Indonesian Journal of Electrical Engineering and Computer Science
Vol. 11, No. 3, September 2018, pp. 975~986
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v11.i3.pp975-986

An Improved Flexible Partial Histogram Bayes Learning Algorithm

Haider O. Lawend, Anuar M. Muad, Aini Hussain
Center for Integrated Systems Engineering and Advanced Technologies (INTEGRA),
Faculty of Engineering and Built Environment, Universiti Kebangsaan Malaysia,
43600 UKM Bangi, Selangor, Malaysia

Article Info

Article history:

Received Mar 31, 2018
Revised Apr 22, 2018
Accepted Jun 14, 2018

Keywords:

Classification
Histogram probability
Distribution
Machine learning
Naïve bayes
PHBayes

ABSTRACT

This paper presents a proposed supervised classification technique namely flexible partial histogram Bayes (fPHBayes) learning algorithm. The traditional classification algorithms like neural network, support vector machine, first nearest neighbor, nearest subclass classifier and Gaussian mixture model classifier are accurate but slow when dealing with large number of instances. In additional to that these algorithms might require to be retrain when the classes changes. On the other hand, algorithms like naïve Bayes and nearest class mean are fast but not accurate. It is difficult and challenging to have a classification algorithm that is fast and accurate when dealing with large number of instances. In our previous work, partial histogram Bayes (PHBayes) learning algorithm showed some advantages in the aspects of speed and accuracy in classification tasks. However, its accuracy declines when dealing with small number of instances or when the class feature distributes in wide area. In this work, the proposed fPHBayes solves these limitations. fPHBayes is able to work fast with good accuracy with large and small number of instances. fPHBayes uses a probability distribution derived from smoothing the observed histogram in order to represent the class. Then it performs the classification using the Bayesian rule. fPHBayes was analyzed and compared with PHBayes and other standard learning algorithms like first nearest neighbor, nearest subclass mean, nearest class mean, naive Bayes and Gaussian mixture model classifier. The experiments were performed using both real data and synthetic data considering different number of instances and different variances of Gaussians. The results showed that fPHBayes is more accurate and flexible to deal with different number of instances and different variances of Gaussians as compared to other classifiers.

Corresponding Author:

Haider O. Lawend,
Center for Integrated Systems Engineering and Advanced Technologies (INTEGRA),
Faculty of Engineering and Built Environment, Universiti Kebangsaan Malaysia,
43600 UKM Bangi, Selangor, Malaysia.
Email: haiderbnomar@yahoo.com

1. INTRODUCTION

Classification is a process of assigning an instance in the data to the class that it belongs to. Classification is included in many applications of pattern recognition, machine learning and data mining. The classification task becomes more challenging when dealing with large databases especially when the data keeps changing from time to time. Data stream is an example in which new information and classes are continuously added and updated. This causes the expansion of the size of the data and may reduce the accuracy of classifiers [1]. Robotic technology is another example of learning large data in which the data is continuously updating. Dealing with large and dynamic data may reduce the learning process of the robot to

perform certain tasks [2]. The challenge of learning big data is also addressed in the work of [3]. For example, in the Apache Hadoop system, processing and storage big data is challenging because of large number of configurations.

A network based classifier like artificial neural network (ANN) is not suitable for large database because it is slow in training phase. This is due to the complexity of its structure that contains the number of nodes, hidden layers and activation function [4]. ANN is also not suitable to deal with dynamic database as it requires to be retrained whenever a change occurs or new data is being added [5]. Instead of using the traditional nueral network, some researchers used extreme learning machine (ELM) which is a type of neural network with only one hidden layer. The work of [6] showed that ELM performed good in term of accuracy and speed in the field of pattern recognition. The work of [7] compared ELM with support vector machine (SVM) and the traditional neural network and showed that ELM is faster and more accurate than both algorithms. However ELM still requires to be retrained whenever a change occurs in the data which makes it not suitable to be used with large and dynamic data.

Distance based classifiers perform the classification by assigning the testing instance to the nearest training instance or prototype in the space. A simple classifier like first nearest neighbor (1stNN) represents a class density in the space using all training instances. 1stNN is very accurate and fast in training phase even when dealing with large number of instances, but it suffers from curse of dimensionality and very slow in testing phase, which makes it impractical [8-11]. Many researchers attempted improving the speed and accuracy of 1stNN [10-14]. Nearest subclass classifier (NSC) represents a class density in the space as multiple prototypes, where K-means clustering algorithm is typically used to find the prototypes [9]. NSC is faster than 1stNN in the testing phase and it is very accurate if it gets correct choice of the prototypes number. However, NSC with k-means is not suitable for large data. The work of [15] showed that k-means is not suitable to be used with large data because it is an iterative algorithm and can be slow with big data. Nearest class mean (NCM) represents a class density in the space as one prototype, which is the center of the class. NCM is a special case of NSC that use one prototype. NCM tends to generalize well for the base configuration [16]. It is very fast in training and testing phases but not accurate especially when the class density in the space spreads wider [9], [11], [17], Support vector machine (SVM) represents a class density in the space as support vectors. The number of support vectors increases when dealing with large data using two approaches: one versus one and one versus rest. The work of [3] showed that SVM can provide good accuracy. However, SVM is slow in training phase especially when dealing with large data [17-20]. It also requires to be retrained whenever a change in the data occurs.

Bayesian classifiers perform the classification by assigning the instance to the class with the highest posterior probability using Bayesian rule. Naïve Bayes (NB) represents a class feature probability density using only a single Gaussian function. NB is a simple, fast and accurate classifier when the class distribution is Gaussian. However, its accuracy decreases if the class distribution is not Gaussian [8]. Due to its simplicity, speed and accuracy in large databases, NB has become a popular classifier in many applications [21-24]. The accuracy of the NB classifier is comparable to the ANN [25]. However, it is lesser than that of the SVM [26-28]. Gaussian Mixture Model Classifier (GMMC) represents a class feature probability density using Gaussian mixture model function. An iterative algorithm like Expectation Maximization (EM) is normally used to estimate Gaussian mixture model (GMM) parameters [29-33]. GMMC is more accurate compared to NB but slower in training phase due to EM. Flexible Naïve Bayes (FNB) represents a class feature probability density using kernel function [22], [34], [35]. Gaussian kernel function is normally used in FNB [36]. FNB with Gaussian kernel function is similar to GMMC in which both of them represent a class feature probability density as mixture of Gaussians.

