An Adaptive Ensemble Classifier for Mining Concept Drifting Data Streams

Presented By

Dr. Dewan Md. Farid

Associate Professor

Department of Computer Science & Engineering

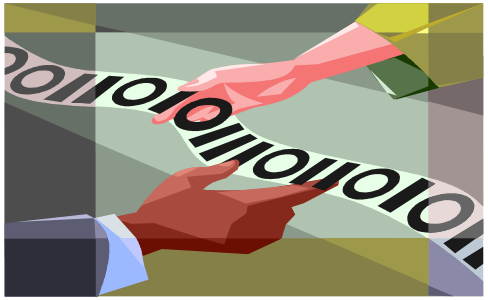
United International University, Bangladesh

Outline of The Presentation

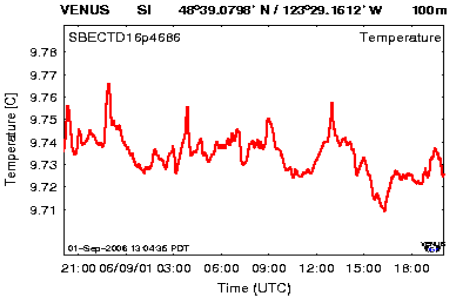
- Research challenges for mining data streams
- Novel class instances
- Adaptive ensemble classifier
- Experimental analysis
- Conclusions and future works

Data Streams Mining

It's a method of extracting knowledge and information from continuous data instances. Instances in data streams are generated with time passing by and cannot be controlled by any pre-defined order.



Continuous data flow



Sensor data

Network traffic

Call center records

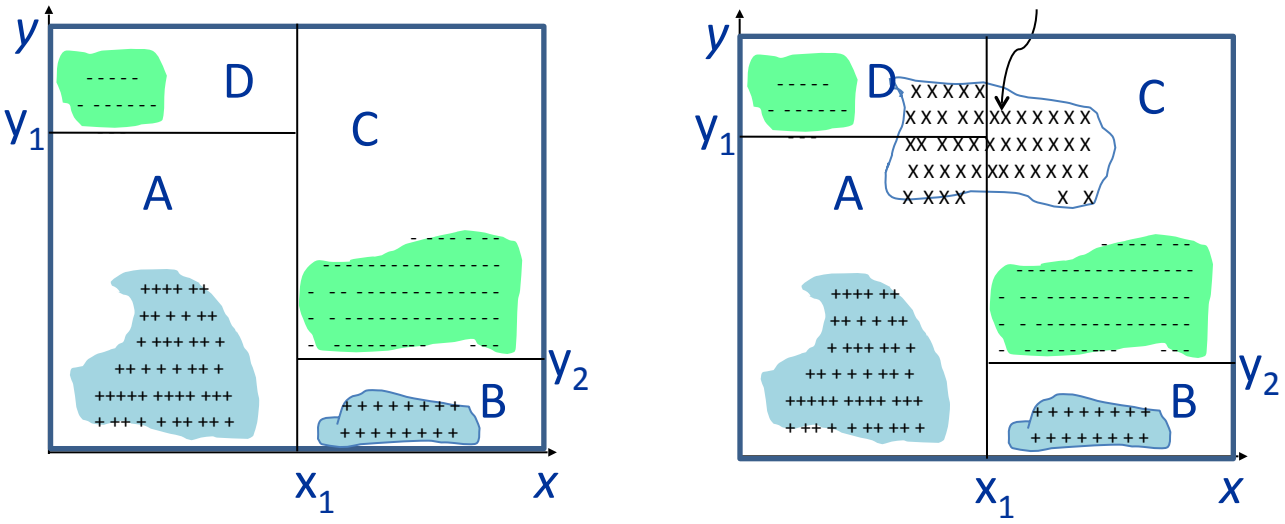
Research Challenges

- Existing mining classifiers need to be updated frequently to adapt to the changes in data streams.
- A data stream has very diverse characteristics compared to the traditional static data or database.
    A. Dynamic
    B. Infinite
    C. High dimensional
    D. High-speed
    E. Non-repetitive
- New instances with new class labels may appear at any time.
- Dynamic feature sets
- Noise in data streams

Novel Class Instances

- Most existing data mining techniques cannot detect and classify such novel class instances in data streaming environments, because they are trained on instances with a fixed number of class labels.
- The classification models become less accurate as time passes.
- The statistical properties of the target class, which the data mining classifiers are trying to classify, change over time in unforeseen ways.

