An Enhanced Adaptative System based on Machine Learning for Predicting the Evolution of Islamic Stock Prices

KAOUTAR ABBAHADDOU, MOHAMMED SALAH CHIADMI, RAJAE ABOULAICH
Mohammed V University in Rabat
MOROCCO

Abstract: - This paper suggests an enhanced machine-learning-based system to guide future stock price decisions. In reality, most existing machine learning systems, such as SEA (Stream Ensemble Algorithm), VFDT (Very Fast Decision Tree), and online bagging and boosting, keep models updated with only new data and reduce training timeframes to allow working rapidly with the most recent model. However, limited learning times and the exclusion of essential information from previous data may result in a bad performance. When it comes to learning models, our system takes a different approach. It builds several models based on random selections of historical data from the main stock as well as related stocks. The best models are then combined to generate a final, performant model. We performed an empirical study on five Islamic stock market indices. We can say from the results that our system outperforms the existing published algorithms. This framework can contribute then to having an enhanced system that will enable different stakeholders to make rapid decisions based on the forecasted trend of indices.

Key-Words: - Machine learning, Forecasting, Prediction, Data Science, Stock Market, Islamic finance.

Received: September 23, 2019. Revised: August 27, 2022. Accepted: September 15, 2022. Published: October 11, 2022.

1 Introduction

Predicting stock market direction is a fertile field of research for many individuals and financial institutions that are interested in finding a proper framework to build their strategies. The goal is to find an accurate decision support system that guides traders regarding future stock price direction. A technique that helps a trader to decide whether to buy or sell a stock and helps him significantly affects his strategy and profitability over time.

Practitioners often employ one of the main four methodologies when forecasting the direction of stock prices, where three of them are conventional. The fundamental methodology is the first one and it helps to investigate some variables related to the economy and influencing the stock price. The second strategy uses conventional technical analysis based on historical data indicators of pricing and volume. This technical analysis seeks to find patterns in data by taking those patterns out and visually examining the stock charts. The third strategy for predicting market direction is known as quantitative technical analysis, developed later from technical analysis into a more quantitative and statistical method, which makes it simple to perform and helps to automate the rules.

On another note, given the ongoing research to improve this area of market prediction, machine learning, and deep learning have found a home in financial applications thanks to recent advancements in technology and algorithms and their ability to predict time series data with high levels of accuracy. Machine learning (ML), which represents the fourth strategy is mainly a branch of data science that makes predictions by using statistical models. In that sense, models learn from past data and automatically alter their settings to enhance or forecast the results.

Nevertheless, when using machine learning for predicting the direction of stock data, three key motifs need special attention: First, the fact that the market changes frequently. Thereby, machine learning models became outdated with the same frequency. Second, data need to be rapidly processed to produce quick and accurate forecasting. Finally, we need to have a consistent volume of data to enable the models to learn enough and make accurate forecasting.

The learning of models is complicated by these elements. Model performance may decline as these factors change, necessitating an update to the training data and/or a change in the model itself. Some attempts to overcome these limitations were done by using contemporary machine learning classification algorithms, which offer some solutions but the results differ and we still have room for improvements.

In that sense, this research is about suggesting an enhanced framework that uses machine learning classification techniques to enable the predictability of stock market direction while avoiding these

limitations along with an empirical study on Islamic stock data that aims to compare our framework with existing ones.
Our research starts with a review of the literature on market forecasting techniques based on machine learning. In the next section, we explore our framework and how it addresses the cited limitations of the different techniques cited in the literature review. This includes all the methodologies and approaches used later in this article for the comparative analysis with current practices. Finally, we analyze our results and conclude our article.