The main problem of classification is that it is difficult and challenging to have a classification algorithm that is fast and accurate when dealing with large number of instances. Reviewing the past researchs showed that the classification algorithms like ANN, SVM, 1stNN, NSC and GMMC are accurate but slow when dealing with large number of instances. In additional to that these algorithms might require to be retrained when the classes changes. On the other hand, algorithms like NB and NCM are fast but not accurate.

In our previous work, PHBayes which is a Bayesian algorithm represents a class feature probability density using histogram [37]. Compared with other algorithms, PHBayes is fast in both training and testing phases. It is also very accurate when dealing with large number of instances. In this paper, a new Flexible Partial Histogram Bayes learning algorithm (fPHBayes) is proposed which is an improvement of PHBayes. Compared with PHBayes in our previous work [37], the proposed fPHBayes is more accurate to deal with small and large number of instances, more flexible to the class probability distribution and requires fewer parameters to be considered. fPHBayes uses a probability distribution derived from smoothing the observed histogram and performs the classification using Bayesian rule.

Our contributions in this work is proposing a new supervised learning algorithm fPHBayes which is an accurate algorithm, able to work with small and large number of instances and flexible to the distribution of the class. fPHBayes is also compared with other standard and most related and recent learning algorithms considering real data from UCI database and synthetic data analysis.

The rest of this paper is organized as follows: PHBayes and the proposed algorithm fPHBayes are explained in more details in the methodology section. Results and analysis are provided in the results and discussion section and the conclusion is provided in the conclusion section.

2. RESEARCH METHOD

2.1. Partial Histogram Bayes Learning (PHBAYES)

PHBayes is a supervised probabilistic learning algorithm. It performs the classification by assigning a training instance, **X**, where $\mathbf{X} = (X_1,\ldots,X_n)$ represents a vector of a training instance with n features, to the class, C, with the highest posterior probability, $P(C|\mathbf{X})$, applying the Bayesian rule as in Equations (1) and (2). The training phase of PHBayes consists of three steps: the generation of the observed histogram, estimation of noise level in the histogram and estimation of histogram probability.

$$P(C|\mathbf{X}) = \frac{P(C)P(\mathbf{X}|C)}{P(\mathbf{X})} \quad (1)$$

$$P(C|\mathbf{X}) = P(C)\prod_{i=1}^{n} P(X_i|C) \quad (2)$$

The observed histogram, $\mathbf{b} = (b_1,\ldots,b_r)$, is generated directly from the instances of the class. The value of each bin in the histogram, b_k, represents the frequency of occurrence. The histogram resolution, r, is an important parameter in order to generate more accurate histogram compared to its own probability function.

Figure 1 shows observed histograms from small and large number of instances generated from mixture of Gaussian with two Gaussian components. In general, increasing the number of instances improves the accuracy of the observed histogram. The estimation of the noise level in the histogram is important procedure toward reducing the error in the histogram especially when dealing with small number of instances. The probability of chance, $p_{ch}(b_k)$, of each bin in the histogram is calculated using De-moivre Laplace theorem by treating the number of instances, s, as trials [38]. According to De-moivre Laplace theorem, when the number of instances is large, the probability of chance can be represented as Gaussian distribution with mean, $\mu_{ch} = sp$ and variance, $\sum_{ch} = spq$, where $p = 1/r$ and $q = 1\text{-}p$. The probability of chance is given in Equation (3). Figure 2 shows De-moivre Laplace theorem applied on an observed histogram taken from six sided dice. Figure 3 shows how to estimate the level of noise in PHBayes for three steps of iteration.

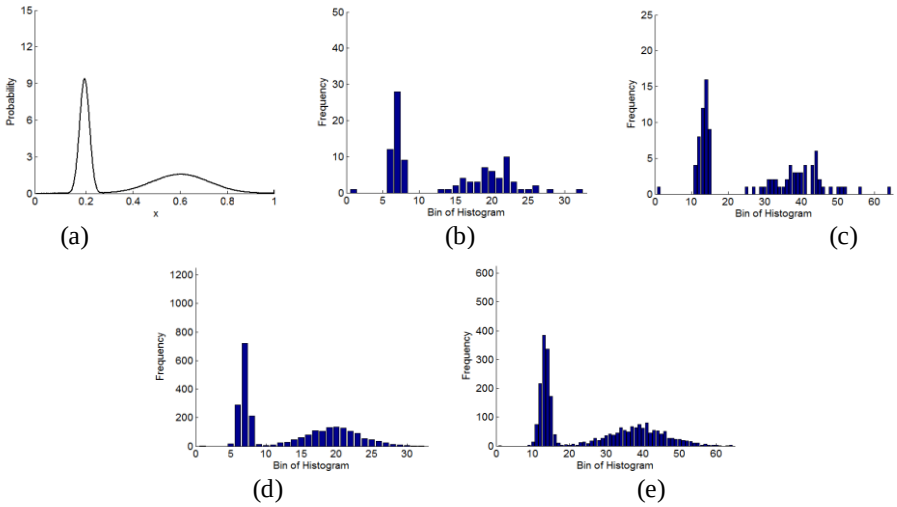


Figure 1. Observed histograms (a) is probability density function with two Gaussians. (b) and (c) are observed histograms from small number of instances using 32 and 64 bins. (d) and (e) are observed histograms from large number of instances using 32 and 64 bins.

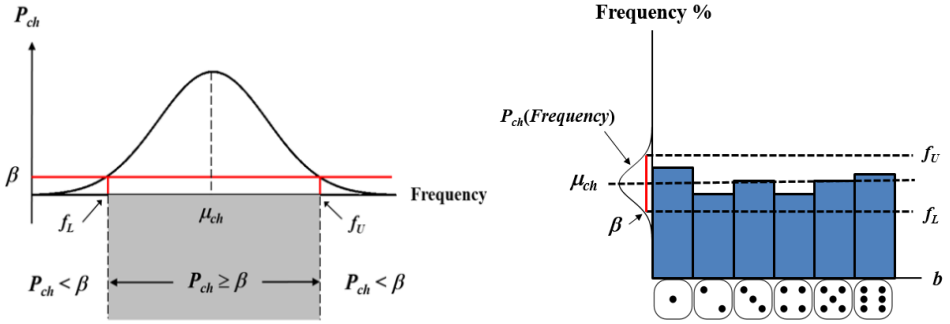


Figure 2. Calculating the probability of chance applied on the observed histogram of six sided dice. (a) The probability of chance. (b) The observed histogram of six sided dice.

