Application of Combined Classifiers to Data Stream Classification

Michał Woźniak

Department of Systems and Computer Networks
Wroclaw University of Technology
Wyb. Wyspianskiego 27, 50-370 Wroclaw, Poland
michal.wozniak@pwr.wroc.pl

Abstract. The progress of computer science caused that many institutions collected huge amount of data, which analysis is impossible by human beings. Nowadays simple methods of data analysis are not sufficient for efficient management of an average enterprize, since for smart decisions the knowledge hidden in data is highly required, as which multiple classifier systems are recently the focus of intense research. Unfortunately the great disadvantage of traditional classification methods is that they "assume" that statistical properties of the discovered concept (which model is predicted) are being unchanged. In real situation we could observe so-called concept drift, which could be caused by changes in the probabilities of classes or/and conditional probability distributions of classes. The potential for considering new training data is an important feature of machine learning methods used in security applications or marketing departments. Unfortunately, the occurrence of this phenomena dramatically decreases classification accuracy.

Keywords: machine learning, classifier ensemble, data stream, concept drift, incremental learning, forgetting.

1 Introduction

Designing such solutions we should take into consideration that in the modern world the most of the data arrive continuously and it causes that smart analytic tools should respect this nature and be able to interpret so-called data streams. They should take into consideration that:

- the statistical dependencies among the observations of given objects and their classifications could change,
- data can come flooding in the analyzer what causes that it is impossible to label all records.

This phenomena is called *concept drift* [1] and it comes in many forms, depending on the type of change. In general, the following approaches can be considered to deal with the mentioned above problem

K. Saeed et al. (Eds.): CISIM 2013, LNCS 8104, pp. 13–23, 2013.

- Rebuilding a classification model if new data becomes available, which is very expensive and impossible from a practical point of view, especially if the concept drift occurs rapidly.
- Detecting concept changes in new data and if these changes are sufficiently "significant", then rebuilding the classifier.
- Adopting an incremental learning algorithm for the classification model.

We will concentrate on the last proposition. Adapting the learner is a part of an incremental learning [2]. The model is either updated (e.g., neural networks) or needs to be partially or completely rebuilt (as CVFDT algorithm [3]). Usually we assume that the data stream is given in a form of data chunks. When dealing with the sliding window the main question is how to adjust the window size. On the one hand, a shorter window allows focusing on the emerging context, though data may not be representative for a longer lasting context. On the other hand, a wider window may result in mixing the instances representing different contexts. Therefore, certain advanced algorithms adjust the window size dynamically depending on the detected state (e.g., FLORA2 [1]) or algorithms can use multiple windows [4]. One of the important group of algorithms dedicated to stream classification exploits strength of ensemble systems, which work pretty well in static environments [5], because according to "no free lunch theorem" [6] there is not a single classifier that is suitable for all the tasks, since each of them has its own domain of competence. A strategy for generating the classifier ensemble [7] should guarantee its diversity improvement and consequently accuracy increasing. Let us enumerate the main propositions how to get a desirable set of individual classifiers:

- The individual classifiers could be train on different datasets, because we hope that classifiers trained on different inputs would be complementary.
- The individual classifiers can use the selected features only.
- Usually it could be easy to decompose the classification problem into simpler ones solved by the individual classifier. The key problem of such approach is how to recover the whole set of possible classes.
- The last and intuitive method is to use individual classifiers trained on different models or different versions of models.

2 Combined Classifier

According to Wolpert's *no free lunch* theorem there is not a single pattern recognition algorithm that is appropriate for all the tasks we are faced with, since each classifier has its own domain of competence [6]. Usually we have a pool of different classifiers at our disposal to solve a given problem. Therefore, methods that can exploit the strengths of individual classifiers are the focus of intense research [8]. There are many relevant works formulated conclusions regarding combined classification quality, such as [9] where a neural network ensemble was considered or [10] where authors dealt with majority voting applied to handwriting recognition. Turner [11] showed that averaging outputs of an infinite number of

unbiased and independent classifiers can lead to the same response as the optimal Bayes classifier. Ho [12] underlined that a decision combination function must receive useful representation of each classifier's decision. Specifically, they considered several method based on decision ranks, such as Borda count. Finally, the landmark works devoted introducing bagging [13] and boosting [14,15] which are able to produce strong classifiers [16], in the (*Probably Approximately Correct*) theory [17] sense, on the basis of the weak ones.

