UDC 519.925.51
DOI: 10.20535/SRIT.2308-8893.2022.4.03

COMPARATIVE ANALYSIS OF MODIFIED SEMI-SUPERVISED LEARNING ALGORITHMS ON A SMALL AMOUNT OF LABELED DATA

L.M. LYUBCHYK, K.S. YAMKOVYI

Abstract. The paper is devoted to improving semi-supervised clustering methods and comparing their accuracy and robustness. The proposed approach is based on expanding a clustering algorithm for using an available set of labels by replacing the distance function. Using the distance function considers not only spatial data but also available labels. Moreover, the proposed distance function could be adopted for working with ordinal variables as labels. An extended approach is also considered, based on a combination of unsupervised *k*-medoids methods, modified for using only labeled data during the medoids calculation step, supervised method of *k* nearest neighbor, and unsupervised *k*-means. The learning algorithm uses information about the nearest points and classes' centers of mass. The results demonstrate that even a small amount of labeled data allows us to use semi-supervised learning, and proposed modifications improve accuracy and algorithm performance, which was found during experiments.

Keywords: center of mass, clustering, distance function, medoids, nearest neighbor, semi-supervised learning.

INTRODUCTION

A large amount of data was produced recently, and nowadays humanity has the opportunity to store and process all this data. In all spheres of life, people try to use various data for optimizing business and life-improving using AI and data mining.

There are several approaches to data processing and analysis problems within the framework of machine learning (ML) paradigms. One of them is unsupervised learning when one tries to detect inner structure or patterns without human supervision. The most efficient approach in ML is supervised learning when we have some data with labels and try to learn a model function on data points as pairs of feature vectors and suitable labels. In many cases, there is no opportunity to label all data from different cases, causes are too complex and expensive experiments, data streaming with large frequency or just high cost of data labeling. Therefore, in this case, a satisfactory compromise is semi-supervised learning [1, 2], when we use datasets with a small amount of label that

allows learning better its inner structure, which is illustrated by (Fig. 1).

Semi-supervised learning includes a variety of different approaches and can be used for any popular data analysis problems, such as clustering, anomaly detection, latent variables models, and many overs. In this paper, the object of the study is the process of the data points classifications, namely, identifying to which of a set of categories a new observation belongs to using a training set of data containing observations whose category membership is known for a piece of data. The purpose is to develop an improved combined semi-supervised method using already existing supervised and unsupervised algorithms and compare their accuracy and robustness.

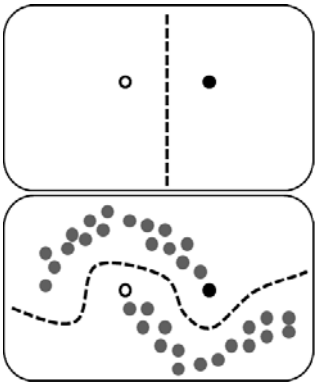


Fig. 1. Example of unlabeled data in semi-supervised learning (adapted from [3])

PROBLEM STATEMENT

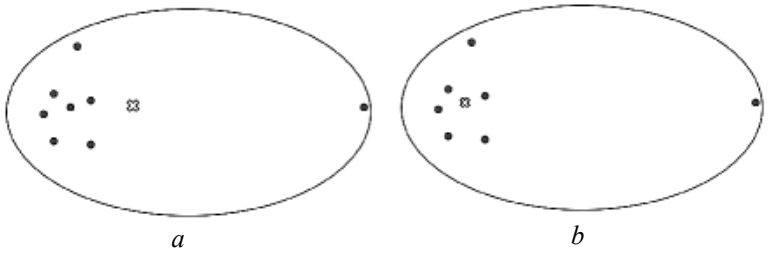
Given a set of l labeled examples $\{\langle x_1, y_1\rangle, \ldots, \langle x_l, y_l\rangle\}$, where x_i – feature vector of i-th example and y_i – its label (class), $y_1, y_2, \ldots, y_l \in Y$, and a set of u unlabeled data $\{x_{l+1}, \ldots, x_{l+u}\}$ $x_1, x_2, \ldots, x_{l+u} \in X$. The goal is to determine some function using given sets that will give correct mapping of points from X to Y: $f(x_j) = y_j$ for any point from X.

