Optimization of Algorithms for Selecting Urgent Factors That Affect the Cost of Legal Real Estate Lease

Nataliya Boyko[a], Oksana Yurynets[a], Iryna Shevchuk[b], Yurii Kryvenchuk[a], Tetiana Helzhynska[a]

[a] *Lviv Polytechnic National University, Lviv79013, Ukraine*
[b] *Ivan Franko National University of Lviv, Lviv 79007, Ukraine*

Abstract

This paper describes the main methods of selection of features, as well as examples of some of them. The results of some of the algorithms for the selection of features for the problem of selecting factors that affect the cost of renting US real estate are shown. When solving problems of data mining, in various application areas often have to operate with large data samples. This leads to a significant amount of time for data processing, and also requires a significant amount of RAM and disk memory of the PC. Therefore, the urgent task is to reduce the dimensionality of data samples. The paper considers algorithms for reducing informative features that determine the cost of real estate rental.

Keywords 1

Algorithm, machine learning, information, noise, forecast, classification tasks, feature, synthesis, model, problem, reduction.

1. Introduction

When solving problems of data mining [1], in various application areas often have to operate with large data samples. This entails significant time for data processing, and also requires a significant amount of RAM and disk memory of the PC. Therefore, the urgent task is to reduce the dimensionality of data samples [1-5]. Reduction occurs in various fields of life: biology, economics, computer science, mathematics, chemistry, linguistics, music, philosophy and others.

At the stages of machine learning and data generation, it is not always clear which features are important for building an optimal algorithm, so the data often contains a lot of extra (noise) information. The appearance of such features degrades the quality of the algorithm and increases the time. Therefore, in most cases, before solving the problem of classification, regression or forecasting, it is necessary to choose those features that are most informative. Choosing the right features can be more important than reducing data processing time or improving classification accuracy. For example, in medicine, finding the minimum set of features that is optimal for a classification task may be useful for developing a diagnostic test [3, 19].

Selection of important features (for example, selection of genes corresponding to a particular type of cancer) can help decipher the mechanisms underlying the problem of interest.

Each mathematical model is based on certain features [5, 20].

A feature is a peculiarity of an object or phenomenon that determines the similarity of its carrier to other objects of knowledge or difference from them, a property. The set of features (which can be

IT&AS'2021: Symposium on Information Technologies & Applied Sciences, March 5, 2021, Bratislava, Slovakia
EMAIL: nataliya.i.boyko@lpnu.ua (N. Boyko); uroksana@gmail.com (O. Yurynets); iryna.shevchuk@lnu.edu.ua (I. Shevchuk); yurkokryvenchuk@gmail.com (Yu. Kryvenchuk); tanyazvarich@ukr.net (T. Helzhynska)
ORCID: 0000-0002-6962-9363 (N. Boyko); 0000-0001-9698-2499 (O. Yurynets); 0000-0003-4386-3730 (I. Shevchuk); 0000-0002-2504-5833 (Yu. Kryvenchuk); 0000-0003-3280-5199 (T. Helzhynska)

reduced to one feature) allows to distinguish the subject (phenomenon) from other objects (phenomena) [3, 4].

For the synthesis of models that determine the input data of neural networks, an input training sample is used, which consists of a large set of features that characterize the object under study. Large data sets, as a rule, contain redundant and uninformative features that complicate not only the process of model synthesis, but also lead to its redundancy, which increases the time for classification according to such a model [2]. Thus, in solving the problems of pattern recognition, an important step is the process of reducing the input set of features. The difficulty of solving the problem of choosing the most significant combination of features lies in its combinatorial nature. The use of a complete search of all possible combinations with a large number of features leads to high resource costs. Therefore, this approach is not used in practice, and you use its optimization, which will be discussed in the following sections.

2. Materials and methods

The method of selection can be implemented by a complete search of features, thus checking all possible sets, select those features on which the error is minimal [6, 18].

In order to understand how this method works, we will describe its iterations in more detail.

Step 1. Generate all possible sets of features.

Step 2. Evaluate all the combinations obtained in the previous step, calculating for each of them the value of the criterion for evaluating a set of features: $J(X_e)$.

Step 3 Step 3. Determine the optimal value of the criterion for evaluating the set of features: $Jopt = \min(J(X_e))$

Step 4. Determine the optimal set of features: $X^* = \arg\min\ J(X_e)$.

