Adaptive Least-Squares Support Vector Machine and Its Online Learning

Yevgeniy Bodyanskiy[a], Anastasiia Deineko[a], Filip Brodetskyi[b] and Danylo Kosmin[a]

[a]*Kharkiv National University of Radio Electronics, Artificial Intelligence Department, Nauky av., 14, Kharkiv, 61166, Ukraine*

[b]*Kharkiv National University of Radio Electronics, Department of Informatics, Nauky av., 14, Kharkiv, 61166, Ukraine*

Abstract

In this paper the adaptive learning method for least-squares support vector machine (LS-SVM) is proposed. Essential feature of this method is that the minimization criterion of empirical risk is realized on the sliding window of fixed dimension that essentially simplifies numerical implementation of the procedure and allows to process information generated by non-stationary nonlinear objects.

Keywords

Neural networks, kernel function, least-squares support vector machine, empirical risk criterion, sliding window

1. Introduction

For solving wide class of tasks like information processing, system and object identification, first of all significantly nonlinear in conditions of structure and parametric uncertainty, artificial neural networks are wieldy used, because of its universal approximation properties and ability to learn. One of the effective neural network to solve this task are least-squares support vector machines that however couldn't be used for processing increasing data sets, when data are fed to the system in online mode. In the paper adaptive approach for learning LS-SVM in online mode using "sliding window" that permits to solve wide class of the tasks in the common problem of the Data Stream Mining is proposed.

Support vector machine is neural hybrid system that combines both learning based on optimization and memory (so-called, lazy learning) and realizes method of empirical risk optimization. The key concept in the synthesis of this network are support vectors that form a small subset of the most informative data vectors allocated in the learning process. Support vector machines are really efficient neural networks in conditions of small datasets, that provide high quality approximation.

ICT&ES-2020: Information-Communication Technologies & Embedded Systems, November 12, 2020, Mykolaiv, Ukraine

yevgeniy.bodyanskiy@nure.ua (Y. Bodyanskiy); anastasiya.deineko@gmail.com (A. Deineko); filip.brodetskyi@nure.ua (F. Brodetskyi); danya.kosmin@gmail.com (D. Kosmin)

0000-0001-5418-2143 (Y. Bodyanskiy); 0000-0002-3279-3135 (A. Deineko); 0000-0002-0300-3886 (F. Brodetskyi)

CEUR Workshop Proceedings (CEUR-WS.org)

2. Adaptive learning method for LS-SVM neural network

The main disadvantage of conventional SVM [1, 2] is the numerical cumbersomeness of the synaptic weight determination procedure, that is reduced to the problem of nonlinear programming with inequality constraints, the number of which is determined by the size of the learning sample. From this point of view more effective are support vectors machines based on the least squares method, however, one way or another both neural networks process data only in batch mode.

The transformation realized by the support vector machine can be written in the form

$$\widehat{y}^{SV}(x) = (w_x^{sv})^T \varphi^{sv}(x) + w_0^{sv} \tag{1}$$

where $w_x^{sv} = (w_1^{sv}, ..., w_l^{sv}, ..., w_{hsv}^{sv})^T$, $\varphi^{sv}(x) = (\varphi_1^{sv}(x), ..., \varphi_{hsv}^{sv}(x))^T$ and its learning (in the case of LS-SVM) [3] is reduced to the simultaneous setting of the activation functions centers at the points of the training dataset $x(l)$, $l = 1, 2, ..., k$ like in the GRNN [4, 5] and optimization of the quadratic criterion

$$E^{sv}(k) = \frac{1}{2}(w_x^{sv})^T w_x^{sv} + \frac{\gamma}{2}\sum_k e^2(k)$$

in the presence of system of $h_{sv} = k$ linear constraints-equations:

$$\begin{cases} y(1) = (w_x^{sv})^T \varphi^{sv}(x(1)) + w_0^{sv} + e(1), \\ ... \\ y(k) = (w_x^{sv})^T \varphi^{sv}(x(k)) + w_0^{sv} + e(k) \end{cases}$$

where $\gamma > 0$ - regularization parameter,

$$e(l) = y(l) - \widehat{y}^{sv}(x(l)), l = 1, 2, ..., k.$$

In the batch mode LS-SVM tuning is connected with finding the saddle point of the Lagrange function

$$\begin{aligned} L(w_x^{sv}, w_0^{sv}, e(k), \lambda(k)) = \\ = E^{sv}(k) + \sum_k \lambda(k)(y(k) - (w_x^{sv})^T \varphi^{sv}(x(k)) - w_0^{sv} - e(k)) = \\ = \frac{1}{1}(w_x^{sv})^T w_x^{sv} + \frac{\gamma}{2}\sum_k \lambda(k)(y(k) - (w_x^{sv})^T \varphi^{sv}(x(k)) - w_0^{sv} - e(k)) \end{aligned} \tag{2}$$

in addition besides synaptic weights w_x^{sv} and w_0^{sv} also k indefinite Lagrange multipliers $\lambda(l)$ must be found.