It is important to notice that the classifier ensemble design does not differ from that of a classical pattern recognition [18] application, in which we select the most valuable features and choose the best classification method from the set of available ones. The design of a classifier ensemble aims to create a set of complementary/diverse classifiers and assign an appropriate combination method of individual classifier outputs as well as possible. We have to mention Ho's work [19] who distinguished two main approaches:

- Coverage optimization focus on the generation of a set of mutually complementary classifiers which can be combined to achieve optimal accuracy using a fixed decision combination function.
- Decision optimization concentrates on designing and training an appropriate decision combination function while a set of individual classifier is given in advance [20].

We can distinguish the important issues that must be taken into consideration when building classifier ensemble grouping them into the following problems:

- Proposing the topology i.e., interconnections among individual classifiers in the ensemble.
- Selecting a pool of diverse and complementary individual classifiers for the ensemble.
- Designing a combination rule, aimed at creating a mechanism that can exploit the strengths of the selected classifiers.

Let's focus on the problem of forming a valuable pool of classifiers called ensemble pruning. Selecting members of classifier ensemble with different kinds of components is a key feature of considered system design, because we have to notice that apart from increasing the computational complexity, combining similar classifiers should not contribute much to the combined classifier under construction. An ideal ensemble includes mutually complementary individual classifiers which are characterized by the high diversity and accuracy [21], because we expect that combined classifier accuracy increases according to the diversity increasing of individual classifier pool [22]. First, classifiers must be selected to obtain positive results from their fusion. In [23] Sharkley et al. proposed four level of diversity based on the majority voting rule answer, coincident error, and possibility of at least one correct answer of ensemble members. Brown et al. [24] reflected that it is not appropriate for the case where diversity of an ensemble is differ in differ subspace of the feature space.

3 Classifier Ensembles for Data Stream

Among the most popular ensemble approaches, the following are worth noting: the Streaming Ensemble Algorithm (SEA) [25] or the Accuracy Weighted Ensemble (AWE)[26]. Both algorithms keep a fixed-size set of classifiers. Incoming data are collected in data chunks, which are used to train new classifiers. If there is a free space in the ensemble, a new classifier joins the committee. Otherwise, all the classifiers are evaluated based on their accuracy and the worst one in the committee is replaced by a new one if the latter has higher accuracy. The SEA uses a majority voting strategy, whereas the AWE uses the more advanced weighted voting strategy. A similar formula for decision making is implemented in the Dynamic Weighted Majority (DWM) algorithm [27]. Nevertheless, unlike the former algorithms, the DWM modifies the weights and updates the ensemble in a more flexible manner. The weight of the classifier is reduced when the classifier makes an incorrect decision. Eventually the classifier is removed from the ensemble when its weight falls below a given threshold. Independently, a new classifier is added to the ensemble when the committee makes a wrong decision. The final group consists of algorithms that address the question of when drift occurs. Not all classification algorithms dealing with concept drift, require drift detection. Some evolving systems continuously adjust the model to incoming data [28]. This technique is called implicit drift detection [29] as opposed to explicit drift detection methods that raise a signal to indicate change. The detector can be based on changes in the probability distribution of the instances [30,31,32] or classification accuracy [33,34]. Among the machine learning methods dealing with concept drift, a new class has recently emerged [35], comprising algorithms that process data streams featuring a recurring context. Two additional requirements are imposed on algorithms in this class:

- the system should maintain knowledge of previously emerged contexts, and
- it should be effective in recognizing contexts and switching to a valid one.

Both issues can be effectively solved by an ensemble system. For example, in [35] the authors propose collecting context oriented learners in a "global set" of classifiers along with their selection procedure. The ensemble consists of classifiers that achieve arguably better results than a random classifier. To address the second issue, certain algorithms use a conceptual representation of contexts. This idea originates from Turney's definition of context-sensitive features [36] and the early work by Widmer [37] on meta-learning algorithms. It is based on mapping attributes into contextual clues representing some characteristic features of the respective context. In [38], the authors exploit an additional similarity measure between context representations for weighting a classifier's contribution to the ensemble. The method presented in [39] incorporates a stream clustering algorithm, which aims to group incoming data batches automatically based on their conceptual representation. An incremental classifier is created (and updated if needed) for each cluster/context and stored in the pool. A similar solution can be found in [40], but here classifiers in the pool are adaptively weighted according to their performance on a recent data batch.