Step 5. Stop.

The classic full search is the easiest to implement and guarantees an optimal solution. The disadvantage of this method is its limitation in solving practical problems, due to the large computational costs of its use.

Unfortunately, despite its simplicity in implementation and guarantee of finding the optimal solution, this method has complexity $O(k^n * x)$ operations, where

k – is the number of possible states of the feature;

n – is the total number of states;

x – is the complexity of the result determination operation for a certain set of features.

We prove that this is true: each of the features can be included in the sequence having one of the k states, so the number of sequences is equal k^n and for each of the sequences found it is necessary to check its optimality [8, 9, 17].

In general, there are three main classes of feature selection algorithms - filters, wrappers and built-in algorithms.

Filters are built on the basis of certain indicators that do not depend on the method of classification. For example, such as the correlation of features with the target vector, the criteria of informativeness. They are used in classification task. One of the advantages of this class is that it can be used as a kind of premature processing to reduce dimensionality and prevent retraining. Filtration methods usually work quite quickly. Filters are used to select features in clustering, to construct an initial approximation. Unfortunately, such methods cannot find a strong relationship between the symptoms, and are usually not sensitive enough to detect all the dependencies in the data.

3. Built-in algorithms

Built-in algorithms perform feature selection during the classifier training procedure, and they explicitly optimize the feature set used for better accuracy. The advantages of built-in algorithms are that they usually find solutions faster, avoiding retraining data from scratch, while eliminating the need to divide the data into training and test samples. However, these algorithms are not universal [8, 10].

Wrapping methods rely on information about the importance of the characteristics obtained from certain classification or regression methods, and therefore can find deeper patterns in the data than filters. Wrappers can use any classifier that determines the importance of the feature. Several wrapper algorithms will be discussed in more detail later in this paper [10, 13].

RandomForest is a machine learning algorithm proposed by LeoBreimani and AdeleCutler. Represents an ensemble of many numerical, sample-sensitive algorithms (decision trees). These algorithms have a small offset. The bias of the learning method is the deviation of the average value of the answer of the learned algorithm from the answer of the ideal algorithm. Each of these classifiers is based on a random subset of objects and a random subset of features. Let the training sample consist of N examples, the dimension of the feature space is M, and the given parameter m. Let's write down the RandomForest algorithm step by step.

All trees of the ensemble are built independently of each other using the following procedure:

1. Generate a random subsample with the repetition of the size n from the training sample.
2. Construct a decision tree that classifies the examples of this subsample. In addition, during the creation of a certain node of the tree, we will choose the feature on the basis of which the partition takes place, not from all Moznaks, but only from m randomly selected.
3. The tree is built until the subsample is completely exhausted.

Objects are classified by voting: each tree in the ensemble assigns an object to one of the classes, and the class for which the largest number of trees voted wins.

The RandomForest algorithm can be used to assess the importance of features. To do this, you need to teach the algorithm on the sample and when building a model for each element to calculate the out-of-bag error [14]. Let X be a bootstrapped tree sample b. Bootstrapping is the selection of a specific set of objects from a sample, with the result that some objects are selected several times and several times not. Adding multiple copies of one object to the bootstrapped sample is responsible for setting the weight for this object - the corresponding value will be included in the functionality several times, and therefore the penalty for error on it will be greater. Let *L(x, y)* be a loss function, y_i – be the answer to the *i*-th object of the training choice, then the out-of-bag error is calculated on the basis of the following formula.

$$OOB = L\left(\sum_{i=1}^{l} y_i \frac{1}{\sum_{n=1}^{N}\left[x_i \notin x_n^l\right]} \sum_{n=1}^{N}\left[x_i \notin X_n^l\right] b_n(x_i)\right) \qquad (1)$$

Then, for each object, this error is averaged over the entire random forest. To assess the importance of a feature, its values are shuffled for all objects in the training sample and the out-of-bag error is calculated again. The importance of the feature is calculated by averaging over all trees the difference of out-of-bag-error before and after mixing the value. The value of such errors is normalized to the mean deviation. Random forest has some other advantages for using it as an algorithm for selecting features: it has very few customizable parameters, relatively fast and efficient operation, which allows you to find the informativeness of the samples without significant estimated costs. Boruta is a heuristic algorithm for selecting important features, based on the use of RandomForest [5, 11]. The essence of the algorithm is that at each iteration the features in which the Z-measure is less than the maximum Z-measure among the added features are removed. To obtain the Z-measure of a feature, you need to calculate the importance of the feature obtained using the RandomForest algorithm and divide it by the standard deviation of the importance of the feature. Added features are obtained as follows: copy the features that are in the sample, and then fill in each new feature by shuffling its feature. In order to obtain statistically significant results, this procedure is repeated several times, and variables are generated independently at each iteration [12, 14].