The system of Kuhn-Tucker equations for Lagrangian (2) can be written in the form

$$\begin{cases} \nabla_{w_x^{sv}} L = w_x^{sv} - \sum_k \lambda(k)\varphi^{sv}(x(k)) = \overrightarrow{0_k}, \\ \frac{\partial L}{\partial w_x^{sv}} = -\sum_k \lambda(k) = 0, \\ \frac{\partial L}{\partial e(l)} = \gamma e(l) - \lambda(l) = 0, \\ \frac{\partial L}{\partial \lambda(l)} = y(l) - (w_x^{sv})^T \varphi(x(l)) - w_0^{sv} - e(l) = 0 \end{cases}$$

here $\overrightarrow{0_k}$ – $(k \times 1)$ – vector formed by zeros) or

$$\begin{cases} w_x^{sv} = \sum_k \lambda(k)\varphi^{sv}(x(k)), \\ \sum_k \lambda(k) = 0, \\ \lambda(l) = \gamma e(l), \\ y(l) - (w_x^{sv})^T \varphi(x(l)) - w_0^{sv} - e(l) = 0 \end{cases} \tag{3}$$

From the first equation of the system (3) it follows that the synaptic weights depend entirely of the values of the indefinite Lagrange multipliers, in connection with which the LS-SVM training is reduced to their definition, and the system (3) can be rewritten in a compact form:

$$\begin{pmatrix} 0 & I_K^T \\ I_k & \Omega(k) + \gamma^{-1} I_{k,k} \end{pmatrix} \begin{pmatrix} w_0^{sv} \\ \Lambda(k) \end{pmatrix} = \begin{pmatrix} 0 \\ Y(k) \end{pmatrix} \tag{4}$$

where $\Lambda(k) = (\lambda(1), ..., \lambda(l), ..., \lambda(k))^T$, I_k – $(k \times 1)$ – vector formed by unities, $I_{k,k}$ – $(k \times k)$ – $i = 1, 2, ..., k, j = 1, 2, ..., k$, $Y(k) = (y(1), ..., y(l), ..., y(k))^T$, $\Omega(k) = \left\{\Omega_{ij} = (\varphi^{sv})^T(x(i))\varphi^{sv} = K(x(i), x(j))\right\}$, $K(x(i), x(k))$ – some kernel function that satisfies the conditions of Mercer's theorem, often the same Gaussian [6]

$$K(x(k), x(k)) = exp\left(-\frac{\|x(i) - x(k)\|^2}{2\sigma^2}\right) \tag{5}$$

In this case, the transformation (1) implemented by the support vector machine can be rewritten in the form

$$\widehat{y}^{sv}(k) = \sum_k \lambda(k) K(x, x(k)) + w_0^{sv}$$

and its parameters $\lambda(k)$, w_0^{sv} can be found directly from (4) as

$$\begin{pmatrix} w_0^{sv} \\ \Lambda(k) \end{pmatrix} = \begin{pmatrix} 0 & I_K^T \\ I_k & \Omega(k) + \gamma^{-1} I_{k,k} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ Y(k) \end{pmatrix} = p^{sv}(k) \begin{pmatrix} 0 \\ Y(k) \end{pmatrix} \tag{6}$$

It is clear that the LS-SVM adaptive learning may be organized based on the numerically simple procedure for matrix inversion in the right part of the system (6).

Rewriting (6) for (k+1)-th time moment as

$$\begin{pmatrix} w_0^{sv} \\ \Lambda(k+1) \end{pmatrix} = \begin{pmatrix} 0 & I_K^T \\ I_k & \Omega(k+1) + \gamma^{-1} I_{k+1,k+1} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ Y(k+1) \end{pmatrix} =$$

$$= \begin{pmatrix} & & 1 \\ 0 & I_k^T & K(x(1), x(k+1)) \\ I_k & \Omega(k) + \gamma^{-1} I_{k,k} & K(x(2), x(k+1)) \\ & & \vdots \\ & & 1 + \gamma^{-1} \end{pmatrix} =$$

$$= \begin{pmatrix} 0 \\ Y(k) \\ y(k+1) \end{pmatrix} = \begin{pmatrix} (P^{sv}(k))^{-1} & \overrightarrow{K}(x(i), x(k+1)) \\ \overrightarrow{K}^T(x(i), x(k+1)) & 1 + \gamma^{-1} \end{pmatrix} \times \begin{pmatrix} \overrightarrow{Y}(k) \\ y(k+1) \end{pmatrix}$$