4 Exemplary Combined Algorithm for Data Stream Classification

We assume that the classified data stream is given in a form of data chunks denotes as $\mathcal{DS}_k$, where k is the chunk index. The concept drift could appear in the incoming data chunks. We do not detect it, but we try to construct self-adapting classifier ensemble. Therefore on the basis of the each chunk one individual is trained and we check if it could form valuable ensemble with the previously trained models. In our algorithm we propose to use the Generalized Diversity (denoted as $\mathcal{GD}$) proposed by Partridge and Krzanowski [41] to assess all possible ensembles and to choose the best one. $\mathcal{GD}$ returns the maximum values in the case of failure of one classifier is accompanied by correct classification by the other one and minimum diversity occurs when failure of one classifier is accompanied by failure of the other.

$$\mathcal{GD}(\Pi) = 1 - \frac{\sum_{i=1}^{L} \frac{i(i-1)p_i}{L(L-1)}}{\sum_{i=1}^{L} \frac{ip_i}{L}} \tag{1}$$

where L is the cardinality of the classifier pool (number of individual classifiers) and p_i stands for the probability that i randomly chosen classifiers from Π will fail on randomly chosen example.
We can also use modification of $\mathcal{GD}$ called Coincident Failure Diversity ($\mathcal{CFD}$) is the modification of Generalized Diversity proposed by Partridge and Krzanowski as well [41]

$$\mathcal{CFD}(\Pi) = \begin{cases} 0 & p_0 = 1 \\ \frac{1}{1-p_0} \sum_{l=1}^{n} \frac{n-l}{n-1} p_l & p_0 < 1 \end{cases} \tag{2}$$

It returns 0 if the pool Π is not diverse and 1 if each classifier errs on different example.

Lets $P_a(\Psi_i)$ denotes frequency of correct classification of classifier Ψ_i and $itter(\Psi_i)$ stands for number of iterations which Ψ_i has been spent in the ensemble. We propose to establish the classifier's weight $w(\Psi_i)$ according to the following formulae

$$w(\Psi_i) = \frac{P_a(\Psi_i)}{\sqrt{itter(\Psi_i)}} \tag{3}$$

This proposition of classifier aging has its root in object weighting algorithms where an instance weight is usually inversely proportional to the time that has passed since the instance was read [42] and Accuracy Weighted Ensemble (AWE)[26], but the proposed method called Weighted Aging Ensemble (WAE) incudes two important modifications:

1. classifier weights depend on the individual classifier accuracies and time they have been spending in the ensemble,
2. individual classifier are chosen to the ensemble on the basis on the non-pairwise diversity measure.

The WAE pseudocode is presented in Alg.1 [43].

Algorithm 1. Weighted Aging Ensemble (WAE)

Require: input data stream, data chunk size, classifier training procedure, ensemble size L

$i := 1$
repeat
  collect new data chunk DS_i
  train classifier Ψ_i on the basis of DS_i
  add Ψ_i to the classifier ensemble Π
  if $i > L$ **then**
    $\Psi_{k+1} = \Psi_i$
    $\Pi_t = \emptyset$
    $GD_t = 0$
    for $j = 1$ **to** $L + 1$ **do**
      if $\mathcal{GD}(\Pi \backslash \Psi_i)$ (calculated according to (1)) $> GD_t$ **then**
        $\Pi_t = \Pi \backslash \Psi_i$
      end if
    end for
    $\Pi = \Pi_t$
  end if
  $w := 0$
  for $j = 1$ **to** L **do**
    calculate $w(\Psi_i)$ according to (3)
    $w := w + w(\Psi_i)$
  end for
  for $j = 1$ **to** L **do**
    $w(\Psi_i) := \frac{w(\Psi_i)}{w}$
  end for
  $i := i + 1$
until end of the input data stream

5 Experimental Investigations

In this section we will present an illustrative example of WAE for a data stream classification problem. The aims of the experiment were to assess if the proposed method of weighting and aging individual classifiers in the ensemble is valuable proposition compared with the methods which do not include aging or weighting techniques.