Let's write down Boruta algorithm step by step:

- Add copies of all features to the data. In the future, copies will be called hidden features.
- Randomly mix each hidden feature.
- Run RandomForest to get the Z-measure of all features.
- Find the maximum Z-measure of all Z-measures for hidden features.
- Delete the features in which the Z-measure is less than found in the last step.
- Delete all hidden features.

- Repeat all steps until the Z-measure of all features becomes greater than the maximum Z-measure of hidden features.

ACE (ArtificialContrastswithEnsembles) [6] is another algorithm that can be used to select features. The main idea of the ACE algorithm is similar to the idea of Boruta - each feature is filled in randomly, by re-shuffling its values. RandomForest is run on the received sample. However, unlike Boruta, the least important features are not removed, which allows you to increase some measurements of important features. The most important features are found by the ACE algorithm, the reverse is removed, which allows the algorithm to find deeper patterns. Deleted features ACE returns as a response.

4. Methods of solving

An article [1] considered an example of the Boruta algorithm for the problem of finding the difference between two subtypes of leukemia. This problem was previously solved by other methods [7], which allowed to compare the results after using Boruta. The data have information about 38 patients. 3051 genes were described for each of them. To assess the quality of the algorithm, another 1,000 semi-synthetic features were added, which were generated by shuffling 1,000 randomly selected genes. A good algorithm will not select the generated features as important. The importance of the gene was assessed using the Boruta algorithm based on RandomForest. The number of trees in RandomForest varied from 500 to 100,000. Each run was repeated 15 times.

We introduce the following notation:

- Dud2002 - 91 isolated gene in 2002, the solution is described in [9]
- Gol1999 - 50 isolated genes in 1999, the solution is described in [7]
- Dram2008 - 30 selected genes in 2008, the solution is described in [8]
- Bor500 - 82 selected genes using Borutan-based RandomForest with 500 trees.

Bor100k - 261 isolated genes using Boruta-based RandomForest with 100,000 trees. Figure 1 shows a graph of the number of genes on the number of trees. The Boruta algorithm is highlighted in black dots, and the results of the algorithms Dud2002, Gol1999, Dram2008 are shown in other colors, which were fed to the input of the features selected by Boruta with different number of trees.