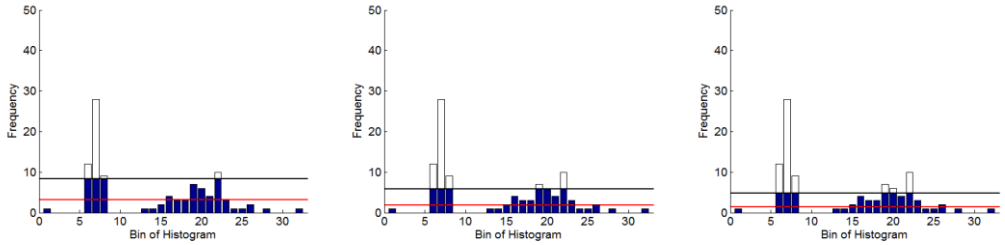


Figure 3. Histogram level of noise estimation in PHBayes (hard approach). (a), (b) and (c) are three steps of iteration where the lower horizontal red line represents the mean of the chance or noise, μ_{ch} and the upper horizontal black line represents the threshold f_U.

After determining the initial μ_{ch} and $\sum_{ch}$, an iteration procedure is performed to estimate the histogram level of noise. In each step of iteration, the instances of each bin, b_k with low probability of chance are removed or subtracted from the total number of instances, s, resulting in the remaining instances or the instances that occur by chance, s_{ch}, as given in Equation (4), where the upper frequency, f_U, is a threshold controlled by β parameter as in Equation (5). As a result of that, μ_{ch} and $\sum_{ch}$ are also updated in each step of iteration from the value of s_{ch}. This iterative procedure stops when it converges. After it converges, the histogram level of noise and its variance are the latest update of μ_{ch} and $\sum_{ch}$. In Figure 3, the upper horizontal dark line represents f_U and the lower horizontal red line represents μ_{ch} while f_L at zero. This approach is the hard thresholding approach. In each step of iteration, the parts of the histogram that are higher than the threshold f_U are removed.

$$p_{ch}(b_k) \sim N(b_k, \mu_{ch}, \Sigma_{ch}) = \frac{1}{\sqrt{2\pi\Sigma_{ch}}} e^{-\frac{(b_k-\mu_{ch})^2}{2\Sigma_{ch}}} \quad (3)$$

$$s_{ch} = s - \sum_{k=1}^{r} \left(b_k - f_U \mid p_{ch}(b_k) < \beta \,\&\, b_k > \mu_{ch} \right) \quad (4)$$

$$f_U = \mu_{ch} + \sqrt{-2\Sigma_{ch} \times \ln\left(\beta - 1/\sqrt{2\pi\Sigma_{ch}}\right)} \quad (5)$$

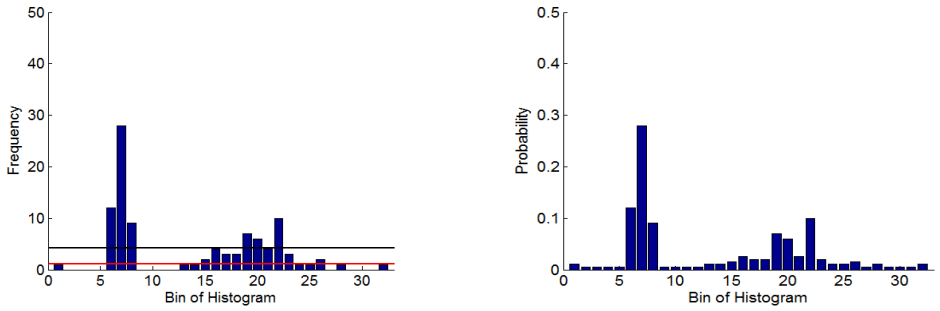
The histogram probability is estimated by smoothing the histogram bins b_k that have high probability of chance $p_{ch}(b_k)$ around the level of noise μ_{ch} then rescaling it by the number of instances s. There are two conditions in the histogram smoothing. First, if the probability of chance of a bin is lower than the threshold $p_{ch}(b_k)<\beta$, then no smoothing process occur. In this case, the bin of the histogram is considered to be reliable. Second, if the probability of chance of the bin is higher than the threshold, $p_{ch}(b_k)\geq\beta$, then the bin

of the histogram is not considered to be reliable and it is rescaled around the level of noise. The parameter α is used to control the scale of smoothing. The histogram probability, $p(b)$, is calculated by rescaling the smoothed histogram by the number of instances. The smoothing and probability estimation is calculated using Equation (6). Figure 4 shows how to smooth the histogram and calculate its probability in PHBayes. Here, only the bins between $f_U$and $f_L$are smoothed around μ_{ch}.

$$p(b_k) = \begin{cases} b_k / s & , p_{ch}(b_k) < \beta \\ \left(\alpha(b_k - \mu_{ch}) + \mu_{ch}\right)/s & , p_{ch}(b_k) \geq \beta \end{cases} \quad (6)$$

The testing phase of PHBayes is done by assigning the instance X to the class C with the highest posterior probability $P(C|\mathbf{X})$ using the Bayesian rule as in Equation (7), where k is the nearest bin to value of X_i.

$$P(C \mid \mathbf{X})_{\text{PHBayes}} = P(C)\prod_{i=1}^{n} p(b_k) \quad (7)$$



(a) (b)

Figure 4. Histogram smoothing (a) and probability estimation (b) in PHBayes. The lower horizontal red line represents the mean of the chance or noise $\mu_{ch}$and the upper horizontal black line represents the thresholdf_U.

The advantages of PHBayes are that it is fast both in training and testing phases. It is accurate especially when the number of instances increase as the error in the probability estimation reduces. It has low memory requirement since it only save bins of the histogram and probability for each class. It is also flexible to the shape and distribution of class by using histogram probability and is not required to be retrained although there are changes in the data and classes.

The limitations of PHBayes is that its performance is highly dependent on the values of the parameters α, β and r. Using different values of these parameters may lead to different classification results. For example, using small r is good to estimate the histogram probability with small number of instances while using large r is better with large number of instances. PHBayes is also not very accurate when dealing with small number of instances or if the distribution of the class spread wider. This is because in these cases, estimation of the histogram probability becomes difficult. These limitations reduce the estimation accuracy of the histogram probability.