5.1 Set-up

All experiments were carried out on the SEA dataset describes in [25]. For each of the experiments we decided to form homogenous ensemble i.e., ensemble which consists of the Naive Bayes classifiers.

During each of the experiment we tried to evaluate dependency between data chunk sizes (which were fixed on 50, 100, 150, 200) and overall classifier quality (accuracy and standard deviation) for the following ensembles:

1. $w0a0$ - an ensemble using majority voting without aging.
2. $w1a0$ - an ensemble using weighted voting without aging, where weight assigned to a given classifier is inversely proportional to its accuracy.
3. $w1a1$ - an ensemble using weighted voting with aging, where weight assigned to a given classifier is calculated according to (3).

Method of ensemble pruning was the same for each ensemble and presented in Alg.1. The only difference was line 19 of the pseudocode what was previously described. All experiments were carried out in the Java environment using Weka classifiers [44].

5.2 Results

The results of experiment are presented in Fig.1-2. The first figure shows the accuracies of the tested ensembles for a chosen experiment. Unfortunately, because of the space limit we are not able to presents all extensive results, but they are

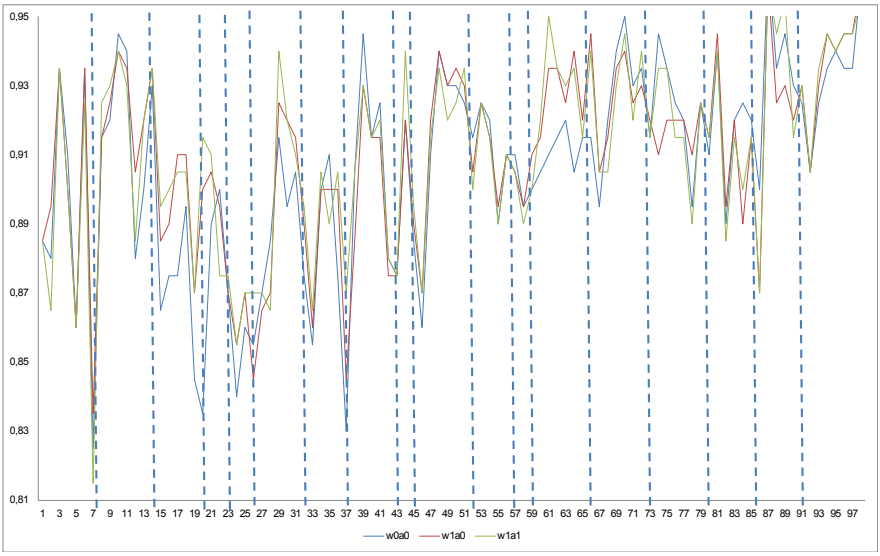

Fig. 1. Classification accuracy of the ensembles consist of Naive Bayes classifiers for the chunk size = 200. Vertical dotted lines indicate concept drift appearances.

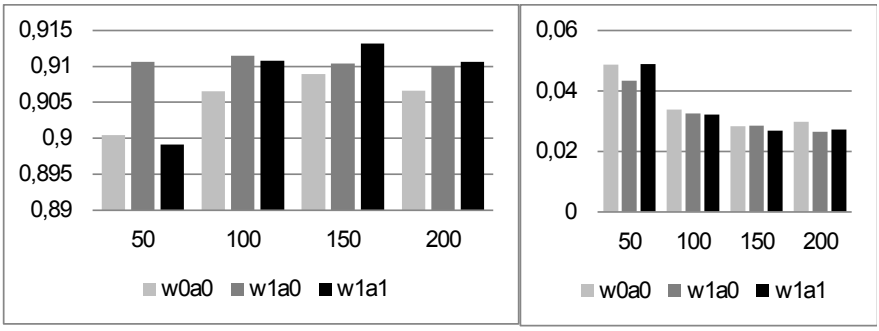

Fig. 2. Classification accuracy (left) and standard deviation (right) of Naive Bayes classifier for different data chunk sizes

available on demand. Fig.2 presents overall accuracy and standard deviation for the tested methods and how they depend on data chunk size.