REVIEW OF LITERATURE

The semi-supervised learning approach described in the literature is not so widely investigated as unsupervised or supervised, especially algorithms implementation. In [2] presented an overview of semi-supervised approaches that describe assumptions of semi-supervised learning especially: smoothness, low-density, and manifold.

In particular, the semi-supervised approach demonstrates high efficiency in solving clustering problems. The idea of the corresponding improvement of clustering algorithm was described in the review [4]. Majority of these methods are modifications of the popular *k*-means clustering method. As the base method chosen for improvement within the semi-supervised paradigm, the unsupervised *k*-medoids approach also known as PAM (Partitioning Around Medoid) algorithm, proposed in [5]. A medoid is a point in the cluster, whose average dissimilarities with all the other points in the cluster is minimum. *k*-medoid is a partitioning technique of clustering, which clusters the data set of n objects into *k* clusters, with the number *k* of clusters assumed known *a priori.* Both the *k*-means and *k*-medoids algorithms are partitional, which breaks the dataset up into groups, and both attempt to minimize the distance between points labeled to be in a cluster, and a point designated as the center of that cluster. In contrast to the *k*-means algorithm, *k*-medoids choose data points as centers and can be used with arbitrary

distances, while in *k*-means the center of a cluster is the average between the points in the cluster (Fig. 2). Consequently, *k*-medoids are more robust to noise and outliers as compared to *k*-means.



a *b*

Fig. 2. Mean and medoid difference (adapted from [6]): *a* —mean; *b* — medoids

Another clustering method refined within the implementation of semi-supervised paradigm is DBSCAN – Density-Based Spatial Clustering of Applications with Noise proposed in [7]. The idea is to find core samples of high density and expand clusters from them. Such an approach is suitable for data that contains clusters of similar density. Based on a set of points, DBSCAN groups together points that are close to each other based on distance measurement, wherein it also marks as outliers the points that are in low-density regions.

A widespread clustering algorithm is also agglomerative clustering, which is the typical type of hierarchical clustering used to group objects in clusters based on their distance to each other. The algorithm starts by treating each object as a singleton cluster. Next, pairs of clusters are successively merged until all clusters have been merged into one big cluster containing all objects. The result is a tree-based representation of the objects – dendrogram (Fig. 3) [8].

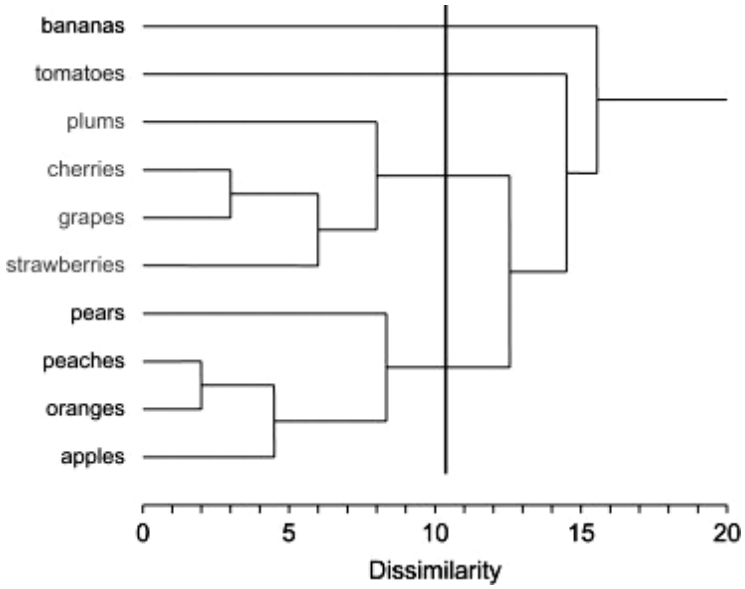


Fig. 3. Dendrogram of hierarchical clustering (adapted from [9])

The supervised approach for clustering problem is described in [10]. The nearest neighbor decision rule assigns to an unclassified sample point the classification of the nearest of a set of previously classified points. Thus, for any number

of categories, the probability of error of the nearest neighbor rule is bounded above by twice the Bayes probability of error. In this sense, it may be said that half the classification information in an infinite sample set is contained in the nearest neighbor.