here $\overrightarrow{K}^T(x(i), x(k+1)) = (1, K(x(1), x(k+1)), ..., K(x(k), x(k+1)))^T$, $\overrightarrow{Y}^T(k) = (0, Y^T(k))$ and applying the Frobenius formula for the inversion of block matrices [7], we obtain at a simple expression for calculating the matrix $P^{sv}(k+1)$

$$P^{sv}(k+1) =$$

$$= \begin{pmatrix} P^{sv}(k) + (P^{sv}(k)\overrightarrow{K}(x(i), x(k+1)))\times & -(P^{sv}(k)\overrightarrow{K}(x(i), x(k+1))) \\ \times\overrightarrow{K}^T(x(i), x(k+1)P^{sv}) & (1+\gamma^{-1} - \overrightarrow{K}^T(x(i), x(k+1))) \\ (1+\gamma^{-1} - \overrightarrow{K}^T(x(i), x(k+1))) & P^{sv}(k)\overrightarrow{K}(x(i), x(k+1))^{-1} \\ P^{sv}(k)\overrightarrow{K}(x(i), x(k+1))^{-1} & \\ -(\overrightarrow{K}^T(x(i), x(k+1))P^{sv}(k)) & \\ (1+\gamma^{-1} - \overrightarrow{K}^T(x(i), x(k+1)) & (1+\gamma^{-1} - \overrightarrow{K}^T(x(i), x(k+1)) \\ P^{sv}(k)\overrightarrow{K}(x(i), x(k+1))^{-1} & P^{sv}(k)\overrightarrow{K}(x(i), x(k+1))^{-1} \end{pmatrix} \quad (7)$$

It is clear that with large volumes of the training set, the inversion of the $(k \times k)$ – matrices is much easier to do using formula (7).

3. The LS-SVM neural network learning on the "sliding window"

It should be kept in mind that as the training data set usually grows in time, so does the number of neurons in the neural network, which will sooner or later lead to "curse of dimensionality".

That is why, it is necessary, in cases that the object which generates data is non-stationary, that is necessary in situation when object generate non-stationary data (medical data sets, time series of forecasting electricity consumption, exchange rates). Thus, in this regards better to organize information processing on the "sliding window" [8, 9], that includes s last observations, which, in turn, will lead to the fact that the neural network will be formed by s nodes. For processing non-stationary data more commonly used exponential weighting of old information method, that in our situation can leads to a significant increasing of kernel functions in the hidden layer of proposed LS-SVM and this operation will make system very bulky.

In this case, introducing into consideration the $(s \times s)$ – kernel function matrix $\Omega(k, s) = \{\Omega_{ij}\varphi^{sv^T}(x(i)\varphi^{sv}x(j)) = K(x(i), x(j), s), i = k-s+1, k-s+2, ..., k; j = k-s+1, k-s+2, ..., k\}$ transformation implemented by a neural network with a fixed number of neurons can be written in the form

$$\hat{y}^{sv}(x, k, s) = \sum_{l=k-s+1}^{k} \lambda(l, s)K(x, x(l), s) + w_0^{sv}(k, s).$$

Thereof parameters $\lambda(l, s)$, $w_0^{sv}(k, s)$can be found by solving a matrix equation of type (6)

$$\begin{pmatrix} w_0^{sv} \\ \Lambda(k, s) \end{pmatrix} = \begin{pmatrix} 0 & I_s^T \\ I_s & \Omega(k, s) + \gamma^{-1}I_{s,s} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ Y(k, s) \end{pmatrix} = p^{sv}(k, s) \begin{pmatrix} 0 \\ Y(k, s) \end{pmatrix} \quad (8)$$

where $\Lambda(k,s) = (\lambda(k-s+1,s),...,\lambda(k-s+1,s),...,\lambda(k,s))^T$, $Y(k,s) = (y(k-s+1),...,y(k-s+2),...,y(k))^T$.

When $(k+1)$-th observation is fed to processing, it should be calculated into $\Omega(k+1,s)$ matrix and in the same time from this matrix should be deleted observation that was fed to processing in $k-s+1$-th moment of time. Herewith

$$\begin{pmatrix} w_0^{sv}(k+1,s) \\ \Lambda(k+1,s) \end{pmatrix} = \\ = \begin{pmatrix} 0 & I_s^T \\ I_s & \Omega(k+1,s) + \gamma^{-1} I_{s,s} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ Y(k+1,s) \end{pmatrix} = P^{sv}(k+1,s) \begin{pmatrix} 0 \\ Y(k+1,s) \end{pmatrix} \tag{9}$$

where $\Lambda(k+1,s) = (\lambda(k-s+2,s),...,\lambda(k-s+3,s),...,\lambda(k+1,s))^T$, $Y(k+1,s) = (y(k-s+2),...,y(k-s+3),...,y(k+1))^T$.

It is easy to see, that relations (8), (9) are essentially an online adaptive algorithm for learning the neural network of fixed architecture.