Novel Class Instances (con.)



Classification rules:

R1. if ($x > x_1$ and $y < y_2$) or ($x < x_1$ and $y < y_1$) then class = +

R2. if ($x > x_1$ and $y > y2$) or ($x < x_1$ and $y > y_1$) then class = -

Existing classification models misclassify novel class instances

Data Streams Mining Models

- **Single model incremental classification**

It incrementally update a single classifier with new data to cope with the evolution of the data stream. Usually it's requires complex operations to modify the internal structure of the classifier. In some cases, this technique perform poorly.

- **Ensemble model based classification**

It's a combination or a set of classifiers, i.e. it combines a series of classifiers with the intention to create an improved composite model. It has higher classification accuracy rates than single model classification techniques.

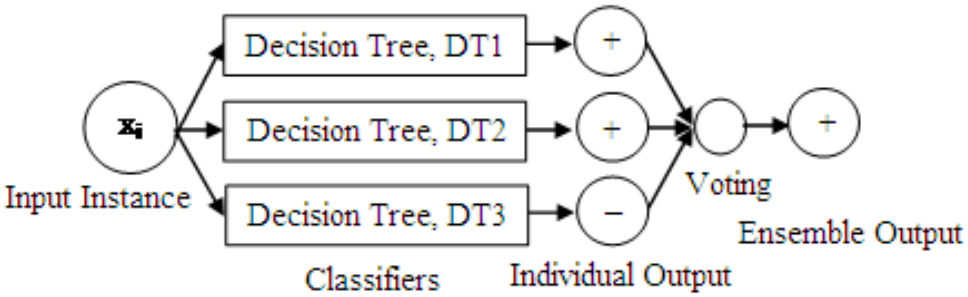


Fig. An example of Ensemble classifier.

Adaptive Ensemble Classifier

The proposed adaptive ensemble classifier addresses several challenges in data stream classifications:

1. Used traditional mining classifiers (Decision tree with Clustering).
2. Updates itself automatically (so that it represents the most recent concepts in data streams).
3. Handled infinite length problem (dividing a data stream into equal-sized sub-stream).
4. A new instance is classified by the majority of weighted voting among the classifiers in *M*.
5. The instances of each sub-stream are labeled by *M* and then a new classifier is trained with the most recent dataset.
6. The classifier with the smallest weight reflecting the minimum classification accuracy rate in *M* is chosen for replacement.

Adaptive Ensemble Classifier (con.)