MATERIALS AND METHODS

Distance function extension

As was shown above the major clustering methods form clusters based only on distance function. So we made the assumption that feature space can be extended by additional dimensions with information about available labels. We develop multiple distance functions that take to account that label dimension. The proposed approach allows concentrating attention on distance function creation and the use of already implemented and optimized clustering algorithms.

As a base distance metric, we use Euclidean distance. If there is additional data from the label space, it is advisable to use this information. An example is a naive solution - to reduce the distance between points if they have the same labels and increase in the opposite case:

$$D_{labeled}(p,q) = (1 + W * S(p,q)) * d(p,q), \quad (1)$$

where W — weight coefficient, $W \in [0,1]$; $S(p,q) = \{-1,$ if $label_p = label_q 1,$ if $label_p \neq label_q$ and $0,$ otherwise $d((p,q) =$ euclidean distance :

$$\sqrt{\sum_{k=1}^{n} (p_k - q_k)^2}\ .$$

In (1) weight coefficient W is used for tuning influence of labels: 0 — has no influence, ignoring label information; 1 — the distance between points with the same label equal to zero.

The suggestions concerning distance function not only decrease the distance between points with the same labels but also increase if points have different labels and improve robustness in cases with noised data and close clusters.

In real cases, data often occurs with labels as ordinal variables wherein the labels should be number type (0, 1, 2…). In this case, we can also use the distance between labels, because rank 1 is closer to rank 2 than rank 3 (for example, “cat” is closer to “dog” than to “fish”) [11]. However, it required additional data analysis before clustering.

Considering the idea above, one can expand the (1) with the distance between labesls:

$$D_{labeled}(p,q) = (1 + K * S(p,q) * * \left| label_p - label_q \right|) * d(p,q).$$

The methods described above are intuitively understood and easy to implement, but have one con:

– labeled and unlabeled data have the same influence on cluster formation, while the labeled point should have more influence;

– only points with labels are considered and do not take neighborhood points without labels, but in most cases, the neighborhood has the same class.

Semi-supervised *k*-medoids algorithm

We will propose some improved techniques that can resolve these issues and use the *k*-medoid approach as a base idea. However, unlike *k*-medoids the proposed algorithm first calculates medoids using only labeled data and next processes unlabeled classes – assign labels of nearest medoid. This approach is described by Algorithm 1.

This algorithm has the following pros:

– reduced processing time, because required only multiple iteration throw points unlike standard *k*-medoid;

– more robustness to wrong assigned labels, because the algorithm gives higher weights to labeled data in the medoids calculation step.

Algorithm 1. Modified *k*-medoids algorithm

Input:

X — feature matrix $n*m$, n — number of objects, m — number of features

y — labels vector of length n, $y[i] = -1$ if no label data for i-th object

Output:

y_predicted – vector of length n with object labels

1: $k \leftarrow$ number of clusters, e.g. number of unique labels in y
2: $X_l \leftarrow$labeled point from X
3: $X_u \leftarrow$unlabeled point from X
4: select k random points out of the X_l as the medoids
5: associate each data point to the closest medoid
6: while the cost of the configuration decreases:
7: for each medoid m, and for each non-medoid data point o from X_l:
8: Consider the swap of m and o, and compute the cost change
9: If the cost change is the current best, remember this m and o combination
10: associate each point from X_u with the nearest medoid
11: for each point o in X:
12: fill *y_predicted* with assigned medoid of point o
13: return *y_predicted*

Semi-supervised *k*-nearest neighbors algorithm

Another proposed approach uses the idea of *k*-nearest neighbors and the *k*-mean algorithm, because for classifying we use both information about the nearest points and classes centers of mass. As a distance metric was used Euclidean distance but any metric could be used.