2.2. Flexible Partial Histogram Bayes Learning (fPHBayes)

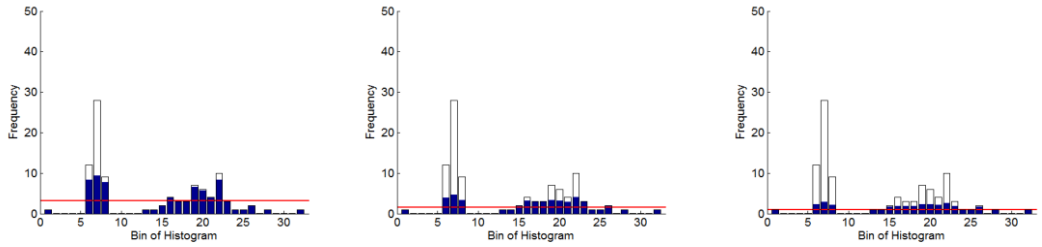
In this paper, a supervised flexible partial histogram Bayes learning algorithm (fPHBayes) is proposed to overcome some of the limitations of the original PHBayes. Compared with PHBayes, fPHBayes requires fewer parameters to consider and its performance does not depend on the parameters α and β. It is also suitable to work with large and small number of instances using higher histogram resolution. This property helps to increase the accuracy of fPHBayes.

Similar to PHBayes, the training phase of fPHBayes also consists of the three steps: the generation of the observed histogram, estimation of the level of noise in the histogram and estimation of the histogram probability. The procedure of generating the observed histogram in fPHBayes is similar to that in PHBayes with the exception that fPHBayes uses higher histogram resolution.

The procedure of estimating the level of noise in the histogram in fPHBayes is improved compare with PHBayes, where a soft approach is used instead of the hard one. The mean, μ_{ch} and variance, $\sum_{ch}$ are calculated from the probability of chance in each step of iteration and the remaining instances, $s_{ch}$is calculated using Equation (8). Compared with the calculation of the s_{ch} in the PHBayes that uses hard threshold f_U which is controlled by the parameter β, the calculation of the s_{ch}in the fPHBayes as in Equation (8) uses soft approach which does not have $f_U$or f_L and not controlled by the parameter β. In another word, to find the remaining instances s_{ch}, the reliable and non-noisy parts of all $b_k$are subtracted from s. The non-noisy part for each b_k, is the multiplication of the inverse probability of chance for the bin (1 - $p_{ch}(b_k)$) by the difference between $b_k$and μ_{ch}. This is only applicable when $b_k>\mu_{ch}$. As a result of that, in each step of iteration, $\mu_{ch}$and $\sum_{ch}$are updated based on the value of s_{ch}. This procedure stops when it converges. The histogram level of noise and its variance are the latest update of $\mu_{ch}$and $\sum_{ch}$.

$$s_{ch} = s - \sum_{k=1}^{r}\left[\left(1-p_{ch}\left(b_k\right)\right)\times\left(b_k-\mu_{ch}\right)\middle| b_k > \mu_{ch}\right] \tag{8}$$

Figure 5 shows how to estimate the level of noise in fPHBayes for three steps of iteration. The soft approach in fPHBayes affects all the parts in the histogram that are higher than the level of chance or noise μ_{ch}. The soft approach does not have fixed threshold like $f_U$as in the hard approach. Instead of that, each bin in the histogram that has a value higher than the level of chance is effected differently based on the probability of chance of the bin $p_{ch}(b_k)$. For example in the first step of iteration, the bin number 7, b_7 with the lowest probability of chance is the most effected bin by the soft approach. On the other hand, the bin b_{21} with high probability of chance is mostly not affected by the soft approach in the first step of iteration. The second and the third steps of iteration also show that when the level of chance μ_{ch}reduces after removing the non-noisy parts of the histogram, more parts of the histogram are affected because their probabilities of chance reduce. This is because the soft approach affects each bin based on its probability of chance. As a result of that the parameter β is not required.



(a) (b) (c)

Figure 5. Histogram level of noise estimation in fPHBayes (soft approach). (a), (b) and (c) are three steps of iteration where the horizontal red line represents the mean of the chance μ_{ch}.

The histogram probability estimation in fPHBayes is also improved compared with PHBayes. In fPHBayse, each bin in the observed histogram, $b_k$is smoothed by redistributing the part that is higher than the level of chance or noise (b_k-μ_{ch}) among the neighboring bins b_j, where $1 \le j \le r$. The redistribution of the bin $b_k$among the neighboring bin b_j, $G(b_j, b_k)$ is following Gaussian distribution $N(j, \mu_G, \sum_G(b_k))$ scaled by (b_k - μ_{ch})as in the Equation (9). The mean of the smoothing or redistribution, μ_G, is the histogram bin number, k, and the variance of the redistribution, $\sum_G$, depends on the probability of chance of bin $p_{ch}(b_k)$ and can be calculated using Equation (10). Therefore, the smoothed value of the bin, $\theta(b_k)$ is the summation of the redistribution of all neighboring bins $G(b_j, b_k)$plus the level of noise μ_{ch}as in the Equation (11). Finally and in order to get 100% probability distribution, the histogram probability, $p(b)$, is estimated by rescaling the smoothed histogram, $\theta(b)$, by the number of instances s as in Equation (12). This means the higher the probability of chance the bin has, the larger the variance of smoothing it has. Here, the parameter α is not required as each bin has a different variance of smoothing based on its $p_{ch}(b_k)$. This makes sure that only the noisy part of the histogram is smoothed and based on the amount of noise it has. This allows fPHBayes to work with small and large number of instances using larger histogram resolution compared with PHBayes. Figure 6 shows how to smooth the histogram and calculate its probability in fPHBayes. Here as an example

in Figure 6, the highlighted bins b_7 and b_{21} have different variance of smoothing represented by black lines. Since $b_7$has lower probability of chance than b_{21}, its instances are redistributed in smaller area compared with b_{21}. Finally, the probability is estimated by redistributing all bins in the histogram.

$$G(b_j, b_k) = N(j, \mu_G, \Sigma_G(b_k)) \times (b_k - \mu_{ch}) \tag{9}$$

$$\Sigma_G(b_k) = \frac{4pqr^2}{s(1 - p_{ch}(b_k))} \tag{10}$$

$$\theta(b_k) = \mu_{ch} + \sum_{j=1}^{r} G(b_j, b_k) \tag{11}$$

$$p(b_k) = \theta(b_k)/s \tag{12}$$

Similar to PHBayes, the testing phase in fPHBayes is done by assigning the instance X to the class C with highest posterior probability $P(C|\mathbf{X})$ using the Bayesian rule as in Equation (7). Figure 7 shows the histogram probability estimation of both PHBayes and fPHBayes using small and large number of instances. It is clear that both PHBayes and fPHBayes estimate the histogram probability better when the number of instances is large with respect to the histogram resolution. However when the number of instances is small, the estimation of fPHBayes is better than PHBayes. This is because the smoothing part in fPHBayes is improved. It tends to redistribute only bins with high probability of chance among other bins. This improvement makes sure that smoothing process only applies on the part that is not reliable in the histogram. This allows fPHBayes to work with large and small number of instances and with large and small variances using large histogram resolution.