On the basis of presented results we can formulate several observations. It does not surprise us that quality improvements for all tested method according to increasing data chunk size. Usually the WAE outperformed others, but the differences are quite small and only in the case of ensemble built on the basis of Naive Bayes classifiers the differences are statistical significant (t-test) [45] i.e., differences among different chunk sizes. The observation is useful because the bigger size of data chunk means that effort dedicated to building new models is smaller because they are being built rarely.

Another interesting observation is that the standard deviation is smaller for bigger data chunk and usually standard deviation of WAE is smallest among all tested methods. It means that the concept drift appearances have the weakest impact on the WAE accuracy.

6 Conclusions

We discussed the ensemble classifier approach applied to data stream classification task. We showed an idea and performance of WAE algorithm, which uses dynamic classifier ensemble i.e., its line-up is formed when new data chunk is come and the decision which classifier is chosen to the ensemble is made on the basis of General Diversity (diversity measure). The decision about object's label is made according to weighted voting where weight assigned to a given classifier depends on its accuracy (proportional) and how long the classifier participates in the ensemble (inversely proportional). It is worth noting that classifier ensemble is a promising research direction for aforementioned problem. Maybe its combination with a drift detection algorithm could have a higher impact to the classification performance.

Acknowledgment. The work was supported by the Polish National Science Center under a grant N N519 650440 for the period 2011-2014.

References

1. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Mach. Learn. 23, 69–101 (1996)
2. Muhlbaier, M.D., Topalis, A., Polikar, R.: Learn^{++}.nc: Combining ensemble of classifiers with dynamically weighted consult-and-vote for efficient incremental learning of new classes. IEEE Transactions on Neural Networks 20, 152–168 (2009)
3. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 97–106 (2001)
4. Lazarescu, M.M., Venkatesh, S., Bui, H.H.: Using multiple windows to track concept drift. Intell. Data Anal. 8, 29–59 (2004)
5. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience (2004)
6. Wolpert, D.H.: The supervised learning no-free-lunch theorems. In: Proc. 6th Online World Conference on Soft Computing in Industrial Applications, pp. 25–42 (2001)
7. Wozniak, M., Grana, M., Corchado, E.: A survey of multiple classifier systems as hybrid systems. Information Fusion (2013)
8. Jain, A., Duin, R., Mao, J.: Statistical pattern recognition: a review. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 4–37 (2000)
9. Hansen, L., Salamon, P.: Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence 12, 993–1001 (1990)
10. Xu, L., Krzyzak, A., Suen, C.: Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Transactions on Systems, Man and Cybernetics 22, 418–435 (1992)
11. Tumer, K., Ghosh, J.: Analysis of decision boundaries in linearly combined neural classifiers. Pattern Recognition 29, 341–348 (1996)
12. Ho, T.K., Hull, J.J., Srihari, S.N.: Decision combination in multiple classifier systems. IEEE Trans. Pattern Anal. Mach. Intell. 16, 66–75 (1994)
13. Breiman, L.: Bagging predictors. Mach. Learn. 24, 123–140 (1996)
14. Schapire, R.E.: The strength of weak learnability. Mach. Learn. 5, 197–227 (1990)
15. Freund, Y.: Boosting a weak learning algorithm by majority. Inf. Comput. 121, 256–285 (1995)
16. Kearns, M.J., Vazirani, U.V.: An introduction to computational learning theory. MIT Press, Cambridge (1994)
17. Angluin, D.: Queries and concept learning. Mach. Learn. 2, 319–342 (1988)
18. Giacinto, G., Roli, F., Fumera, G.: Design of effective multiple classifier systems by clustering of classifiers. In: Proceedings of the 15th International Conference on Pattern Recognition, vol. 2, pp. 160–163 (2000)
19. Ho, T.K.: Complexity of classification problems and comparative advantages of combined classifiers. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 97–106. Springer, Heidelberg (2000)
20. Roli, F., Giacinto, G.: Design of Multiple Classifier Systems. World Scientific Publishing (2002)
21. Krogh, A., Vedelsby, J.: Neural network ensembles, cross validation, and active learning. In: Advances in Neural Information Processing Systems, vol. 7, pp. 231–238 (1995)