2 Literature Review

Markets are fundamentally noisy and do not remain stable. This was confirmed by many researchers who stated that ultimately the predicted market events and patterns will disappear over time [1]. In that sense, using only the most recent data to train models may be inappropriate for all cases. Therefore, finding means to predict and develop models that follow the market dynamically is the ideal solution to this issue.
One of the earliest techniques for updating models based on the emergence of data changes is the supervised algorithm, the Floating Rough Approximation (FLORA), developed by Kubat and Widmer starting in 1989 [2, 3]. Thanks to this technique, learning occurs within a moving window on portions of data with a fixed size. FLORA method will then progressively remove the portions that have not been used for a time and retrain the classifier.
Later on, in 1992, the performance of the algorithm was taken into account by adjusting the sliding window size in an upgraded version, by analyzing the accuracy of the algorithm, known as FLORA2. Later, FLORA3 was also released to help preserve and search among old models previously trained for later use and check each time if the old models could better explain the events in a determined window. Nevertheless, it was discovered that FLORA3 was unstable during extremely noisy periods. This resulted in the development of FLORA4 to improve the missed features of FLORA3 as explained earlier.
In 1998, Klinkenberg and Renz used measurements like recall, precision, and accuracy to identify the timing when drastic changes occur by tracking how they changed over time [4]. For each of these performance measurements, the standard error and the average value are first calculated using the most recent batches from a sliding window. A change happens if the values were less than the confidence interval of times.
One of the most well-known algorithms for usage with learning while data change was conceptualized by Hulten and Domingos in 2000. It is called the VFDT, which stands for the Very Fast Decision Tree method [5]. It functions like a decision tree that learns on batches of data (like the C4.5 method), which must be completely rebuilt as soon as a new training instance is received. It incorporates training instances as the data is received.
Street and Kim in 2001 with the Streaming Ensemble Algorithm (SEA), don't explicitly check for changes but instead operate under the percept that they do change [6]. This approach was developed because the latest data is not necessarily always the needed data to have a successful learning phase. By using SEA, some instances from a batch of data are used to create a classifier. The latter is compared to a group of classifiers trained by using prior batches of data. In case the quality of forecasting is improved, it is added to the group and the weakest classifier is deleted.
Another method using an ensemble of classifiers was provided by Wang et al. in 2003 [7]. This method adopts one classifier from the latest batch of data and other classifiers trained on earlier batches. The ensemble is created as a weighted ensemble of classifiers, where the weights are equivalent to the performance of these classifiers. Cross-validation is used to decide on the performance of the classifiers by using the most recent sliding windows. According to their findings, training an ensemble outperformed training just one classifier by using identical data.
In 2004, to answer the questions of whether it is preferable to train models using the most recent data only or both recent and former data; and how old should the data be? Fan [8] picked up where Wang et al. [7] left off. He concluded a series of studies that using old data is useful when we don’t notice changes in the stream of data and when there is not enough new data. If there is a change in data, historical data is still helpful if the new and old concepts are consistent and a strategy for selecting the best historical data for model construction can be adopted.
Another wrapping method is offered in the same year 2004 by Chu and Zaniolo [9], which uses ensemble learning to accelerate learning on concepts that are changing. The process starts by dividing the data into equal-sized groups, and then, like AdaBoost, the algorithm gives misclassified samples heavier weights. After the sample weights have been normalized, a classifier is created using