4. Experimental results

In experimental modeling were investigated tuning of LS-SVM [10] model on the several data sets [10, 11, 12, 13] and the influence of the choosing model parameters on the results of data processing and classification. For first example data set "Double Donut" was taken, this data set includes two discharged circles one inside the other artificially generated. To the input of the neural system the number of observations, the noise level and the factor of their scale between the inner and outer circles were transmitted. On the Figure 1 initial classification of data set "Double Donut" is shown.

In the described earlier SVM model there are two parameters that affect learning. Also, it is necessary to choose the kernel function for learning LS-SVM model. The polynomial, radial-basis (Gaussian) or linear function can be used for tuning neural network SVM.

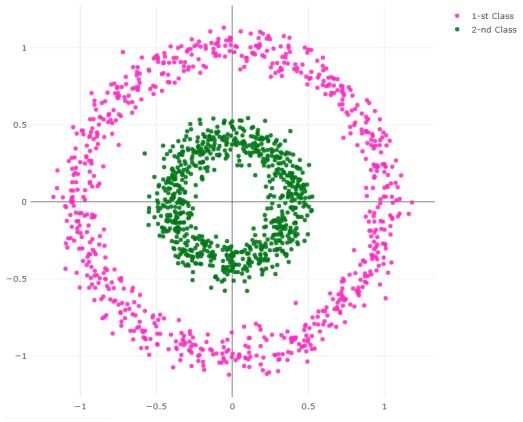


Figure (1): Initial classification of the data set "Double Donut"

For comparison of classification results linear, radial-basis and polynomial activation functions were used. On the Figure 2 results of classification by SVM model with linear activation function are demonstrated.

Next, as the activation function the radial basis kernel (5) was used, in which the standard width parameter, which is adjusted manually was taken, let is note that in the learning process criterion (10) was optimized

$$\Phi(\lambda) = \sum_{k=1}^{N} \lambda_k - \frac{1}{2} \| \sum_{k=1}^{N} \lambda_k y^{(k)} x^{(k)} \|^2 \tag{10}$$

Thus, for adequate classification it is necessary to choose the parameter so that the classifier is not very general, and not overfitted.

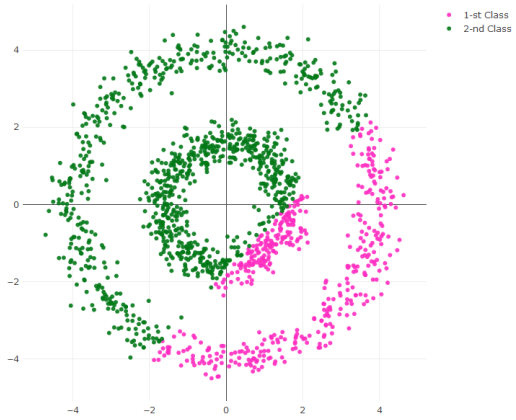
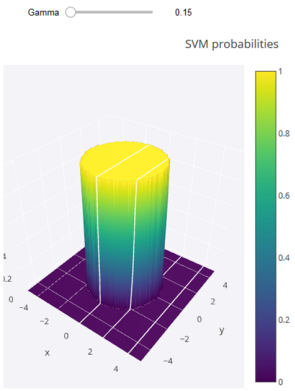


Figure (2): Classification results by SVM model with linear activation function

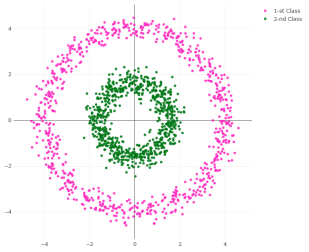
Modifying the output layer of SVM neural network obtained probability of occurrence of each of the classes. Due to the probabilities of belonging to the class, the 3D graph that will show the probability of a given point to one of the classes can be built. Consider the influence of the parameter σ from formula (5) on the classification results. The regularization parameter γ should be selected:

$$\gamma = \frac{1}{2\sigma^2} \tag{11}$$

Increasing of this parameter lead to increasing of SVM overfitting and becomes less common. If parameter γ is equal to 0.15 classification results are shown in the Figure 3a and 3b.



(a) The surface of probability of the 2nd class occurrence



(b) The classification results of the SVM with radial basis kernel

Figure (3): The classification results

Thus, it is easy to see that radial basis kernel was chosen correctly. And as can be seen at the Figure 3 separating hypersurface is constructed correctly. But also it should be noted that all observations that are outside the outer circle will belong to the first class, and others – to the second. It is interesting to see what will be with increasing of the parameter γ . At the Figure 4 is show result of tuning SVM model with $\gamma = 10$. The influence of the parameter γ is illustrated on the Figure 5. Here as an example SVM model with radial basis kernel was used.

Based on the results obtained in the first experiment, we can conclude that for this sample the parameter of the fine does not have much effect. This may be due to the fact that at these location parameters the dividing hyperplane is found quickly and the error during training is very small.