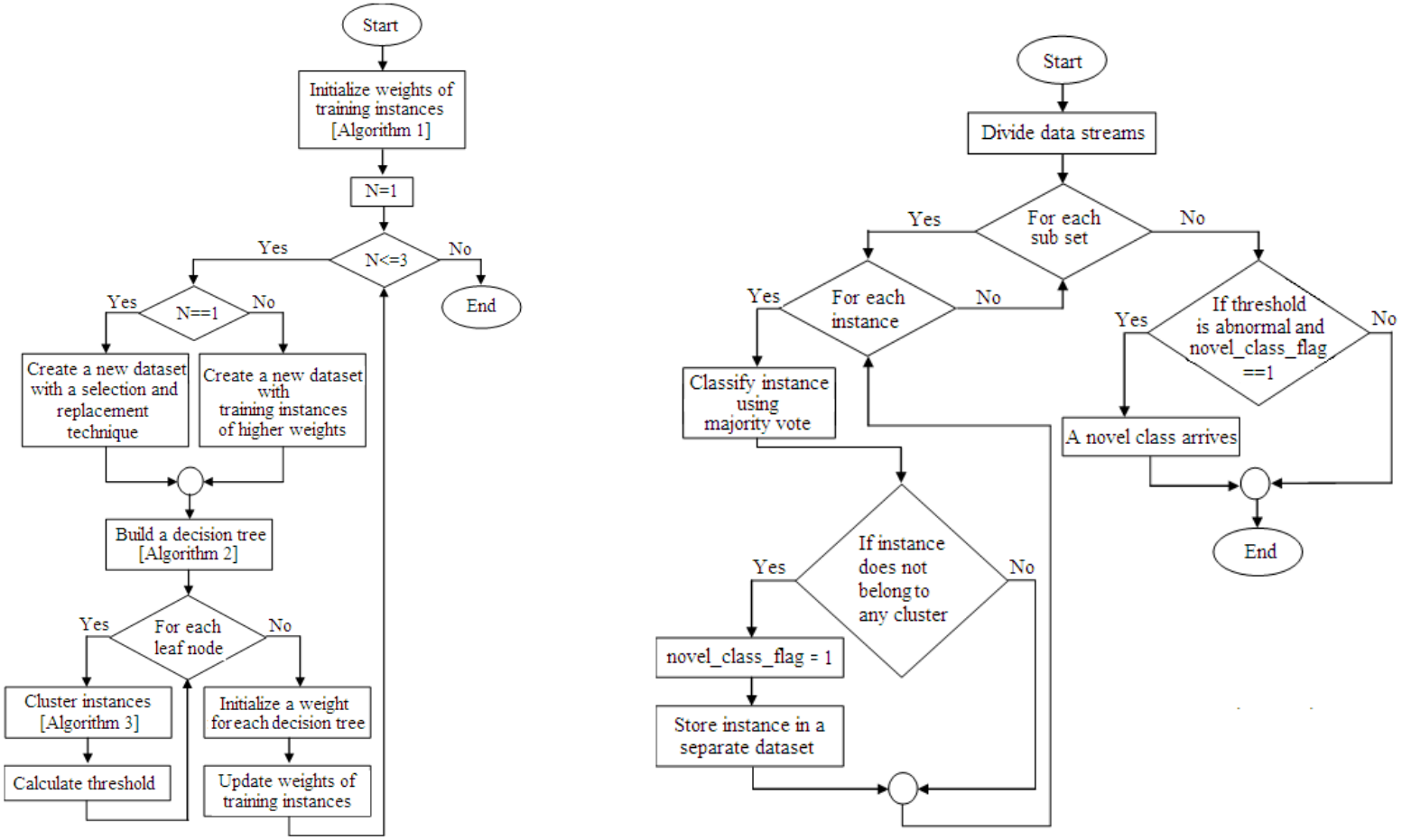


Fig. Flow charts of the proposed adaptive ensemble classifier.

Adaptive Ensemble Classifier (con.)

Algorithms:

A. Algorithm 1. Instance Weighting using NB Classifier
B. Algorithm 2. Decision Tree Learning
C. Algorithm 3. Similarity Based Clustering
D. Algorithm 4. The Adaptive Ensemble Classifier
E. Algorithm 5. Classification and Novel Class Detection

Algorithm 1. Instance Weighting using NB Classifier

Input: $D = \{x_1, x_2, \ldots, x_n\}$// Training data.
Output: $w_i \rightarrow x_i \in D$.// Weight for each instance.
Method:
1: **for** each class, $C_i \in D$, **do**
2: Find the prior probabilities, $P(C_i)$.
3: **end for**
4: **for** each attribute value, $A_{ij} \in D$, **do**
5: Find the class conditional probabilities, $P(A_{ij}|C_i)$.
6: **end for**
7: **for** each training instance, $x_i \in D$, **do**
8: Find the posterior probability, $P(C_i|x_i)$
9: Assign the weight, $w_i \rightarrow x_i \in D$, with Maximum Likelihood (ML) of posterior probability, $w_i = P_{ML}(C_i|x_i)$;
10: **end for**

Algorithm 2. Decision Tree Learning

Input: $D_i = \{x_1, x_2, \ldots, x_n\}$// Training data.
Output: T, Decision tree.
Method:
1: $T = \emptyset$;
2: Determine the best splitting attribute;
3: T = Create the root node and label it with the splitting attribute;
4: T = Add an arc to the root node for each split predicate and label it with a corresponding attribute value;
5: **for** each arc **do**
6: D_{ij} = Create sub dataset by applying splitting predicate to D_i;
7: **if** stopping point reached for this path,
8: T' = Create a leaf node and label it with an appropriate class;
9: **else**
10: T' = DTBuild (D_i);
11: **end if**
12: T = Add T' to arc;
13: **end for**