22. Zenobi, G., Cunningham, P.: Using diversity in preparing ensembles of classifiers based on different feature subsets to minimize generalization error. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 576–587. Springer, Heidelberg (2001)
23. Sharkey, A.J.C., Sharkey, N.E.: Combining diverse neural nets. Knowl. Eng. Rev. 12, 231–247 (1997)
24. Brown, G., Wyatt, J.L., Harris, R., Yao, X.: Diversity creation methods: a survey and categorisation. Information Fusion 6, 5–20 (2005)
25. Street, W.N., Kim, Y.: A streaming ensemble algorithm (sea) for large-scale classification. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2001, pp. 377–382. ACM, New York (2001)
26. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2003, pp. 226–235. ACM, New York (2003)
27. Kolter, J., Maloof, M.: Dynamic weighted majority: a new ensemble method for tracking concept drift. In: Third IEEE International Conference on Data Mining, ICDM 2003, pp. 123–130 (2003)
28. Zliobaite, I.: Change with delayed labeling: When is it detectable? In: Proceedings of the 2010 IEEE International Conference on Data Mining Workshops, ICDMW 2010, pp. 843–850. IEEE Computer Society, Washington, DC (2010)
29. Kuncheva, L.I.: Classifier ensembles for detecting concept change in streaming data: Overview and perspectives. In: 2nd Workshop SUEMA 2008 (ECAI 2008), pp. 5–10 (2008)
30. Gaber, M.M., Yu, P.S.: Classification of changes in evolving data streams using online clustering result deviation. In: Proc. of International Workshop on Knowledge Discovery in Data Streams (2006)
31. Markou, M., Singh, S.: Novelty detection: a review—part 1: statistical approaches. Signal Process. 83, 2481–2497 (2003)
32. Salganicoff, M.: Density-adaptive learning and forgetting. In: Machine Learning: Proceedings of the Tenth Annual Conference. Morgan Kaufmann, San Francisco (1993)
33. Klinkenberg, R., Joachims, T.: Detecting concept drift with support vector machines. In: Proceedings of the Seventeenth International Conference on Machine Learning, ICML 2000, pp. 487–494. Morgan Kaufmann Publishers Inc., San Francisco (2000)
34. Baena-García, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavaldá, R., Morales-Bueno, R.: Early drift detection method. In: Fourth International Workshop on Knowledge Discovery from Data Streams (2006)
35. Ramamurthy, S., Bhatnagar, R.: Tracking recurrent concept drift in streaming data using ensemble classifiers. In: Proceedings of the Sixth International Conference on Machine Learning and Applications, ICMLA 2007, pp. 404–409. IEEE Computer Society, Washington, DC (2007)
36. Turney, P.D.: Exploiting context when learning to classify. In: Brazdil, P.B. (ed.) ECML 1993. LNCS, vol. 667, pp. 402–407. Springer, Heidelberg (1993)
37. Widmer, G.: Tracking context changes through meta-learning. Mach. Learn. 27, 259–286 (1997)

38. Bártolo Gomes, J., Ruiz, E.M., Sousa, P.A.C.: Learning recurring concepts from data streams with a context-aware ensemble. In: Chu, W.C., Wong, W.E., Palakal, M.J., Hung, C.C. (eds.) Proceedings of the 2011 ACM Symposium on Applied Computing (SAC), TaiChung, Taiwan, March 21-24, pp. 994–999. ACM (2011)
39. Katakis, I., Tsoumakas, G., Vlahavas, I.: Tracking recurring contexts using ensemble classifiers: an application to email filtering. Knowl. Inf. Syst. 22, 371–391 (2010)
40. Hosseini, M.J., Ahmadi, Z., Beigy, H.: Pool and accuracy based stream classification: A new ensemble algorithm on data stream classification using recurring concepts detection. In: Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops, ICDMW 2011, pp. 588–595. IEEE Computer Society, Washington, DC (2011)
41. Partridge, D., Krzanowski, W.: Software diversity: practical statistics for its measurement and exploitation. Information and Software Technology 39, 707–717 (1997)
42. Klinkenberg, R., Renz, I.: Adaptive information filtering: Learning in the presence of concept drifts, pp. 33–40 (1998)
43. Wozniak, M., Kasprzak, A., Cal, P.: Application of combined classifiers to data stream classification. In: FQAS 2013. LNCS(LNAI), vol. 8132, pp. 579–588. Springer, Heidelberg (2013)
44. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. 11, 10–18 (2009)
45. Alpaydin, E.: Introduction to Machine Learning, 2nd edn. The MIT Press (2010)