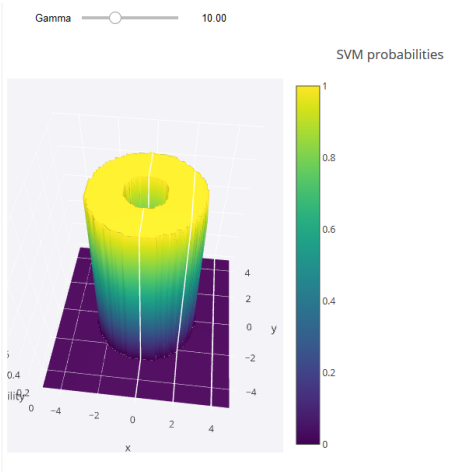


Figure (4): The surface of probability of the second class occurrence (with $\gamma = 10$)

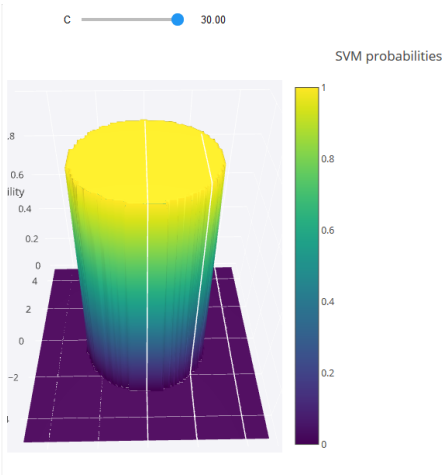


Figure (5): The influence of the parameter γ at the classification

Second part of experimental modeling was made on the data set “Double helix”. This data set was generated by mathematical equations and it could not be linearly separated. This data set represents the two helixes, which are inside one another. Figure 6 shows what this data set looks like.

The comparative analysis of classification the support vector machine with linear activation function and the support vector machine with kernel activation function were held. The results of the experiment represented in Figure 7 - Figure 10.

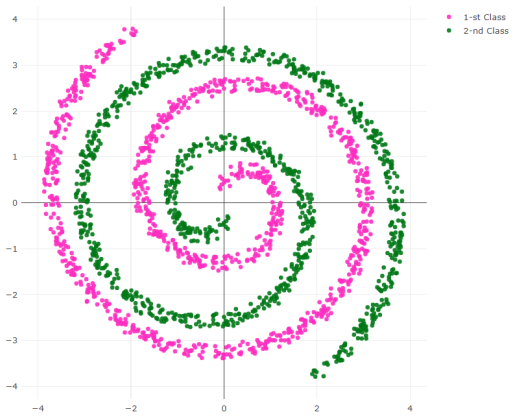


Figure (6): Data set “Double helix”

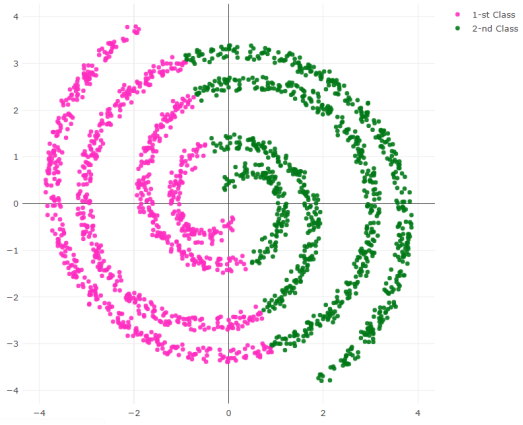


Figure (7): The classification results data set “Double helix” (SVM with linear kernels)

As can be seen in the Figure 7, the linear classifier divides the observation space into two halves, minimizing, as far as possible, the classification error. Since the linear classifier does not give the desired result, it is necessary to use the radial basis kernel.

For learning SVM model with kernel activation function the probabilities surface of one of the classes was developed. The probabilities surface of belonging to the first class with penalty function and kernel parameter equal to one is presented in the Figure 8.

As can be seen, the surface does not describe this data set well, as the model parameters are selected incorrectly for good classification. This is due to the fact that the small value of the penalty function and radial basis function parameters for complex models create an insufficiently trained model but generalize it. However, with increased parameters, the model becomes overfit. The results of classification are represented at the Figure 9.