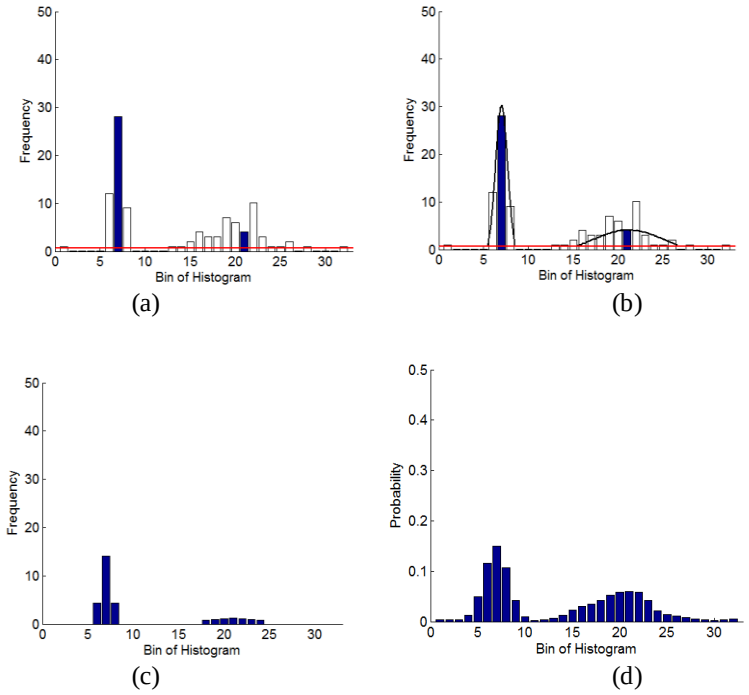


Figure 6. Histogram smoothing and probability estimation in fPHBayes. (a), (b) and (c) are the steps to smooth the histogram bins $b_7$andb_{21}. (d) The probability estimation by smoothing all bins

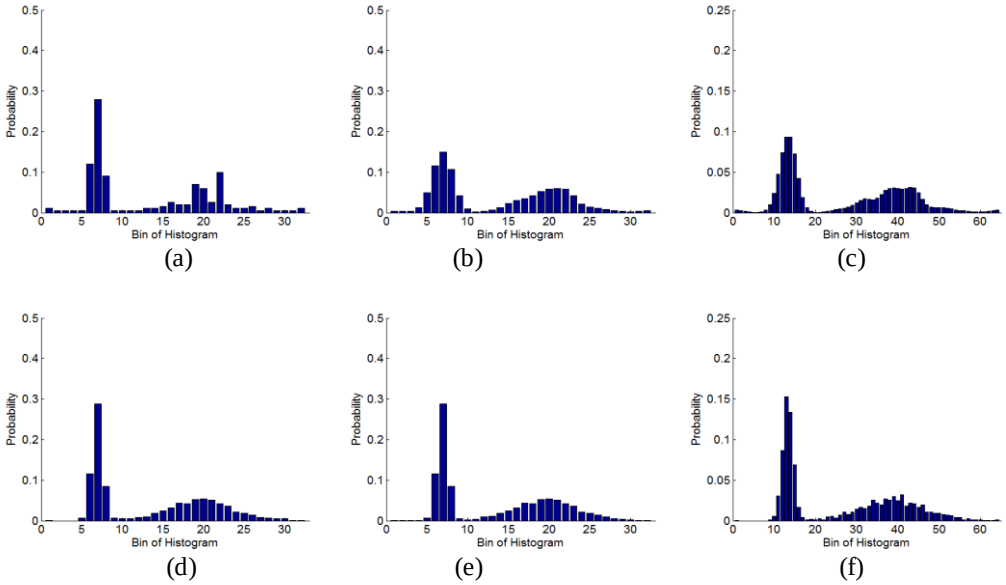


Figure 7. Probability estimation of PHBayes and fPHBayes. (a), (b) and (c) represent the probability estimation of PHBayes(32), fPHBayes(32) and fPHBayes(64) with small number of instances. (d), (e) and (f) represent the probability estimation of PHBayes(32), fPHBayes(32), fPHBayes(64) with large number of instances

3. RESULTS AND DISCUSSION

3.1. Results of Synthetic Data

In synthetic data experiments, two factors that affect the accuracy of the classifiers were considered. These factors were the number of training instances and variance of Gaussians. These factors were selected for these experiments to show the improvement in the proposed fPHBayes compared with PHBayes. Different classifiers were considered for the experiments. They were 1stNN, NCM, NSC with 5 components, GMMC with 5 components, PHbayes with r=16, PHBayes(16), PHBayes with r=32, PHBayes(32), PHBayes with r=48, PHBayes(48) and the proposed algorithm with r=64, fPHBayes(64). The values for α and β of the PHBayes were set to α=0.5 and β=0.005 [37]. The reason of using 64 bins of histogram in fPHBayes is to allow it to cover all ranges of histogram lower than 64 bins. This means fPHBayes with 64 bins of histogram is flexible to use any histogram lower than 64 bins. These different classifiers were selected for the test because they are standards and recent algorithms and used in many reseearchs. For example NSC was used in the works of [11, 17], GMMC was used in the work of [32, 33], and our previous PHBayes was proposed [37]. It is also important to mention that NCM is a special case of NSC in which a single prototype is used to represent a class density and NB is a special case of GMMC in which a single Gaussian component is used. The generation of the synthetic data was based on the representation of probability density functions of features of the classes as mixture of Gaussians. For all the classes, instances were derived from the probability density function and the number of training instance was the same as number of testing instances. The default value of the number of training instances per class was 300 and for the variance of Gaussians was around 0.1. Ten classes were used in each experiment. Each class had 16 features (8 independents and 8 dependents). Figure 8 shows the results of synthetic data experiments.

Figure 8a shows an analysis on the impact of the number of training instance as one of the factors that affects the classification accuracy. Number of training instance was varied from 60 to 540. The values for other factors were fixed at their respective default values. In general, the accuracy of all the classifiers increased with the addition of number of instance. However, the performance rate of the classifiers was at the different rate. The accuracy of the NB and NCM was less affected with the addition of the number of instance. PHBayes produced significant improvement as the number of instance increased. Large number of instance helped to improve the estimation of the histogram in the PHBayes. fPHBayes was very accurate when the number of instances was small as well as when the number of instances is large compared with PHBayes.