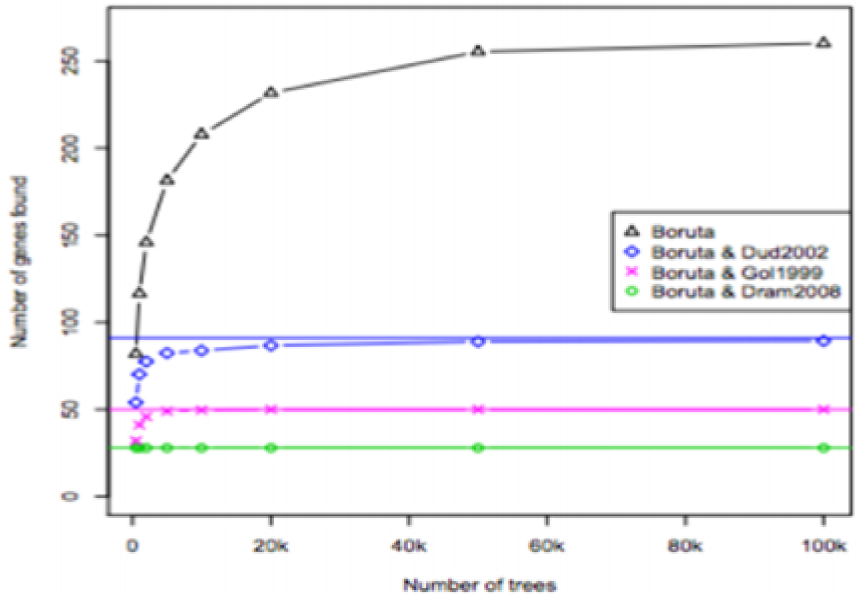

Figure 1: The dependence of the number of selected genes on the number of trees

It is seen that as the number of trees increases, the number of selected genes increases. Solid horizontal lines show the total number of features selected by this algorithm. As a result, the algorithm never selected as important genes the semi-synthetic data added for verification. Based on this, the authors of this experiment concluded that the Boruta algorithm effectively copes with the task of selecting features. To understand how features selected by the Boruta algorithm correlate with previously selected features, and how the set of previously selected features correlates with each other, a graphical representation of the sets of selected genes is given. Figura 2 shows how the selected sets of important genes Dud2002, Gol1999, Dram2008, Bor500, Bor100k intersect. The area covered with blue, purple and green stripes is represented by the intersections of Bor100k with Dud2002, Gol1999 and Dram2008 datasets, respectively.

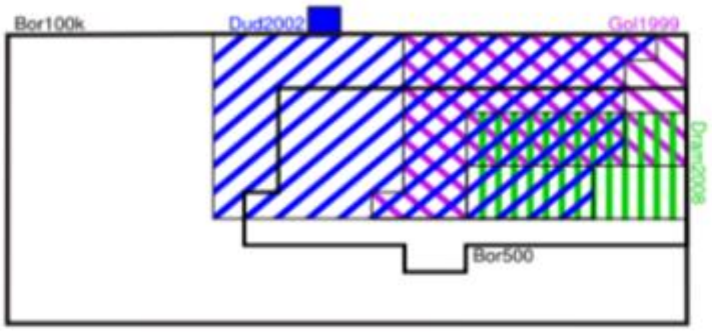

Figure 2: Intersection of important genes in different algorithms

It can be seen that many of the genes isolated by Boruta with 500 trees intersect with Dud2002, Gol1999 and Dram2008. The genes obtained in the methods of Dud2002, Gol1999 and Dram2008 are weakly correlated with each other (their retinal area is quite small). Genes isolated from 100,000 trees include all genes isolated from the rest of the algorithms. This experiment confirms that all the previously identified features are really important. Moreover, the algorithm has identified 150 new genes. The result of this experiment shows that the sensitivity of the Boruta algorithm depends on the number of trees in RandomForest. This is due to the Z-measure parameter. It is evaluated using important features derived from the built-in algorithm in RandomForest, and the importance of the features is the average decrease in the accuracy of the trees that use this feature. Therefore, the relevance of the gene can be learned only with a fairly large number of trees.

5. Experiments

Data on 2227 different types of real estate will be used to study the operation of feature selection algorithms.

On the basis of the received data it is necessary to predict cost of rent and to define features on which the cost strongly depends.

The data have 19 features, among which 2 are categorical:

- SpaceType - The type of building in which the object is located, has 31 values;
- LeaseType - Provides information about what costs are included in the rent.

9 integers:

- SpaceSize - Room size;
- Numberoftransportspots - Number of parking spaces;
- Population - Population in the region;
- Landarea - Area of the region;
- Socialcharscore - Prestige (social score);
- AverageHHincome 2013 - Average income of the population in 2013;
- Average salary of employees ($ 000s) - The average income of a worker in a new business;
- Number of new retail places 2013 - 2010 - Number of new places for rent for 2010-2013.

8 rational:

- Populationchange 2013-2010 - Change in population as a percentage;
- Density of people living area - Density of people in the region;
- Density of people working in area (based on lat / lon) - Density of working people;

- Total density (living + working) - The sum of two past features;
- Household size - The size of the warehouse;
- Income change 2013 - 2010 - Income change for 2010 - 2013;
- Changein%ofbachelordegrees - Change in the percentage of people with a Bachelor's degree.
- % of employees in new companies vs all - Percentage of employees in new companies.