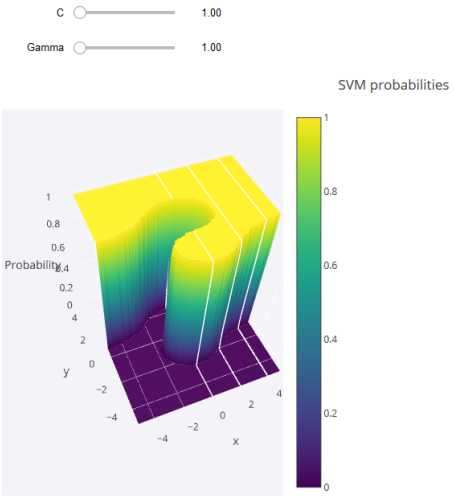


Figure (8): The classification results data set “Double helix” (SVM with linear kernels)

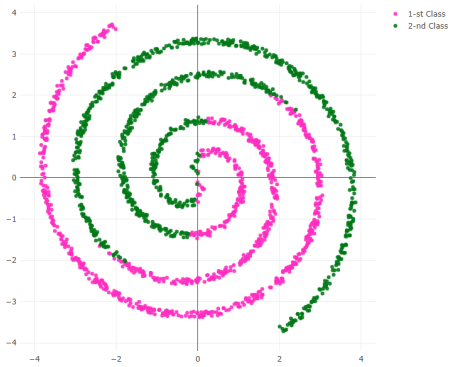
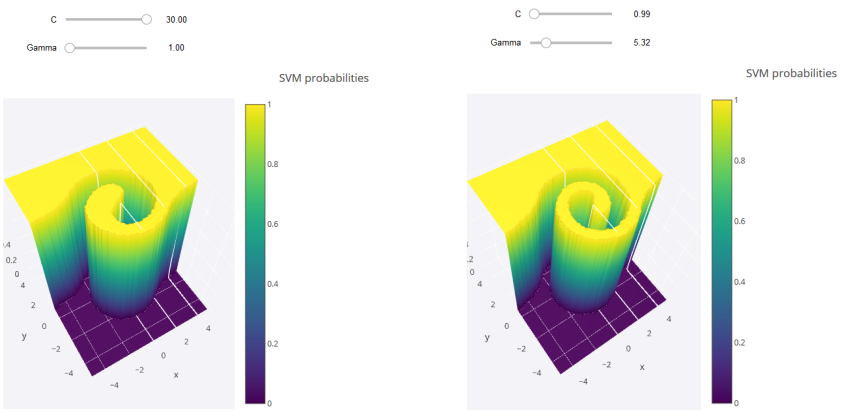


Figure (9): The results of classification with small values of tuned parameters

As can be seen in the Figure 9, the classification of classes is not done well enough to correctly classify even those observations that were used in the model learning. Because the model is very general and cannot create a valid dividing hyperplane, as the model parameters do not allow to create a more complex model.

To solve this problem, it is necessary to increase the penalty function parameter or parameter of radial basis function. Graphs of probability surfaces of the first class with increasing parameter γ and radial basis function represented in the Figure 10a and 10b.

Based on the results presented in the Figures 10a and 10b, can be said that changing both of the parameters greatly influences the classification results. At very large values of the parameters, the model loses the ability to generalize and can classify only those observations that were in the training sample, which is presented in Figure 11.



(a) Increasing of the penalty function parameter

(b) Increasing of the radial basis function parameter

Figure (10): Graphs of probability surfaces

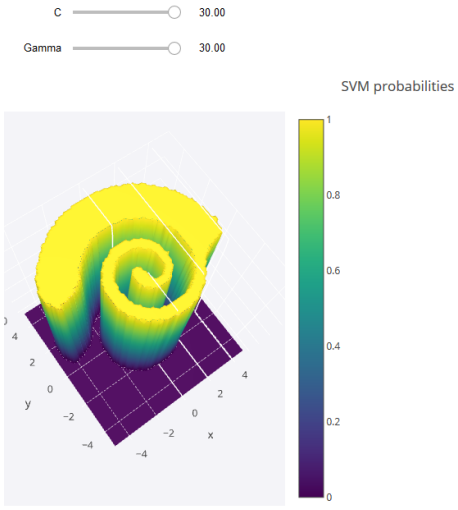


Figure (11): The classification results with very large values of the parameters

In the next series of experiments data set «Breast Cancer Wisconsin Diagnostic» was used. This data set consist of observations that describe breast cancer diseases of Indian women. This data set contains features that were calculated based on the digitized results of fine-needle aspiration taken from the chest weight. Also, for each of the observations the classification attribute with correct mark is presented: "M" - malignant, "B" - benign. Because this data set is high dimensional for visualization principal component analysis was used for compression initial data. The compression results based on the principal component analysis are shown in the Figure 12.

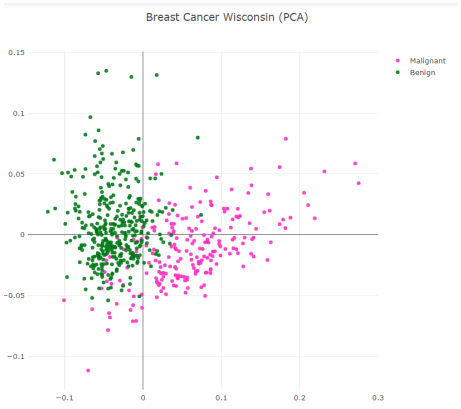


Figure (12): The compressed data set "Breast Cancer Wisconsin"

As can be seen in this figure observation are not linearly separable. Next 3D compression was made for building separating hyperplane. In the Figure 13 is demonstrate results of 3D compression.