Figure 8b shows the results of variance of Gaussian that ranges from 0.02 to 0.18. Other factors were set to their default values. The increment of the variance typically reduced the classification accuracy of all the techniques. Compared to the other classifiers, the declining rate of the accuracy of the PHBayes was

more apparent because the class probability density was spread wider. fPHBayes was less effected by this factor than PHBayes.

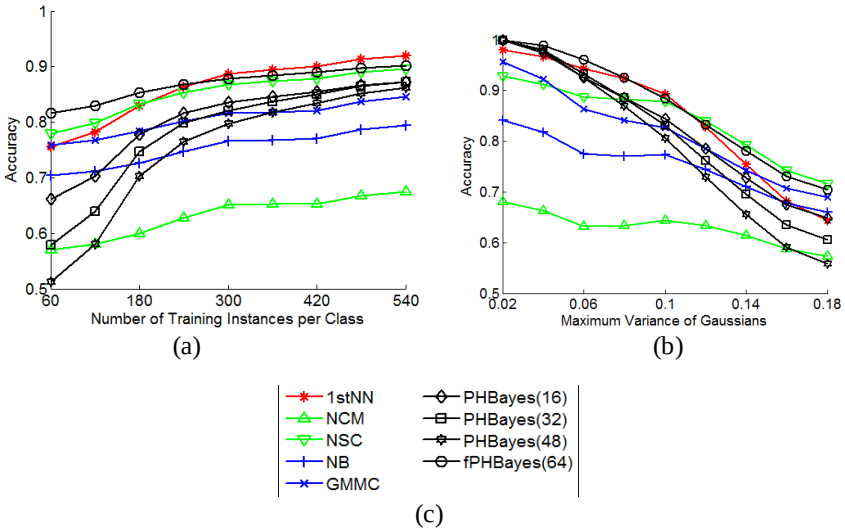


Figure 8. Accuracies results (a) Results from varying the number of training instances (b) Results from varying the variances of Gaussians (c) Legends

3.2. Resuts of Real Data

In real data experiments, twenty different databases from UCI machine learning database (http://www.ics.uci.edu/~mlearn/MLRepository.html) containing more than 50,000 instances from different classes were used. The database is presented in Table 1. Each database contains different classes, features, and instances. In order to standardize the analysis for all the classifiers, the features were scaled equally and nominal feature were ignored. About half of the instances were randomly selected as training instances and the other half were used for testing. fPHBayes was compared with other classifiers, 1stNN, NSC, NCM, NB, GMMC and PHBayes.

Table 2 presents the accuracy results of the classifiers on the databases. In general, fPHBayes produced high accurate results in 12 of the 20 databases (60%) as compared to other classifiers. fPHBayes was the most accurate classifier with average accuracy 0.7888. Compared with PHBayes, fPHBayes is the winner for 16 out of 20 of the databases.

The average computational time using the 20 databases is shown in Table 3. The time of fPHBayes was comparable to the fastest algorithm PHBayes in the testing phase, but slower in the training phase.

Table 1. Twenty UCI databases used in the analyses

No.	Database name	No. of instance	No. of features	No. of classes
1	Abalone	4177	8	3
2	Balance-scale	625	4	3
3	Blood Transfusion Service Center	748	4	2
4	Car evaluation	1728	6	4
5	Connectionist Bench (Sonar,	208	60	2
6	Contraceptive Method Choice	1473	9	3
7	Glass identification	214	9	6
8	Haberman's survival	306	3	2
9	Hayes-Roth	132	5	3
10	Hepatitis	155	19	2
11	Ionosphere	351	34	2
12	Iris	150	4	3
13	MAGIC Gamma Telescope	19020	10	2
14	Mammographic Mass	961	5	2
15	Nursery	12960	8	5
16	Pima Indians Diabetes	768	8	2
17	Spambase	4601	57	2
18	Statlog (Image Segmentation)	2310	19	7
19	Wine	178	13	3
20	Wisc. Breast Cancer (Diagnostic)	699	10	2

Table 2. Accuracy results

No.	1st NN	NCM	NSC	NB	GMMC	PHBayes (16)	PHBayes (32)	PHBayes (48)	fPHBayes (64)
1	0.4756	0.3987	0.4158	0.4400	0.3574	0.4993	0.4938	0.4961	**0.5036**
2	0.7994	0.7340	0.6875	**0.8782**	0.8750	0.7785	0.7785	0.7785	0.8772
3	0.7064	0.6631	0.5992	0.7644	0.7468	0.7535	0.7591	0.7535	**0.7623**
4	**0.9060**	0.7187	0.7141	0.6858	0.7541	0.8010	0.8010	0.8010	0.8135
5	0.9346	0.9923	0.9481	0.9865	0.9865	0.9231	0.9635	0.9394	**0.9942**
6	0.4359	0.4359	0.4088	0.4734	0.3971	**0.5075**	0.5026	0.5001	0.5043
7	0.6168	0.5673	0.6037	0.5869	0.7449	0.7598	0.6402	0.6047	**0.8065**
8	0.7085	0.7438	0.5699	0.7471	0.7392	0.7444	0.7216	0.7157	**0.7608**
9	0.5485	0.5273	0.4970	0.6152	**0.6182**	0.5318	0.5242	0.5242	0.5576
10	0.6519	0.7013	0.6714	0.6818	0.7143	0.8026	0.7766	0.7390	**0.8117**
11	0.8097	0.7743	0.7034	0.7663	0.7926	0.8526	0.8589	**0.8606**	0.7817
12	**0.8867**	0.4813	0.8080	0.5213	0.8520	0.8093	0.7493	0.7960	0.6640
13	0.9935	0.9053	0.9171	0.9087	0.9687	0.9665	0.9927	0.9853	**0.9950**
14	0.7558	0.7913	0.7527	0.8127	0.8285	0.8358	0.8327	0.8335	**0.8340**
15	**0.8889**	0.8200	0.7869	0.7206	0.7744	0.8664	0.8664	0.8664	0.8717
16	0.6411	0.6602	0.6331	0.6839	0.6859	0.6828	0.6690	0.6667	**0.6909**
17	0.9203	0.8234	0.6483	0.7122	0.7956	0.9370	0.9384	0.9333	**0.9438**
18	0.4423	0.4472	0.4239	0.3548	0.6072	0.6416	0.7106	**0.7375**	0.6523
19	0.7101	0.7730	0.7416	0.8393	0.8146	0.7247	0.7494	0.7596	**0.9753**
20	0.9613	0.9676	0.8914	0.9610	0.9642	0.9736	0.9736	0.9736	**0.9759**
Total Average	0.7397	0.6963	0.6711	0.7070	0.7509	0.7696	0.7651	0.7632	**0.7888**