Algorithm 3. Similarity Based Clustering

Input: $D_i = \{x_1, x_2, \ldots, x_n\}$// A set of instances.
Output: A set of K clusters.
Method:
1: $K_1 = \{x_1\}$;
2: $K = \{K_1\}$;
3: $k = 1$;
4: **for** $i = 2$ to n **do**
5: find x_m in some cluster K_m in K so that $sim(x_i, x_m)$ is the maximum;
6: **if** $sim(x_i, x_m) \geqslant threshold\_value$ **then**
7: $K_m = K_m \cup x_i$
8: **else**
9: $k = k + 1$;
10: $K_k = \{x_i\}$;
11: **end if**
12: **end for**

Algorithm 4. The Adaptive Ensemble Classifier

Input: $D = \{x_1, x_2, \ldots, x_n\}$// Training dataset.
Output: M, An ensemble model.
Method:
1: **for** each instance, $x_i \in D$, **do**
2: Initialize the weight, $w_i \rightarrow x_i \in D$, using Algorithm 1.
3: **end for**
4: **for** 1 to N, where $N = 3$ **do**
5: **if** $N = 1$ **then**
6: Generate a new dataset, D_{new}, from D using a selection and replacement technique.
7: **else**
8: Generate a new dataset, D_{new}, from D with x_i of higher weights.
9: **end if**
10: Build a decision tree, T, from D_{new} using Algorithm 2.
11: **for** each leaf node in T **do**
12: Cluster the instances of D_{new} using Algorithm 3.
13: Calculate the threshold value based on the ratio of percentage of instances in this leaf node and instances in D_{new}.
14: **end for**
15: Initialize the weight, $W_i \rightarrow T$, based on its classification accuracy rate for the categorization of the instances, $x_i \in D$.
16: Update the weight, w_i, of each $x_i \in D$.
17: **end for**

Algorithm 5. Classification and Novel Class Detection

Input: Real data streams and the ensemble model, M.
Output: Labelling data streams and novel class detection
Method:
1: Divide the data stream into equal-sized sub-data streams.
2: **for** each sub-data stream **do**
3: **for** each instance
4: Classify the instance using each $T_i \in M$.
5: Return a weighted vote (which counts as one vote).
6: Assign a class label with majority of weighted votes.
7: **if** an instance does not belong to any existing clusters **then**
8: *novel_class_flag* = 1
9: Store this instance in a separate dataset.
10: **end if**
11: **end for**
12: **end for**
13: **if** threshold of $T_i \in M$ is abnormal and *novel_class_flag* == 1 **then**
14: A novel class arrives.
15: **end if**

Experimental Analysis

Used Symbols and Terms

Symbol	Term
N	Total instances in the data stream
N_c	Total novel class instances in the data stream
F_p	Total existing class instances misclassified as novel class
F_n	Total novel class instances misclassified as existing class
F_e	Total existing class instances misclassified
M_{new}	% of novel class instances misclassified as existing class
F_{new}	% of existing class instances falsely identified as novel class
ERR	Total misclassified error

The equations used to evaluate learning algorithms:

$$M_{new} = \frac{F_n * 100}{N_c}$$

$$F_{new} = \frac{F_p * 100}{N - N_c}$$

$$ERR = \frac{(F_p + F_n + F_e) * 100}{N}$$

Experimental Analysis (con.)

- We implement our algorithm in Java.
- The code for decision tree has been adapted from the Weka machine learning open source repository (http://www.cs.waikato.ac.nz/ml/weka).
- Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization.
- The experiments were run on an Intel Core 2 Duo Processor 2.0 GHz processor (2 MB Cache, 800 MHz FSB) with 1 GB of RAM.

Datasets from UCI Machine Learning Repository

Dataset	No of Attributes	Attribute Types	No of Instances	No of Class Attribute
NSL-KDD Dataset	41	Real & Nominal	25192	23
Large Soybean Database	35	Nominal	683	19
Image Segmentation	19	Real	2310	7

Experimental Analysis

Table 5
Performance comparison of classifiers in the presence of novel classes using the NSL-KDD dataset.