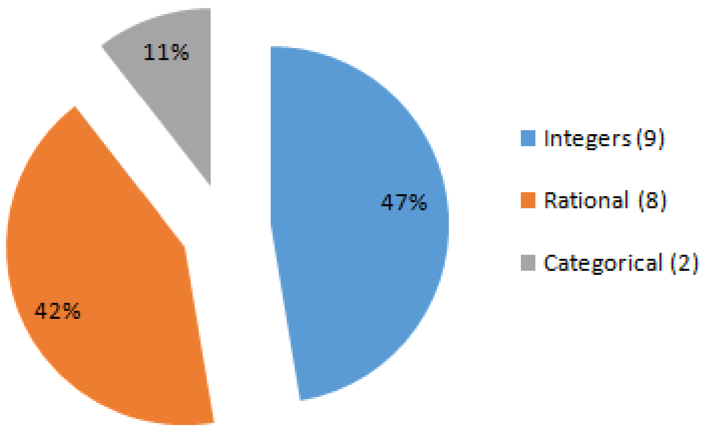

Figure 3: Distribution of variables in the Dataset

6. Assessing the importance of features

The initial sample was divided into three parts, to use cross-validation with three folds. Initially, categorical features were coded by a set of Boolean vectors, one for each value of the feature. Figura 4 shows a graph in which for each feature together with a set of Boolean vectors showed their importance for the three parts of the sample using the built-in RandomForest algorithm.

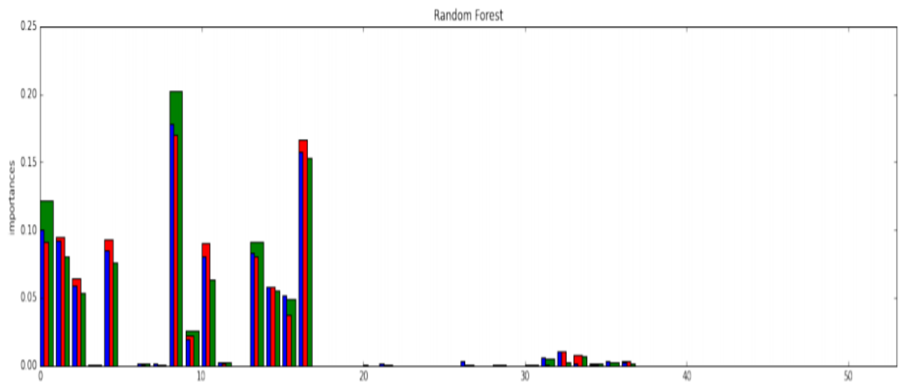

Figure 4: The importance of features (added set of Boolean vectors - the last 37 features)

The importance of adding binary features to almost all is zero. And after many experiments, it turned out that the set of Boolean vectors for categorical features, only worsened some of all the algorithms used.

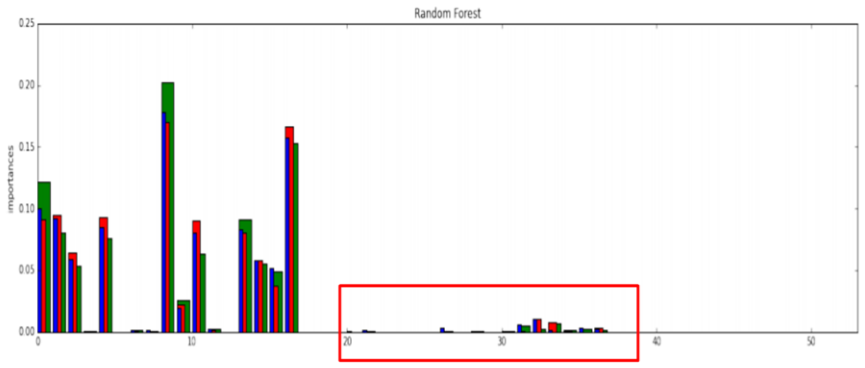

Figure 5: The importance of categorical features

Therefore, the features responsible for the categorical were removed from the dataset. Figure 6 shows a graph showing the importance of each of the 17 non-categorical features for the three parts of the sample using the built-in RandomForest algorithm.