Table 3. Training and testing times on the 20 databases

Algorithm	Training Time (s)	Testing Time (s)
1stNN	---	354.3157
NCM	1.7284	0.2485
NSC	22.2335	1.1067
NB	3.8252	0.5948
GMMC	202.8376	3.5611
PHBayes(16)	**1.5672**	0.1299
PHBayes(32)	1.8040	**0.1277**
PHBayes(48)	1.9970	0.1308
fPHBayes(64)	4.8354	0.1315

3.3. Speed and Memory Analyses

To analysis the speed and memory requirement for fPHBayes and PHBayes, it is important to look at their structures. The training phase in both algorithms consists of three steps: the observed histogram building, histogram level of noise estimation and histogram probability estimation. Both algorithms build the observed histogram the same way, thus the required time of building the observed histogram is $sn\tau$ where τ is unit of time. Both soft approach in fPHBayes and hard approach in PHBayes require the same number of operands to estimate the histogram level of noise. As a result, the required time to estimate the histogram level of noise in both algorithms is $lrn\tau$ where l is the number of iterations. Smoothing process and estimating the histogram probability in fPHBayes is slower than PHBayes since it redistributes the noisy instances of the bins among neighboring bins. This makes fPHBayes takes longer time equal to $r^2n\tau$ compared with only $rn\tau$ in PHBayes. The total required times for fPHBayes and PHBayes are $(s+lr+r^2)n\tau$ and $(s+lr+r)n\tau$ respectively. In the testing phase, both algorithms assign a testing instance to the class with the highest posterior probability applying Bayesian rule. Because of both algorithms use histogram probability, their total testing time is only $n\tau$. The memory size of both algorithms is $2rnM$. This is in order to save the observed histogram rnM plus the histogram probability rnM, where M is unit of memory.

3.4. Discussion

Our contributions in this work is proposing a new supervised learning algorithm fPHBayes which is an accurate algorithm, able to work with small and large number of instances and flexible to the distribution of the class. fPHBayes is also compared with other standard and most related and recent learning algorithms considering real data from UCI database and synthetic data analysis.

fPHBayes demonstrated more accurate results than other tested algorithms in most of the experiments. It is able to work with small as well as large number of instances and flexible to the class distribution. fPHBayes is a Bayesian algorithm which does not fall into curse of dimensionality. It is also important to mention than by using histogram probability, it is easy to handle the changes in the data. This is

because it is easy to update the histogram probability compared with retraining like in SVM, ANN, GMMC and NSC.

Speed and memory analysis demonstrated that fPHBayes is fast in training and testing phases and requires small memory. Speed analysis showed that the iteration part of the training phase does not depend on the number of instances for histogram algorithms which make them very fast in training phase. It also showed that the speed of these algorithms does not depend on the number of instances in the testing phase. This makes them have the fastest speed in testing phase compared with other tested algorithms. Results showed that histogram algorithms are very fast in training phase and the fastest in testing phase compared with NCM, NSC, KNN, NB and GMMC. Memory analysis showed that the memory requirement of histogram algorithms does not depend on the number of instances but only on histogram resolution, which is very small.

Limitation of fPHBayes can be inherited from the limitation of Bayesian algorithms. All Bayesian algorithms like NB, GMMC, FNB, PHBayes and fPHBayes assume that all features are independents which may lead to lost in the accuracy if the features become more dependent.

In general, fPHBayes showed very good accuracy, remarkable speed and small memory requirement with the flexibility to handle changes. With these characteristics, it is expected to use this algorithm in the applications that require continuous learning like humanoid robots and expert systems. Future work will focus on the accuracy and how to improve it in the case of working with dependent features.

4. CONCLUSION

In this paper, fPHBayes is proposed to work fast and accurate with large and small data and to cover some limitations in our previous work PHBayes. Compared with other tested algorithms, fPHBayes is more accurate and more flexible to handle different number of instances with different variances of Gaussians. Compared with PHBayes, the level of noise estimation and the probability estimation in fPHBayes is improved. fPHBayes does not require the parameters α and β as it applies soft approach and redistributes the instances based on the probability of chance. These features allow fPHBayes to work with higher resolution and provide better accuracy than PHBayes. The speed and memory analysis showed that the speed and memory of fPHBayes do not depend on the number of instances. The experiments on synthetic and real data showed that fPHBayes provided the most accurate results compared with PHBayes, 1stNN, NSC, NCM, NB and GMMC.

ACKNOWLEDGEMENT

Funding by the UniversitiKebangsaan Malaysia under the research grant DIP-2015-012 is gratefully acknowledged.

REFERENCES

[1] Mohammad MM, Jing G, Latifur K, Jiawei H, Bhavani T. Classification and Novel Class Detection in Concept-Drifting Data streams under Time Constraints. *IEEE Transactions on Knowledge and Data Engineering*. 2011; 23(6): 859-874.
[2] Freedman ST, Adams, J. A. Filtering Data Based on Human-Inspired Forgetting. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*. 2011; 41(6): 1544-1555.
[3] Rahman A, Hossen J, Venkataseshaiah C, CK Ho, Geok TK, Sultana A, Jesmeen MZH, Hossain F. A Survey of Machine Learning Techniques for Self-tuning Hadoop Performance, *International Journal of Electrical and Computer Engineering (IJECE)*, 2018; 8(3): 1854-1862.
[4] Bogdan WM. Neural Network Architectures and Learning Algorithms. *IEEE Industrial Electronics Magazine*. 2009; 3(4): 56-63.
[5] French RM. Catastrophic Forgetting in Connectionist Networks. *Trends in Cognitive Sciences*.1999; 3(4): 128-135.
[6] Anam K, Al-Jumaily A. Optimized Kernel Extreme Learning Machine for Myoelectric Pattern Recognition, *International Journal of Electrical and Computer Engineering (IJECE)*, 2018; 8(1): 483-496.
[7] Khalid K, Mohamed A, Mohamed R, Shareef H. Performance Comparison of Artificial Intelligence Techniques for Non-intrusive Electrical Load Monitoring, *Bulletin of Electrical Engineering and Informatics*, 2018; 7(2).
[8] Chaudhuri P, Ghosh AK, Oja H. Classification Based on Hybridization of Parametric and Nonparametric Classifiers. *Transactions on Pattern Analysis and Machine Intelligence*. 2009; 31(7): 1153-1164.
[9] Veenman CJ, Reinders MJT. The Nearest Subclass Classifier: A Compromise Between the Nearest Mean and Nearest Neighbor Classifier. *Transactions on Pattern Analysis and Machine Intelligence*. 2005; 27(9):1417-1429.
[10] Viswanath P, Sarma TH. 2011. *An Improvement to k-Nearest Neighbor Classifier*. Recent Advances in Intelligent Computational Systems (RAICS). Trivandrum, India. 2011; 227-231.
[11] Yiguang L, Sam GS, Chunguang L, ZhishengY. k-NS: A classifier by the Distance to the Nearest Subspace. *IEEE Transactions on Neural Networks*. 2011; 22(8): 1256-1268.