these weighted groups. The forecasting is made by each classifier, and a mean rule adds up the probabilities of the forecast; the answer is the classifier with the maximum probability.
Also, in 2004 Gama et al. issued another technique named the Drift Detection Method (DDM), based on tracking some rates to determine when a change occurred [10]. DDM determines the probability of misclassification and the standard deviation for each item in the data. The probability will then decline as the sample size grows if the distribution of the sample is stationary and its statistical characteristics do not change over time. That way, the latest constructed model becomes incompatible with the most recent data if the error rate of the learning algorithm rises noticeably, suggesting then changes in the distribution of classes and indicating the need to update the model.
The majority of ensemble methods call for learning by batches or repeated treatment of the full dataset. The ensemble classifier is cumbersome to utilize for big streaming datasets since each base classifier requires one pass over the dataset. By developing a version of the Boosting and Bagging algorithms that can be used online, Oza resolves this issue in 2005 [11]. Because the model already contains all of the training examples that have been seen so far, this online version only needs to process the training batch once, without taking into consideration how many classifiers we have in the ensemble. This eliminates storing some models for further processing.
In the same year 2005, Law and Zaniolo issued the Adaptive Nearest Neighbor Classification Algorithm (ANNCAD) [12]. By partitioning data into distinct groups of the same size, ANNCAD speeds up the process by taking into account progressively equal chunks of data because the nearest neighbor algorithm can be slow and compute extensively. A threshold value is determined to accurately distinguish between the classes. The model then uses exponential fading to gradually ignore outdated data. One of the disadvantages of this model is that unexpected changes can be unreported because of the gradual learning process.
Another approach is provided by Bifet and Gavalda in 2007 named the Adaptive Window algorithm (ADWIN) [13]. It uses changeable windows in the same way as FLORA. Yet, their algorithm modifies the window size after assessing the changes in data. The window size expands in the absence of change and decreases in the presence of changes. By using ADWIN, the older portion of a sliding window is dropped when two sub-windows show distinct averages. This part is retained meanwhile until a statistical test can fully reject it.
In 2013, we had the Accuracy Updated Ensemble algorithm (AUE2) by Stefanowski and Brzezinski [14] that employs Very Fast Decision Trees (VFDT) incrementally in a wrapper architecture. On uniformly sized chunks, each containing instances, the VFDT incrementally builds classifiers. The classifiers are sorted and assessed for each incoming chunk. A classifier that is built on the specific set of data used previously to evaluate the other classifiers, is used itself to replace the classifier that performed the worst and is referred to as the main classifier. Then, all classifiers are weighted using a formula that gives the best classifiers a bigger weight and the main classifier the heaviest weight. The selected classifiers are then disclosed gradually and added to a weighted ensemble.
As we can see, a variety of models were put up to address some of the difficulties associated with machine learning. The goal of our article is to develop an enhanced machine-learning-based decision support system inspired by some benefits of the current practices to forecast the development of Islamic stock indexes. This work is important because as the literature review the current systems have many limitations that we will try to solve through our framework.

3 Problem Solution

3.1 Our framework

We can see from the literature review that there are several methods for figuring out how to forecast the direction of stock market returns. Nevertheless, some limitations are noticed like the fact that the most recent data is not always the best option, since stocks frequently exhibit recurring behaviors. In that sense, it can be useful to use information from earlier days, weeks, and years. On the other hand, many of these methods focus on one model while it can be highly practical to be able to employ many conventional classifiers like decision trees (DT), support vector machines (SVM), and artificial neural networks.
In this part, we present our enhanced framework for predicting the short-term movement of stock prices which can be seen in figure 1. The benefits of this method include the capacity to transfer knowledge held within other stocks in addition to the capacity to employ conventional classifiers, operate in parallel, and work with earlier models. Our system largely eliminates bottlenecks that are present in conventional techniques, such as the need to wait

for classifier training to be finished before implementing predictions.
Step one relates to training classifiers using random-length samples of prior data to optimize concepts as possible and to provide enough material for generalization. A pool of models is expanded then as the classifiers are trained. We employ a variety of classifier types that are detailed later in section 3.2. In addition, we train also classifiers using data from the highly correlated stocks, as the learning is better when using training data that is similar to the data of the stock, we are projecting is used.
Step two involves testing each classifier from the pool on examples from the latest batches of data to assess its quality. A measure of performance is used to assess classifier performance that is presented later on, in section 3.3.
Steps 1 and 2 continuously train and test new classifiers throughout operations on the data stream.
Step three involves selecting the best classifiers from the group as determined by using the evaluation of the latest data.
Step four involves using the newly formed ensemble to forecast the direction of the future stock price results by weighting the result of each classifier with the evaluation of the confidence of the classifier from the previous step.