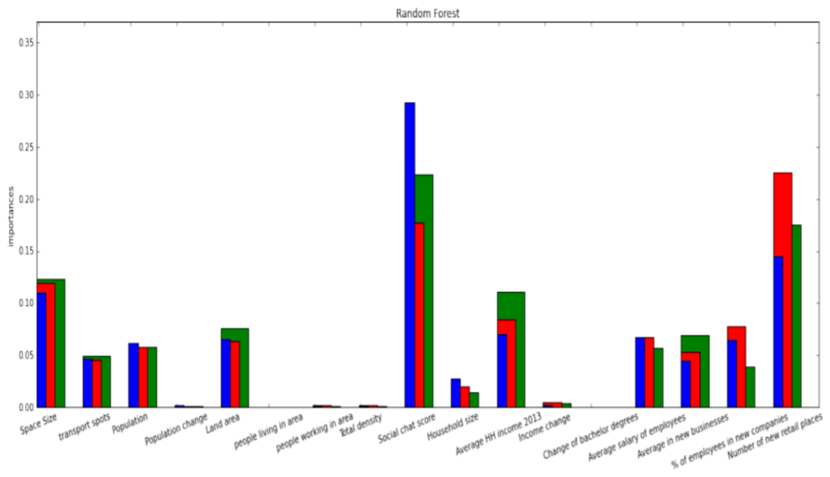

Figure 6: Importance of features after removal of categorical

The graph in different colors shows the importance of the feature obtained on three samples. Let`s find the average value for the three tests to reflect the importance of each feature.

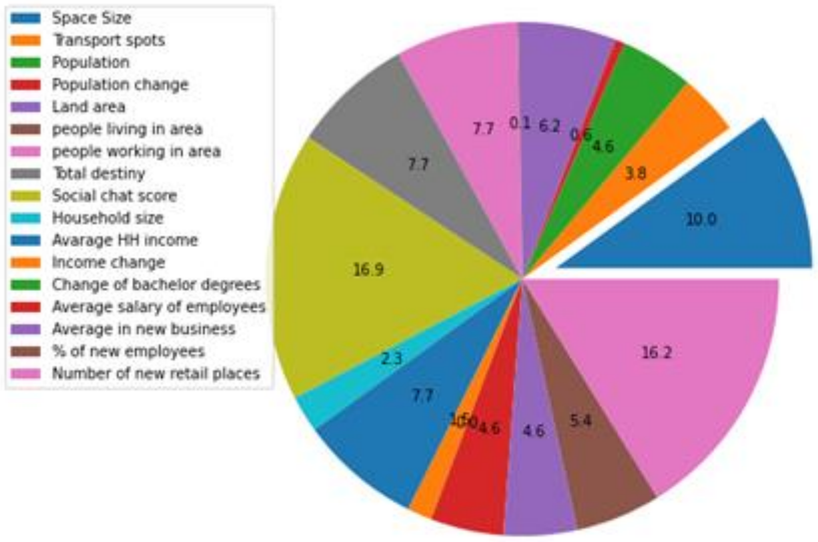

Figure 7: The average importance of non-categorical features

Table 1

The following features have the maximum importance as a result

Title	Importance	Title
Social chat score	16.9%	Social chat score
Number of new retail places	16.2%	Number of new retail places
Space Size	10%	Space Size
People working in area	7.7%	People working in area
Total density	7.7%	Total density
Average HH income	7.7%	Average HH income

The remaining features were less than 5% important (this is much less important than the selected features). Using only these features, which were of maximum importance, a simple algorithm was built, the average error of which differs little from RandomForest. This algorithm sorts the value of each important feature in the object and, depending on the range in which the features lie, assigns a certain constant to the current variable.

The parameters were adjusted on two-thirds of the randomly selected sample objects and tested on the remaining one-third. After many experiments, it turned out that RandomForest is the best machine learning algorithm for solving the problem of predicting the cost of rent, among the revised algorithms. Figura 8 shows the results of those algorithms that showed the best result.