Classes' centers do not recalculate after each assignment, because experiments show that it does not bring benefits but takes more computation time.

Algorithm 2 implements the proposed approach.

Algorithm 2. Object clustering using *k*-NN based approach

Input:

X – feature matrix $n*m$, n – number of objects, m – number of features

y – labels vector of length n, $y[i] = -1$ if no label data for i-th object

K – number of nearest points

C – the weight of the nearest class center

Output:

y_predicted – vector of length *n* with object labels

```
y_predicted ← empty list of length n
unlabeled_idxs ← list of indexes where y = -1
labeled_idxs ←list of indexes where y > -1
center_coordinates ← list of center coordinates for each class, calculated using available labels
random shuffle unlabeld_idxs
for i in unlabeld_idxs do
    distances_i ← distances from i-th object to each object with indexes in labeled_idxs
    argsort distances_i
    nearest_idxs ← indexes of first K elements from distances_i
    classes_dist_i ← distance from i-th object to each classes' center
    neares_class_idx ← index of nearest class to i-th object
    cls_counts ← list, where j-th element denote numbers of points belonging to j-th class among nearest_idxs
    cls_counts[neares_class_idx] ← cls_counts[neares_class_idx] + C // add additional value for class with nearest center
    label ← argmax(cls_counts)
    y_predicted[i] ← label
end for
for i in labeled_idxs do
    y_predicted[i] ← y[i]
end for
return y_predicted
```

So, the method described above allows:

– consider information about the nearest point, because in most cases point has the same label as its neighbors;

– combine a different kind of information;

– tune the weight of different sources using input parameters.

EXPERIMENTS

For experiments purpose was generated synthetic multiple datasets using sklearn library. Each dataset contains 250 points in 2D space. Available only 10% of labels as default. In addition, datasets have multiple clusters with different distributions and shapes (Fig. 4).

We will compare different approaches to find the average accuracy score on all these datasets for each approach with different combinations of base clustering methods and distance functions. Table included combinations that have improved compared to the base clustering method.