Instances	Existing classes	Novel classes	Classifier	Correctly classified instances (%)
Training: 13449 Testing: 9234	1	6	EM C4.5 k-NN	90.74 86.49 80.75
Training: 22683 Testing: 220	7	10	EM C4.5 k-NN	85.00 44.09 41.36
Training: 22903 Testing: 2289	19	4	EM C4.5 k-NN	79.03 52.55 51.90

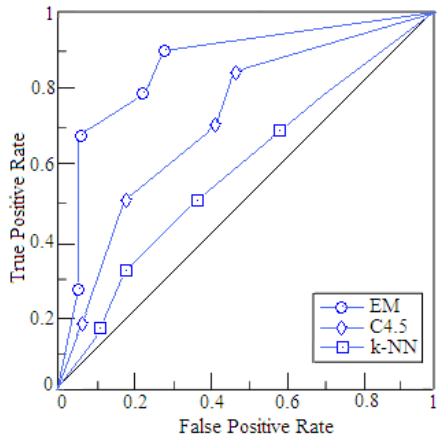


Fig. The ROC curves of classifiers using the entire NSL-KDD dataset.

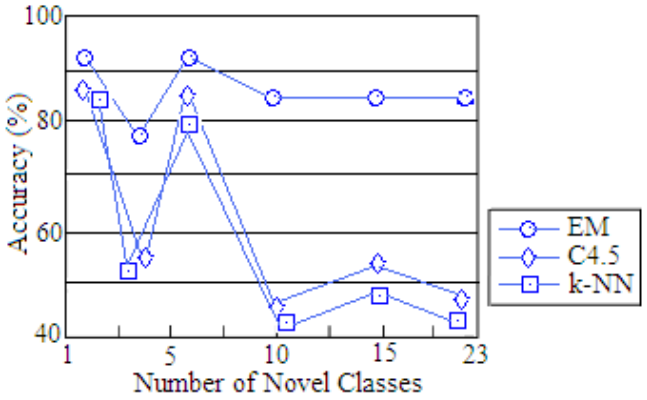


Fig. The comparison of accuracy rate of classifiers in concept drifting using the NSL-KDD dataset.

Experimental Analysis (con.)

Table 6
Performance comparison of classifiers in the presence of novel classes using the soybean dataset.

Instances	Existing classes	Novel classes	Classifier	Correctly classified instances (%)
Training: 192 Testing: 90	5	2	EM	95.55
			C4.5	81.11
			k-NN	78.88
Training: 364 Testing: 172	10	3	EM	95.93
			C4.5	71.51
			k-NN	68.02
Training: 630 Testing: 53	15	4	EM	93.23
			C4.5	77.81
			k-NN	75.56

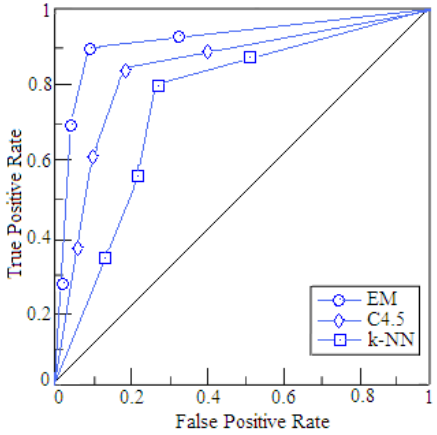


Fig. The ROC curves of classifiers using the entire soybean dataset.

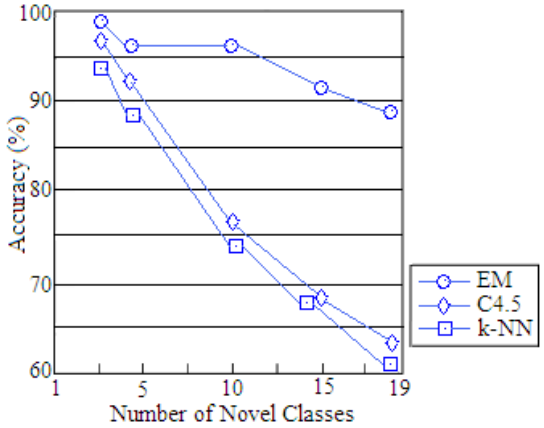


Fig. The comparison of accuracy rate of classifiers in concept drifting using the soybean dataset.