Table (1)
The numerical results (LS-SVM with linear kernel)

Model score	Malignant score	Benign score
0.850615	0.608490	0.994397

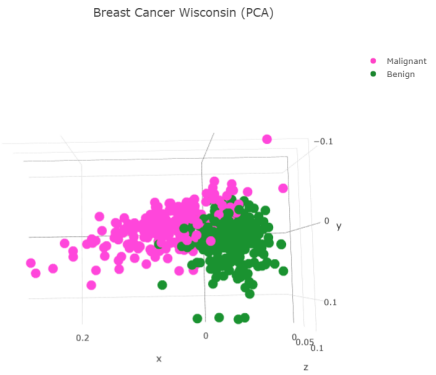
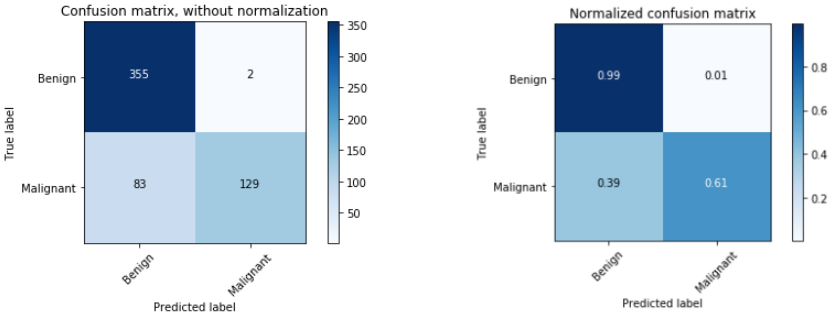


Figure (13): Compressed data set in the 3D space

The classification results of data set «Breast Cancer Wisconsin Diagnostic» by LS-SVM with linear kernel are presented in the table 1.

As can be seen from these results, the precisions of the model is 85%, among malignant neoplasms correctly classified about 60%, and among benign - about 99.5%. Based on these data, we can say that the classification of malignant neoplasms by the model of the LS-SVM does not give good enough results, as only much more than half of the observations were correctly classified. But the classification of malignant neoplasms gives a relatively good result.

The developed model is not good enough to use because a probability of 85% does not provide an sufficiently adequate result to inform the patient or for using to make a diagnosis. The matrix of the LS-SVM with linear kernel errors is shown in the Figure 14.



(a) quantitative estimate (b) percentage estimate

Figure (14): The matrix of the LS-SVM with linear kernel errors

Table (2)
The numerical results (LS-SVM with radial basis kernel)

Model score	Malignant score	Benign score
1.0	1.0	1.0

Table (3)
The numerical results (LS-SVM with radial basis kernel after retraining)

Model score	Malignant score	Benign score
0.95614	0.96875	0.95121

Figure 15 represented final classification using the LS-SVM model with linear kernel.

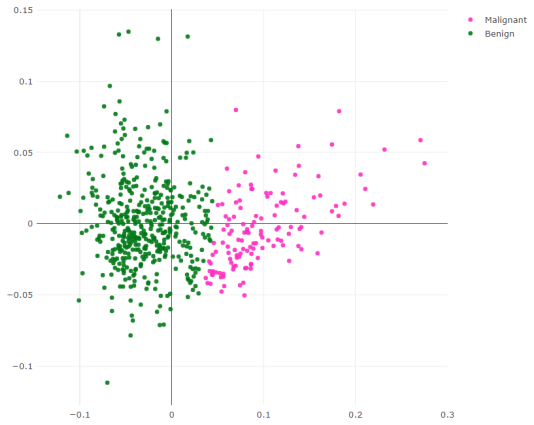
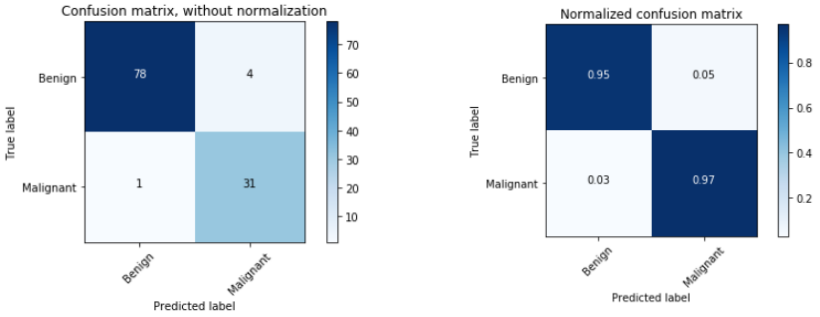


Figure (15): The final classification using the LS-SVM model with linear kernel

Thus, after analyzing these results, it is clear that using the linear core for this model is not a good enough solution. For improving classification results let is activation function and used redial basis kernel. The results of the classification by the LS-SVM model with radial basis kernel are represented in the table 2.