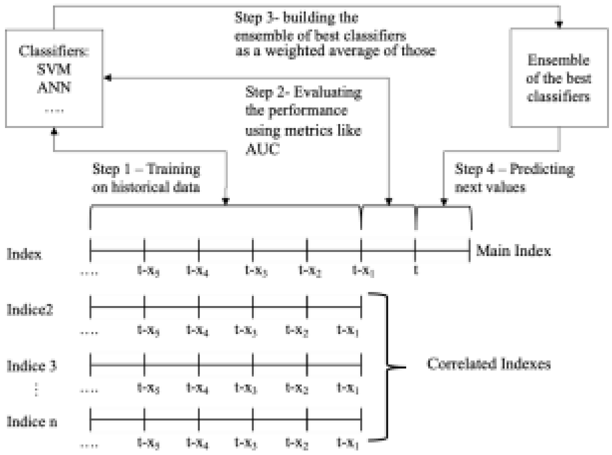

Fig. 1: The enhanced adaptative system for predicting the evolution of stock prices

Our framework offers many advantages. First, we enable shorter computation times as the procedure is easily distributed, particularly in steps one and two. Next, we extend the volume of data available to train all classifiers as we use data from correlated stocks. Finally, our group of classifiers can embrace some that train slowly, and others that compute extensively.

To define the right and most suitable size of data to train classifiers with changes, we adopt a trial and error approach, which can be problematic. If we consider a small volume of data, it may include just some events that are not enough. While if we consider a large volume of data, we can have many events that can override or contradict each other and will cause a decline in terms of performance. To address this issue, we suggest training our classifiers on batches of data with random lengths. Our goal is to train all the classifiers in the framework and decide on the performance of each one by using the latest batch of data to select the best one to create an ensemble.

3.2 Selection of Machine Learning Model

In step 1, we have to identify the pool of conventional classifiers to train them throughout the process. In our case, we include three distinct basic classifiers for our experiment, namely, support vector machines (SVM), artificial neural networks (ANN), and decision trees (DT). Knowing that the methodology is general and can include other models.

3.2.1 Support Vector Machines (SVMs)

SVMs were created as a result of Vladimir Vapnik's work in the 1990s [15]. SVMs were soon embraced in machine learning because of their efficiency in handling vast amounts of knowledge, lack of hyperparameters, theoretical solidity, and practical effectiveness.
The fundamental idea behind SVM is to seek out the simplest hyperplane as an answer to an optimization problem with restrictions to separate the classes. To implement SVMs, we used the R package "e1071" for SVMs.

3.2.2 Artificial Neural Networks (ANNs)

Artificial neural networks (ANNs) are central and related to deep learning which belongs to machine learning. These networks function like the brain of humans and the biological neurons and that is what inspired their structure and name.
Artificial neural networks (ANNs) are made from several distinct nodes or neuron layers. We can have a first layer that collects inputs, a certain number of layers that hide inside, and the last layers for the outputs. On different layers, we have different artificial neurons that are connected with a certain threshold and weight. A node can become active if its output value is larger than the threshold. Once it is activated, data is sent then to the subsequent network layer.

In our algorithmic implementation, we used the R 'neuralnet' package for that.

3.2.3 Decision Tree C4.5:

C4.5 is an algorithm designed to generate a call tree created by Quinlan Ross [16]. C4.5 constructs decision trees after training on data and composing its nodes. For each node, the tree chooses the characteristic of the information that almost all efficiently divides its sample set into sub-groups fed by some classes. The splitting criterion is the normalized information gain or the entropy difference. The attribute with the very best normalized information gain is chosen to form the choice. Algorithm C4.5 then returns to the partitioned sub-lists.
We originally used the "party" package, which could be a package for decision trees, for its implementation on R. With the assistance of this package, we were ready to create our model using the function named "ctree".

3.3 The Performance Measure

In step 2 of our system, we need to select a performance measure. In our case, we choose to assess our models with the AUC. This is a very popular metric that is well established and used more often compared to metrics like the error rate. This latter is not performant when we have to deal with data that is skewed because accuracy will be higher within the entities with a bigger number. Moreover, this latter ignores as well the confidence of the forecast, while it is well handled by the AUC.
AUC is essentially related to the ROC curve. The ROC (Receiver Operating Characteristic) curve has been used in signal processing to distinguish between signal and noise. It is widely used in machine learning to evaluate the performance of classifiers. This is a curve where we cross the rate of true positives (TVP) with that of false negatives (TFN) for all the classification thresholds. We use takes a baseline that represents the random ranking of our instances. The closer the ROC curve is to the upper corner, the better the classification performance.
The probability threshold can be a positive value smaller than one in theory and with this kind of value, we can have many collections of forecasted answers. We construct a confusion matrix equivalent to each collection of forecasts and therefore in that case the evaluation metrics are forced to be modified. This is shown in the diagram below for a threshold of 0.5. The area under the ROC curve (AUC: Area Under the Curve) represents a measure that allows us to numerically quantify the performance of our classifiers:

• if AUC = 1, This is a model that makes a perfect separation between our classes. It classifies all positive instances correctly and does the same with other instances.
• if AUC = 0.5, the classification is not better than that which would be obtained if we randomly generate our instances. The model in this case makes no distinction between our classes.
• if AUC < 0.5, our model does worse than a random classification. It is better to guess randomly than to use this pattern.

The AUC is very useful when you want to make a comparison between different models as we can identify the best one by looking for the highest score.

3.4 Models for Benchmarks

To be able to measure and review the performance of our framework, we compare it with the following well-established models as seen in the literature as benchmarks.

3.4.1 Streaming Ensemble Algorithm (SEA)

The algorithm was suggested by Street and Kim in 2001 [6]. It reads several entities from previous data and uses it to construct a classifier. This latter is compared to a set of classifiers that were trained on preceding data batches. If the performance of the set is better, it is integrated while the weakest classifier is removed. This set of classifiers is used afterward to predict subsequent entities.

3.4.2 Very Fast Decision Tree (VFDT)

The second model we will use for the comparison is the Very Fast Decision Tree (VFDT) which uses a C4.5 decision tree. The VFDT uses batches of data with a fixed size to adapt models to changes, the result is very similar to the C4.5 decision tree. The concept is based on embedding learning instances when data comes in. The tree needs the data set to be treated completely in the learning phase.

3.4.3 Online Bagging and Boosting Algorithms

The last benchmark is the online boosting and bagging algorithm by Oza. We will use a method adapted and approximated to it [17].
We have first, Online Bagging which stands for Online Boostrap aggregating. For this method, the models are generated independently and trained in parallel. The motivation behind these methods is that the prediction error can be significantly reduced by combining the predictions. Concretely, the

bagging process performs a sampling of the data and trains the algorithms separately on each of these samples. It then assembles the results of the models obtained using a voting system on the final result.
Second, we have Online Boosting where the base classifiers are generated sequentially and dependently like AdaBoost, unlike parallel methods. Each time a base classifier is trained, the previously misclassified instances are weighted with a higher weight so that in the next iterations, the new models correct the errors of the previous models, which should improve the overall performance.

4 Empirical Results

4.1 Data and Pre-Processing

We used in our study two main Islamic stock market indices that are used separately for our comparative analysis:
- Morgan Stanley Capital International Islamic (MSCII), which began in 2007 and includes over sixty countries.
- Jakarta Islamic Index (JKII) which began in 2000. It focuses on the food field and represents thirty businesses.

As we need to have to use correlated indices for our system to train our models, we use the two following Islamic stock indices:
- Dow Jones Islamic Market Index (DJIMI) was established in 1999 and includes 66 countries and 90 indices across many areas and sectors that are Shariah compliant.
- Financial Time Stock Exchange Shariah (FTSE Shariah) was created in January 1999 by the Islamic investment bank. Its goal is to monitor the performance of prestigious publicly traded businesses whose operations comply with Islamic Sharia.

We have ten years of historical daily data. Our data is composed of the following information: High, Low, Open, Close, Volume, and Adj. Close. We have to prepare our data for avoiding outliers and missing values, which we either removed or substituted them using some methods like the mean.
Our algorithms are designed to compute both high and low signals. The first one will inform us is the price will increase and the second one will inform us of the contrary. So we can confirm that it would be a classification problem.