Experimental Analysis (con.)

Table 7
Performance comparison of classifiers in the presence of novel classes using the image segmentation dataset.

Instances	Existing classes	Novel classes	Classifier	Correctly classified instances (%)
Training: 180 Testing: 50	6	1	EM C4.5 k-NN	90.00 40.00 38.00
Training: 987 Testing: 315	5	2	EM C4.5 k-NN	90.13 34.51 33.72
Training: 1343 Testing: 417	4	3	EM C4.5 k-NN	87.53 31.81 31.76

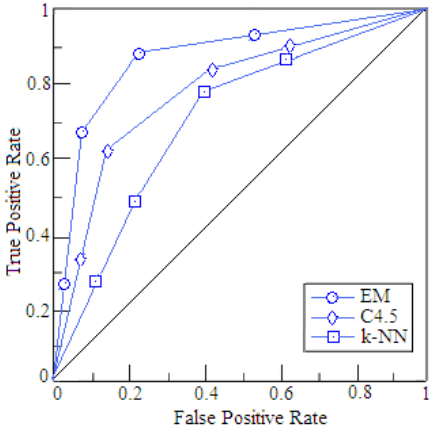


Fig. The ROC curves of classifiers using the entire image dataset.

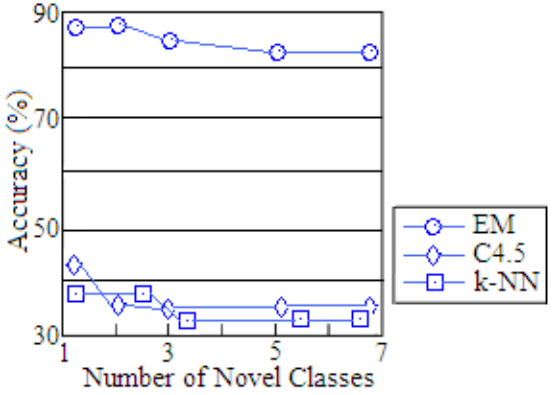


Fig. The comparison of accuracy rate of classifiers in concept drifting using the image segmentation dataset.

Experimental Analysis (con.)

Table 8
Performance comparison of different classifiers.

Dataset	Classifier	M_{new}	F_{new}	ERR
NSL-KDD	EM	0.9	0.4	0.7
	C4.5	1.9	0.9	1.5
	k-NN	5.8	0.9	3.3
Soybean	EM	3.7	0.0	2.0
	C4.5	5.0	1.0	4.0
	k-NN	5.6	1.9	5.1
Image seg.	EM	0.0	0.0	3.3
	C4.5	6.2	2.3	8.0
	k-NN	3.7	3.0	10.4

Table 9
Performance comparison of classifiers using 10-fold cross-validation.

Dataset with instances	Classifier	Classification rate (%)	Misclassification rate (%)
NSL-KDD (25192 instances)	EM	92.65	7.34
	C4.5	86.35	13.64
	k-NN	83.30	16.69
Soybean (683 instances)	EM	98.24	1.75
	C4.5	91.50	8.49
	k-NN	89.75	10.24
Image seg. (2310 instances)	EM	92.77	7.22
	C4.5	90.34	9.65
	k-NN	84.71	15.28

Conclusion & Future Works

Our work addresses

A. Infinite length
B. Limited labeled data
C. Concept drift
D. Concept-evolution

Future works

A. Concept drifting under dynamic feature sets
B. Test instance weighting approach
C. Selecting the most informative test instances

Publication:

Dewan Md. Farid, Li Zhang, Alamgir Hossain, Chowdhury Mofizur Rahman, Rebecca Strachan, Graham Sexton, and Keshav Dahal, "An Adaptive Ensemble Classifier for Mining Concept Drifting Data Streams," *Expert Systems with Applications, Elsevier Science*, Vol. 40, Issue 15, 1 November 2013, pp. 5895-5906 (Impact Factor: 2.981).

- **Questions**