As can be seen from these results, the model of radial basis kernel gives 100% correct classification, which is practically impossible with real world data. This fact may indicate that the model is overfeted. In this situation better to divided initial data set into training and testing sets (size of the testing set is 20% of the total sample). The results of the classification by the LS-SVM model with radial basis kernel after retraining are represented in the table 3.

The accuracy of the testing sample is more than 95%, which is a good indicator. And almost 97% of malignant neoplasms were found correctly. Error matrices are presented in Figure 16.



(a) quantitative estimate (b) percentage estimate

Figure (16): The matrix of the LS-SVM with radial basis kernel errors

Figure 17 represented final classification using the LS-SVM model with redial basis kernel.

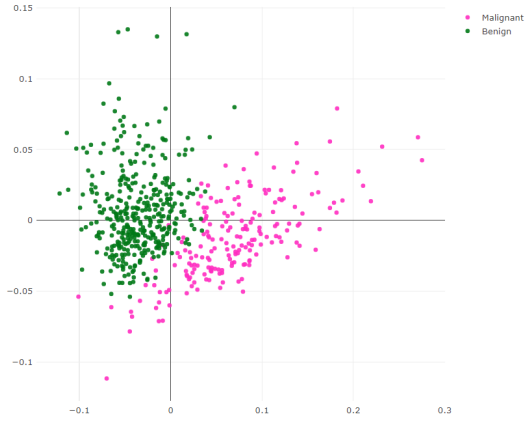


Figure (17): The final classification using the LS-SVM model with radial basis kernel

The experimental research confirms effectiveness of proposed approach for solving task of Big Data Mining in situation than these data are sequentially fed to processing in online mode.

5. Conclusions

The adaptive learning method for least square support vector machine neural network (LS-SVM) with fixed architecture was proposed. The distinctive feature of this method is that the empirical risk minimization criterion occurred on the fixed dimension sliding window, which simplifies the numerical implementation procedures and allows to process information generated by non-stationary objects.

The main benefit of the investigated approach is this method could be used in situation when observations are fed to process in the online mode from nonlinear and nonstationary objects in conditions of outliers in input data. Also proposed system does not suffer from the curse of dimensionality because amount of radial basis function in the hidden layer is limited by the size

of the sliding window that helps to protect from the inherited to SVM and LS-SVM curse of dimensionality.

References

[1] V. N. Vapnik, A. J. Chervonenkis, Pattern Recognition Theory (The Nature of Statistical Learning Theory), Nauka, Moscow, 1974.

[2] Y. V. Bodyanskiy, A. O. Deineko, F. M. Eze, Kernel fuzzy kohonen's clustering neural network and it's recursive learning, Automatic Control and Computer Sciences 52 (2018) 166–174. doi:10.3103/S0146411618030045.

[3] J. A. K. Suykens, T. V. Gestel, J. D. Brabanter, B. D. Moor, J. Vandewalle, Least Squares Support Vector Machines, World Scientific, Singapore, 2002. doi:10.1142/5089.

[4] D. F. Specht, A general regression neural network, ieee trans. on neural networks, IEEE Transactions on Neural Networks 2 (1991) 568–576. doi:10.1109/72.97934.

[5] Y. V. Bodyanskiy, A. O. Deineko, Y. V. Kutsenko, On-line kernel clustering based on the general regression neural network and t. kohonen's self-organizing map, Automatic Control and Computer Sciences 51 (2017) 55–62. doi:10.3103/S0146411617010023.

[6] I. Izonin, M. Gregus, R. Tkachenko, P. Tkachenko, N. Kryvinska, P. Vitynskyi, Committee of sgtm neural-like structures with rbf kernel for insurance cost prediction task, in: 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON 2019), IEEE, 2019, pp. 1037–1040. doi:10.1109/UKRCON.2019.8879905.

[7] F. R. Gantmacher, The Theory of Matrices, Chelsea Publishing Company, New York, 1959. doi:10.1126/science.131.3408.1216-a.

[8] O. Herman-Saffar, Time based cross validation, 2020. URL: https://towardsdatascience.com/time-based-cross-validation-d259b13d42b8.

[9] L. L. Peterson, B. S. Davie, Computer Networks: A Systems Approach, Morgan Kaufmann, 2000.

[10] S. Haykin, Neural Networks. A Comprehensive Foundation, Prentice Hall, Inc., New Jersey, 1999.

[11] Jupyter notebook documentation, 2020. URL: http://jupyter.org/documentation.

[12] Scikit-learn documentation, 2020. URL: http://scikit-learn.org/stable/documentation.html.

[13] Numpy library documentation, 2020. URL: http://www.numpy.org.