[12] Hui W. Nearest Neighbors by Neighborhood Counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2006; 28(6): 942-953.
[13] Hui W. Neighborhood Counting Measure and Minimum Risk Metric. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2010; 32(4): 766-768.
[14] Yoonho H, Bohyung H,Hee-Kap A. *A Fast NearestNeighbor Search Algorithm by Nonlinear Embedding*. IEEE Conference on Computer Vision and Pattern Recognition. Providence, RI, USA. 2012; 3053-3060.
[15] Bathla G, Aggarwal H, Rani R. A Novel Approach for clustering Big Data based on Map Reduce, *International Journal of Electrical and Computer Engineering (IJECE)*, 2018; 8(3): 1711-1719.
[16] Moutafis P, Leng M, Kakadiaris I. An Overview and Empirical Comparison of Distance Metric Learning Methods, *Cybernetics IEEE Transactions*, 2017; 47(3): 612-625.
[17] Mensink T, Verbeek J, Perronnin F,Csurka G. Distance-Based Image Classification: Generalizing to New Classes at Near-Zero Cost. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2013; 35(11): 2624-2637.
[18] Mathur A, Foody GM. Multiclass and Binary SVM Classification: Implications for Trainingand Classification Users. *IEEE Geoscience and Remote Sensing Letters*. 2008; 5(2): 241-245.
[19] Chih-Wei H, Chih-Jen L. A Comparison of Methods for Multiclass Support Vector Machines. *IEEE Transactions on Neural Networks*. 2002; 13(2): 415-425.
[20] Ravikumar B, Thukaram D,Khincha HP.Comparison of Multiclass SVM Classification Methods to Use in A Supportive System for Distance Relay Coordination. *IEEE Transactions on Power Delivery*. 2010; 25(3): 1296-1305.
[21] Yu J, Bai M, Wang G, Shi X. Fault Diagnosis of Planetary Gearbox with Incomplete Information Using Assignment Reduction and Flexible Naive Bayesian Classifier", *Journal of Mechanical Science and Technology*, 2018; 32(1): 37-47.
[22] Xi-Zhao W, Yu-Lin H, Wang DD. Non-Naive Bayesian Classifiers for Classification Problems with Continuous Attributes. *IEEE Transactions on Cybernetics*. 2014; 44(1): 21-39.
[23] Hung-Ju H, Chun-Nan H. Bayesian Classification for Data from the Same Unknown Class. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*. 2002; 32(2): 137-145.
[24] Asr M, Ettefagh M, Hassannejad R, Razavi S. Diagnosis of Combined Faults in Rotary Machinery by Non-Naive Bayesian Approach, *Mechanical Systems and Signal Processing*, 2017; 85: 56-70.
[25] Soria D, Garibaldi JM, Biganzoli E, Ellis IO. *A Comparison of Three Different Methods for Classification of Breast Cancer Data*. International Conference on Machine Learning and Applications, San Diego, CA, USA. 2008; 619-624.
[26] Yan-Shi D, Ke-Song H. *A Comparison of Several Ensemble Methods for Text Categorization*. IEEE International Conference on Services Computing (SCC 2004). Shanghai, China, China. 2004; 419-422.
[27] Dilrukshi I, Zoysa KD. *Twitter News Classification: Theoretical and Practical Comparison of SVM AgainstNaive Bayes Algorithms*. International Conference on Advances in ICT for Emerging Regions (ICTer), Colombo, Sri Lanka. 2013; 278-278.
[28] Shameem F,Nisar H. *Comparison of Classification Techniques-SVM and Naives Bayes to Predict the Arboviral Disease-Dengue*. IEEE International Conference on Bioinformatics and Biomedicine, Atlanta, GA, USA. 2011; 538-539.
[29] Yonghong H, Englehart KB, Hudgins B, Chan ADC. A Gaussian Mixture Model Based Classification Scheme for Myoelectric Control of Powered Upper Limb Prostheses. *IEEE Transactions on Biomedical Engineering*. 2005; 52(11): 1801-1811.
[30] Karthikeyan G, Rajendra AU, Chua Kuang C, Choo Min L, Thomas AK. One-Class Classification of Mammograms Using Trace Transform Functionals. *IEEE Transactions on Instrumentation and Measurement*. 2014; 63(2): 304-311.
[31] Wei L, Saurabh P, James FE, Mann BL. Locality-Preserving Dimensionality Reduction and Classification for Hyperspectral Image Analysis. *IEEE Transactions on Geoscience and Remote Sensing*. 2012; 50(4): 1185-1198.
[32] Wei L, Saurabh P, James FE. Hyperspectral Image Classification using Gaussian Mixture Models and Markov Random Fields. *IEEE Geoscience and Remote Sensing Letters*. 2014; 11(1): 153-157.
[33] Saurabh P, Minshan C, Wei L, James FE. Segmented Mixture-of-Gaussian Classification for Hyperspectral Image Analysis. *IEEE Geoscience and Remote Sensing Letters*. 2014; 11(1): 138-142.
[34] John GH, Langley P. *Estimating Continuous Distributions in Bayesian Classifiers*. Conference of. Uncertain Artificial Intelligence;1995; 338–345.City???.
[35] Perez A, Larranaga P,Inza I. Bayesian Classifiers Based on Kernel Density Estimation: FlexibleClassifiers. *International Journal of Approximate Reasoning*. 2009; 50(2): 341-362.
[36] Liu JNK, Yu-Lin H, Xi-Zhao W, Yan-Xing H. A *Comparative Study AmongDifferent Kernel Functions in Flexible Naive Bayesian Classification*. International Conference on Machine Learning and Cybernetics, Guilin, China. 2011. 638-643.
[37] Lawend HO, Muad AM. *A Non Narametric Partial Histogram Bayes Learning Algorithm for Classification Applications*. Proceedings of IEEE International Conference on Control System, Computing and Engineering (ICCSCE). Penang. 2014; 35-39.
[38] Papoulis A. Probability, Random Variables, and Stochastic Processes. ThirdEdition: New York: McGraw-Hill. 1991.