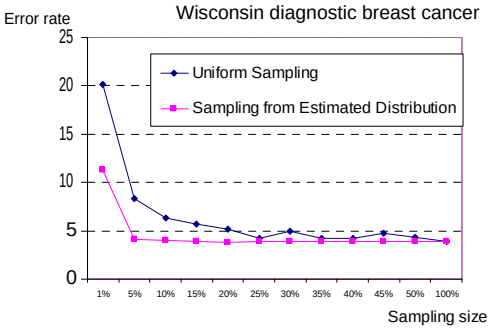
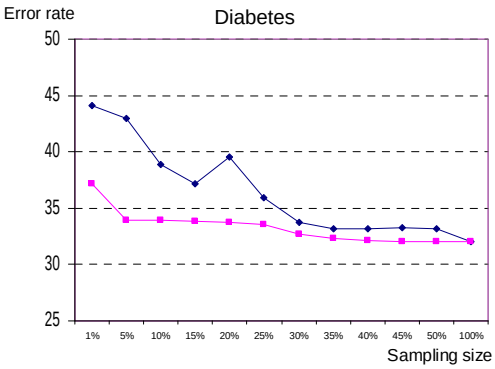
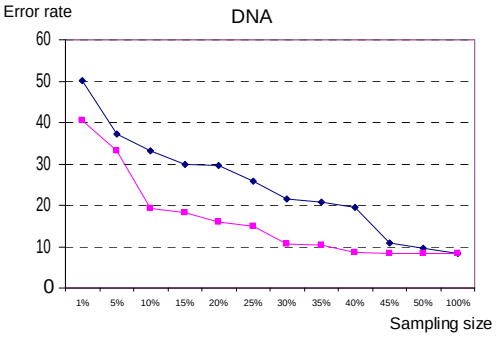


Fig. 5 Clustering results on four data sets







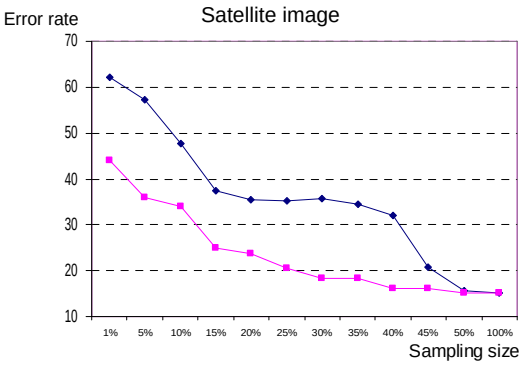


Fig. 6 Classification results on four data sets