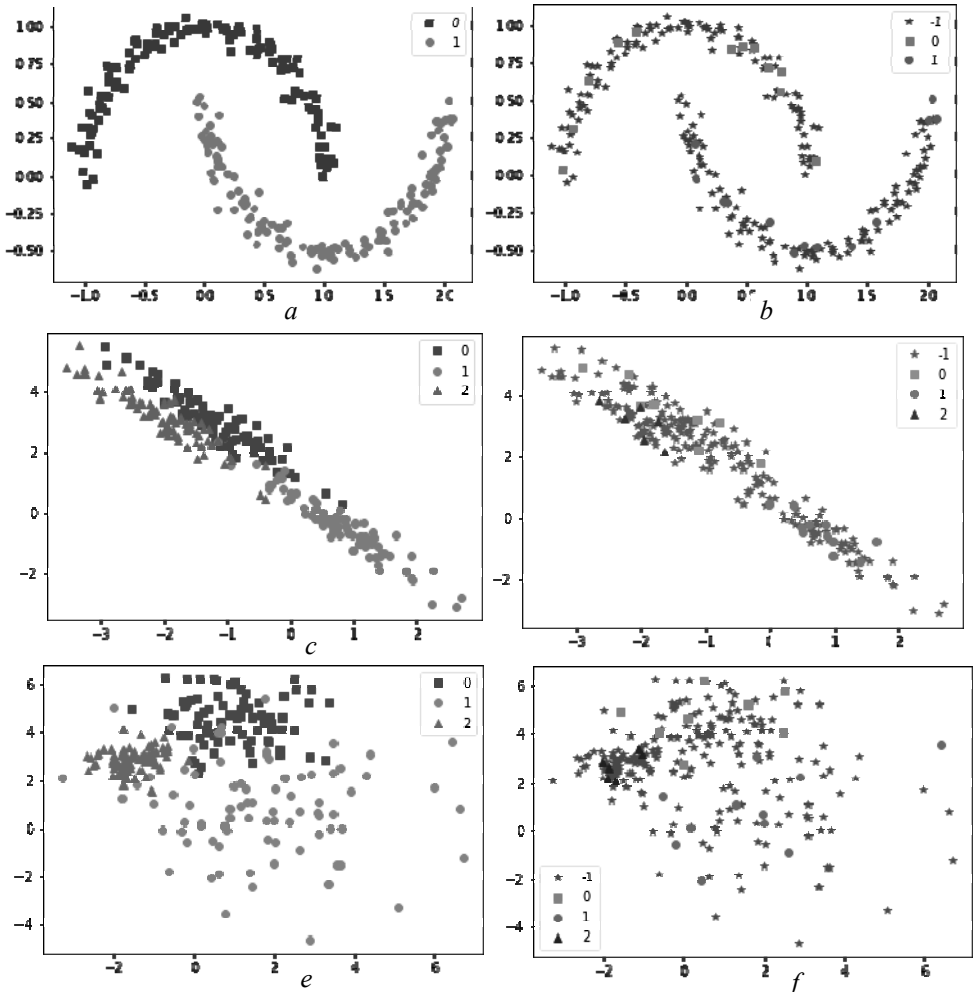


Fig. 4. Datasets visualization. The legend shows classes' labels, -1 – unlabeled point; *a, b* – Moons dataset, 2 classes, with non-convex and separable shapes; *c, d* – Aniso dataset, 3 classes, convex shape with same class variation, not separable; *e, f* – Varied dataset, 3 classes with a convex shape and different class variation, also not separable

Accuracy comparison

Method name	**Dataset name**			**Avg accuracy**
	Moons	Aniso	Varied	
Agglomerative + custom distance with ordinal variables ($W = 0.8$)	1.000	0.824	0.888	0.904
DBSCAN + custom distance ($W = 1.0$)	1.000	0.488	0.360	0.616
K-Medoids based	0.86	0.864	0.904	0.876
k-NN based ($N = 5$, $C = 2$)	0.904	0.900	0.912	0.905

The results shown in Table show that the best-unsupervised method is *k*-medoid and the *k*-NN based algorithm has higher average accuracy.

Fig. 5 illustrates the difference between unsupervised and semi-supervised methods, which is especially pronounced for non-convex data localization areas and for clusters with the same variation and located nearest to each other.