4.2 Analysis of the Impact of Changes on the Learning of Models

Before we analyze the performance of our system, it is important to show the effect of changes on the performance of our classifiers. For this first analysis, the dataset of each of our two main indices MSCII and JKII was divided into two parts, eighty percent of the data is dedicated to training our models and twenty percent for testing models with intervals of 10 days. We calculate then the AUC throughout the intervals on the test data to see the evolution of the performance measure as seen in the following figure 2.
Using the AUC and by analyzing the charts in figure 2, we notice that the performance of the model decreases as the tests move away from the training data for both indices and all models. Mainly, the SVM that used the radial kernel for greater precision, the ANN in which we employed five layers with nine neurons each and the sigmoid activation function, and finally C4.5 whose outcome was outcome is a two-layer tree. This indicates the need to update the learning data progressively. Hence the need for adaptive models. Therefore, we can firmly confirm that as the market dynamics change, the model's performance may decrease, which requires a constant update of the Learning.

4.3 Analysis of the Performance of our System

Intending to assess the performance of our system, we first assess the performance against it with well-established benchmarks (SEA, CVFDT, online boosting, and online bagging). The experiment is done on each of the main indexes MSCII and JKII along with the correlated stock indexes (FTSE and DJIMI). Data was divided here also, 80% is used for the training and 20% for the test with intervals of 10 days. We calculated the AUC for our system and each benchmark, each interval, and each main index. This enables us to rank all the methodologies according to the AUC. Intending to compare the global performance in all those cases, we calculate the avg rank of each system for all these experiments as we can see in table 1.
Following the average rank of the benchmark classifiers in Table 1, we should note that the lower the rank, the better the model is because it means that it was ranked higher more often. From this table, excluding our system, we can say that the Online bagging benchmark is the classifier that performs the best, and the Very Fast Decision Tree is the worst. This can be explained by the fact that VFDT uses fixed-length windows for training,

which is not the case for other methods. In general, Online bagging comes first, second, we have Online boosting, SEA, and then VFDT. Following the results, we can confirm as well that our system comes first, which can be explained by the use of the correlated indices data, the fact that we don't ignore historical data, and the choice of models that keeps the top performant and includes the C4.5 tree that used in some of the benchmarks.

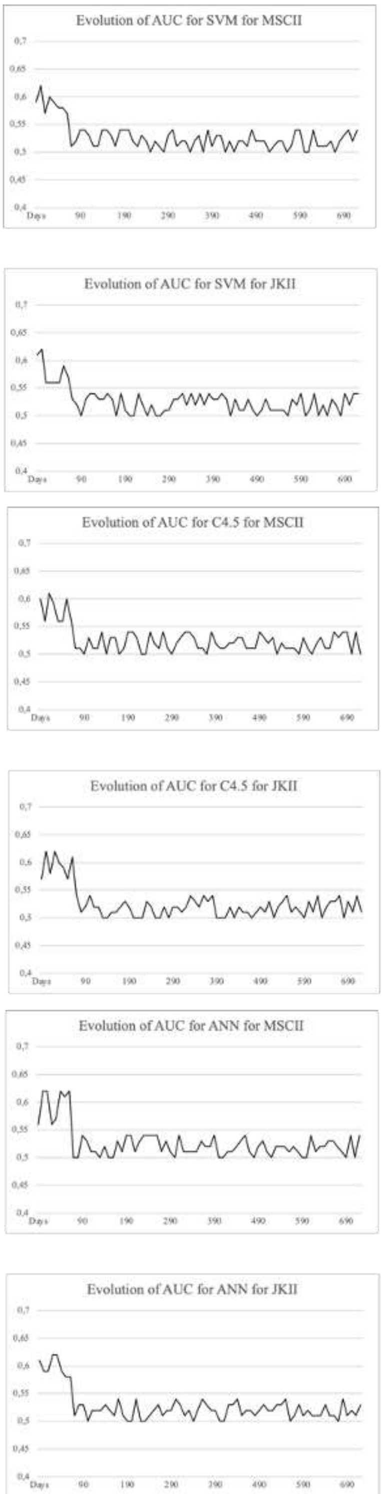

Fig. 2: The performance of models using AUC on test data of Islamic stock indices

Table 1. Average rank of different systems

Systems	Avg. Rank
VFDT	4,2
Online bagging	2.8
Online boosting	3
Streaming Ensemble Algorithm (SEA)	3,4
The enhanced model	1,6