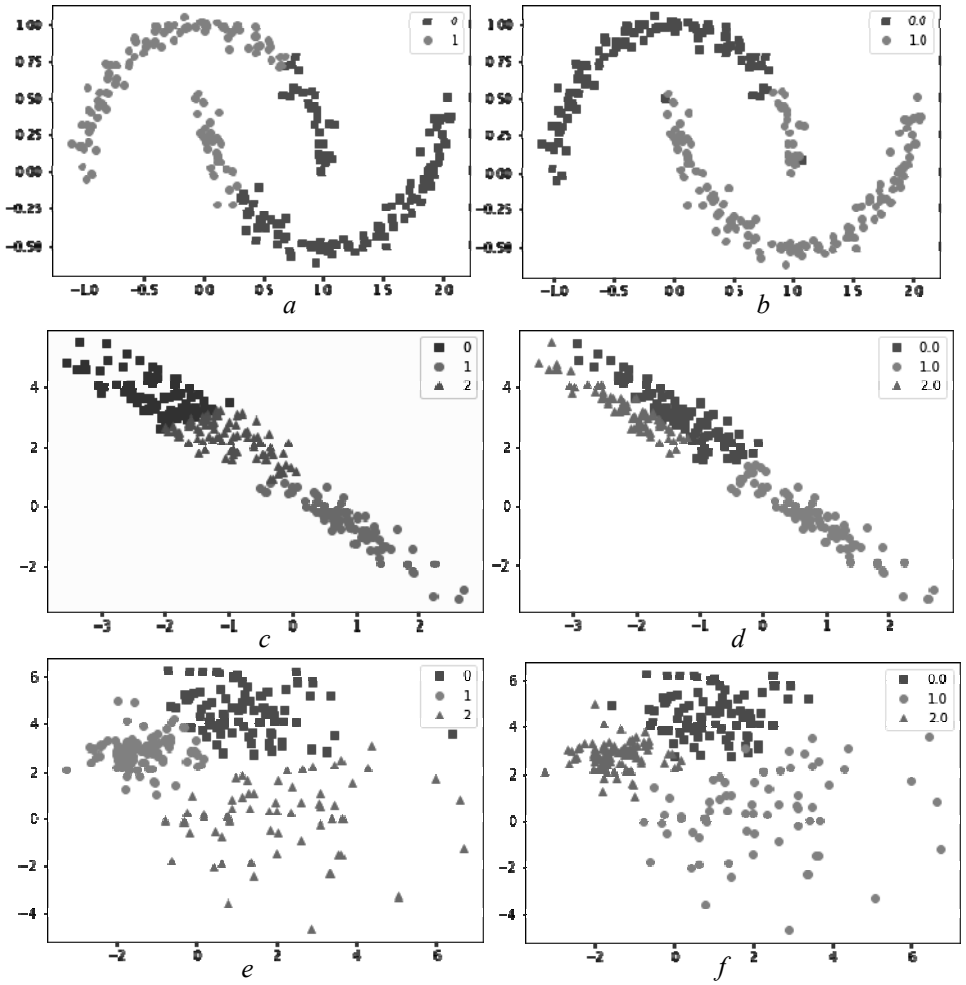


Fig. 5. Predicted labels visualization. *a*, *c*, *e* — unsupervised *k*-medoids, *b*, *d*, *f* — semi-supervised *k*-NN based method

Another required feature of a semi-supervised algorithm is quality versus a umber of labels dependency: more labels – higher quality and vice versa. However, Fig. 6 shows that clustering methods with custom distance functions do not have this feature. Therefore, this approach can be easy and fast, because it requires implementation only of the distance function. However, on the other hand, it is necessary to develop and tune the distance function for each case with a different number of available labels.

DISCUSSIONS

In Fig. 6 we can see that with the percentage of available labels increasing the accuracy of *k*-NN based and *k*-medoids based algorithms increased too. In addition, these algorithms have high accuracy according to Table. At that time, DBSCAN and Agglomerative methods did not respond to increasing labels. It means that we need to develop and tune the distance function for each case with a different number of available labels.

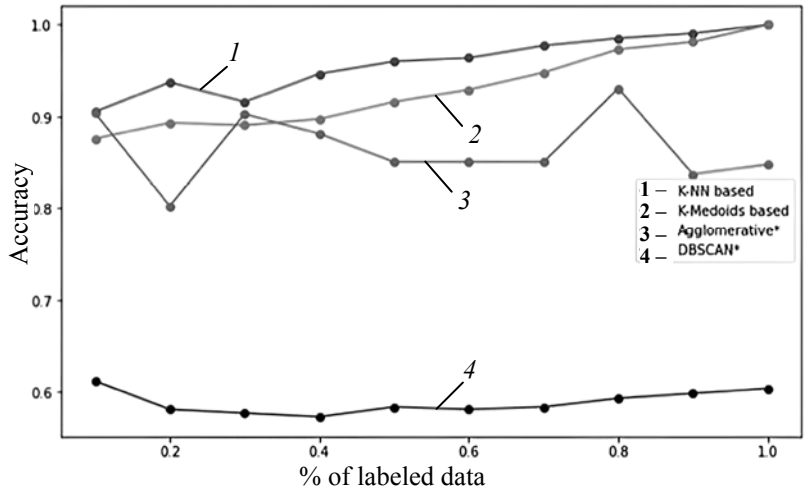


Fig. 6. Accuracy versus the quantity of labeled data comparison plot

CONCLUSIONS

In this study, we had shown that even small amounts of labeled data allow the use of semi-supervised learning and improve accuracy. At that, semi-supervised learning can improve algorithm performance too. Multiple approaches to semi-supervised learning were proposed, one of them is using a distance metric that considers available label information.

Further development of this work was a modification of other methods of classification and clustering and a deeper study of the influence of the distance function on the accuracy of clustering.

REFERENCES

1. L. Lyubchyk, A. Galuza, and G. Grinberg, "Semi-supervised Learning to Rank with Nonlinear Preference Model," *Recent Developments in Fuzzy Logic and Fuzzy Sets Studies in Fuzziness and Soft Computing*, pp. 81–103, 2020.
2. J.E.V. Engelen and H.H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2019.
3. Wikipedia contributors, "Semi-supervised learning", in *Wikipedia, The Free Encyclopedia*. [Online]. Available: https://en.wikipedia.org/wiki/Semi-supervised_ learning
4. E. Bair, "Semi-supervised clustering methods," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 5, no. 5, pp. 349–361, 2013.
5. A.S. Hadi, L. Kaufman, and P.J. Rousseeuw, "Finding Groups in Data: An Introduction to Cluster Analysis," *Technometrics*, vol. 34, no. 1, pp. 111, 1992.
6. X. Jin and J. Han, "*K*-Medoids Clustering," in *Encyclopedia of Machine Learning*. Boston, MA: Springer, 2011
7. M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second InternationalConference on Knowledge Discovery and Data Mining (KDD-96)*, *Portland, Oregon, USA*, E. Simoudis, J. Han, and U.M. Fayyad, Eds. AAAI Press, 1996, pp. 226–231. [Online]. Available: http://www.aaai.org/Library/KDD/1996/kdd96-037.php
8. Daniel Müllner, *Modern hierarchical, agglomerative clustering algorithms*. [Online]. Available: https://arxiv.org/pdf/1109.2378.pdf

9. T. Tullis and A. Bill, *Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics*. Elsevier/Morgan Kaufmann, 2013
10. T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.
11. L. Lyubchyk, G. Grinberg, and K. Yamkovyi, "Integral Indicator for Complex System Building Based on Semi-Supervised Learning," *2018 IEEE First International Conference on System Analysis & Intelligent Computing (SAIC)*, 2018.

Received 09.12.2021

INFORMATION ON THE ARTICLE

Leonid M. Lyubchyk, National Technical University «Kharkiv Polytechnic Institute», Ukraine, e-mail: Leonid.Liubchyk@kpi.edu.ua

Klym S. Yamkovyi, National Technical University «Kharkiv Polytechnic Institute», Ukraine, e-mail: klym.yamkovyi@cs.khpi.edu.ua

ПОРІВНЯЛЬНИЙ АНАЛІЗ МОДИФІКОВАНИХ АЛГОРИТМІВ НАВЧАННЯ З ЧАСТКОВИМ ЗАЛУЧЕННЯМ УЧИТЕЛЯ НА МАЛІЙ КІЛЬКОСТІ РОЗМІЧЕНИХ ДАНИХ / Л.М. Любчик, К.С. Ямковий

Анотація. Присвячено вдосконаленню методів кластеризації з частковим підкріпленням, а також порівнянню їх точності та стійкості. Запропонований підхід заснований на розширенні алгоритмів кластеризації шляхом використання доступного набору міток класів за допомогою заміни функції відстані, при цьому за використання запропонованої функції відстані враховуються не тільки просторові дані, але й мітки. Більше того, запропонована функція відстані може бути адаптована для роботи з порядковими змінними як мітки. Також запропоновано підхід, заснований на методі навчання без вчителя *k*-медоїдів, модифікований для використання лише розмічених даних на етапі обчислення медоїдів кластерів, комбінацію методу навчання з учителем *k* найближчих сусідів та без вчителя – *k*-середніх. При цьому алгоритм навчання використовує інформацію як про найближчі точки, так і про центри мас класів. Отримані результати демонструють, що навіть невеликий обсяг помічених даних дає змогу використовувати навчання з частковим підкріпленням, а запропоновані модифікації забезпечують підвищення точності і стійкості алгоритму, що продемонстровано під час експериментів.

Ключові слова: центр мас, кластеризація, функція відстані, найближчий сусід, навчання з частковим залученням вчителя, медоід.