5 Conclusion

Forecasting the trend of the stock market remains a difficult task that interests many stakeholders in the industry and academic field. In this context, we attempt through this article to explore this subject, mainly since we can find only a few numbers published frameworks considering machine learning that can do this forecast. We described first our enhanced framework for forecasting the stock trend. The existing systems offer many advantages, like the capacity to use well-established classifiers, learn simultaneously, and use previous events concepts. Our framework not only offers these advantages but can in addition to that gain knowledge in additional related indices. The concept behind our framework is to build many classifiers that use different batches of data with random lengths from the main index but also from similar indices. We, then, merge the top performers that are selected after assessing them on the latest interval of data. The knowledge gained from the related stocks improves the overall performance of our models without increasing the number of models. However, our framework has some limitations as it addresses today just classification problems. There are still several tracks that we wish to explore and enhance in our framework, like the inclusion of regression problems and the inclusion of statistical models along with the performance measures that are suitable to these methodologies. This will be subject to future research, and for which we wish to bring some answers during our next research.

References:

[1] Guidolin, M., & Timmermann, A. (2008). International asset allocation under regime switching, skew, and kurtosis preferences. The Review of Financial Studies, 21(2), 889-935.

[2] Kubat, M. (1989). Floating approximation in time-varying knowledge bases. Pattern recognition letters, 10(4), 223-227.

[3] Widmer, G., & Kubat, M. (1992). Learning flexible concepts from streams of examples: FLORA2.

[4] Klinkenberg, R., & Renz, I. (1998). Adaptive information filtering: Learning in the presence of concept drifts. Learning for text categorization, 33-40.

[5] Domingos, P., & Hulten, G. (2000, August). Mining high-speed data streams. In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 71-80).

[6] Street, W. N., & Kim, Y. (2001, August). A streaming ensemble algorithm (SEA) for large-scale classification. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 377-382).

[7] Wang, H., Fan, W., Yu, P. S., & Han, J. (2003, August). Mining concept-drifting data streams using ensemble classifiers. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 226-235).

[8] Fan, W., Ellacott, K. L., Halatchev, I. G., Takahashi, K., Yu, P., & Cone, R. D. (2004). Cholecystokinin-mediated suppression of feeding involves the brainstem melanocortin system. Nature Neuroscience, 7(4), 335-336

[9] Chu, F., & Zaniolo, C. (2004, May). Fast and light boosting for adaptive mining of data streams. In Pacific-Asia conference on knowledge discovery and data mining (pp. 282-292). Springer, Berlin, Heidelberg.

[10] Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004, September). Learning with drift detection. In Brazilian symposium on artificial intelligence (pp. 286-295). Springer, Berlin, Heidelberg.

[11] Srivastava, A. N., Oza, N. C., & Stroeve, J. (2005). Virtual sensors: Using data mining techniques to efficiently estimate remote sensing spectra. IEEE Transactions on Geoscience and Remote Sensing, 43(3), 590-600.

[12] Law, Y. N., & Zaniolo, C. (2005, October). An adaptive nearest neighbor classification algorithm for data streams. In European Conference on Principles of Data Mining and Knowledge Discovery (pp. 108-120). Springer, Berlin, Heidelberg.

[13] Bifet, A., & Gavalda, R. (2007, April). Learning from time-changing data with adaptive windowing. In Proceedings of the 2007 SIAM international conference on data mining (pp. 443-448). Society for Industrial and Applied Mathematics.

[14] Brzezinski, D., & Stefanowski, J. (2013). Reacting to different types of concept drift: The accuracy updated ensemble algorithm. IEEE Transactions on Neural Networks and Learning Systems, 25(1), 81-94.

[15] Vapnik, V. (1999). The nature of statistical learning theory. Springer science & business media.

[16] Salzberg, S. L. (1994). C4. 5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993.

[17] Oza, N. C., & Russell, S. J. (2001, January). Online bagging and boosting. In International Workshop on Artificial Intelligence and Statistics (pp. 229-236). PMLR.