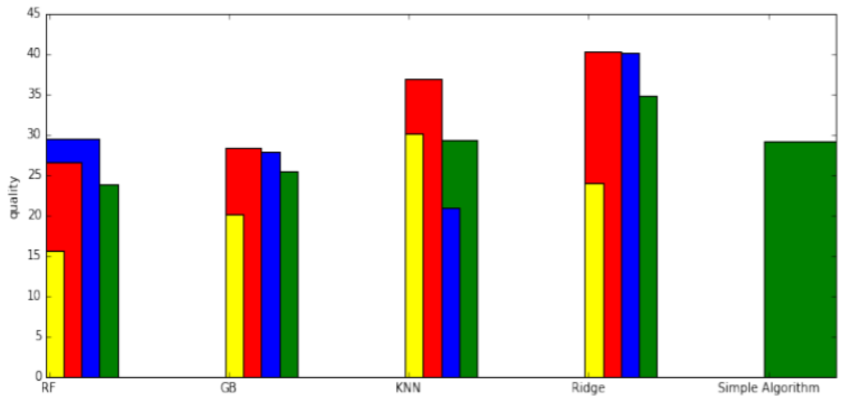

Figure 8: Results of the best algorithms

The graph shows the algorithm error on the MSE metric (MeanSquaredError) for each of the three folds and the average error on all three. Each algorithm was applied first to the entire sample and then only to the features that were selected as important. The table shows the average error of the algorithms for the MSE metric.

Table 2
The average error of the algorithms for the MSE metric

Algorithm	Quality on all features	Quality on important features
Random Forest	24.1	23.8
GB	25.4	25.7
KNN	30.2	28.93
Ridge	35.0	34.7

Almost all used models showed the best result on the selected features, but for this purpose it was necessary to reconfigure all parameters.

Implementations of all machine learning algorithms were used from the scikit-learn library in the Python programming language. For each algorithm, all parameters were configured using the GridSearchCV function, but the average error for the three folds on the configured parameters was almost the same as the error on the standard parameters. All algorithms were run on selected features using the described feature selection algorithms, but the average error for the three folds remained almost the same.

The Boruta algorithm was applied to the data. Figura 9 shows the importance of the features determined by the Boruta algorithm. Important features are identified in green. The importance of the features was determined on 200 iterations of the algorithm.

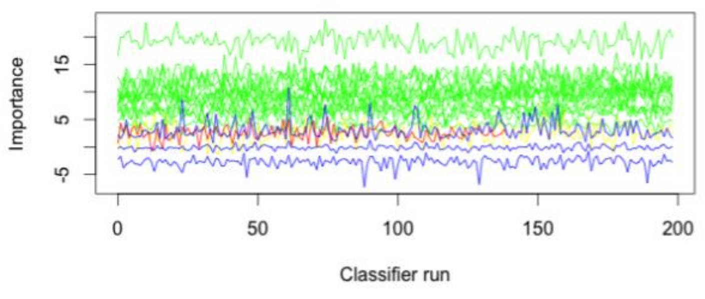

Figure 9: The importance of features with Boruta

Here are the most significant features selected with Boruta.

Table 3
The most significant features selected with Boruta

Feature	The average significance of the feature
Space Size	18.6
Numberofnewretailplaces 2013-2010	12.8
Socialchatscore	12.3
Densityofpeopleworkinginareabasedonlatlon	12.2
Totaldensitylivingworking.	11.3
Average HH income 2013	9.4
Householdsize	8.7
Population	8.2

It turned out that this algorithm selected almost all the important features that were selected using the built-in algorithm.

Let's compare the results of the Boruta algorithm with nested RandomForest.

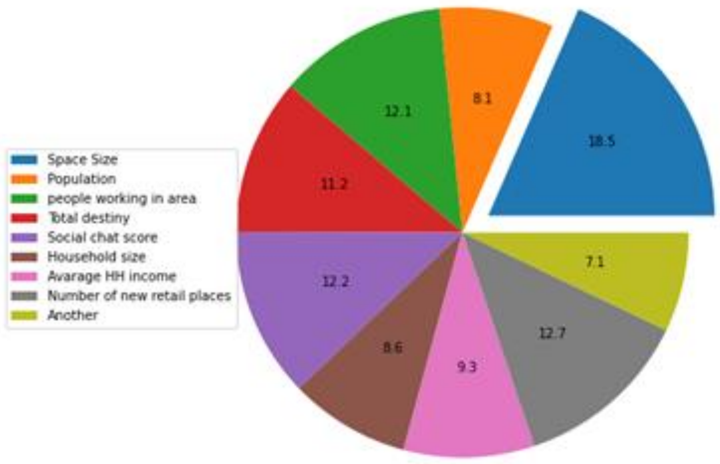

Figure 10: The most important features are determined by the Boruta algorithm

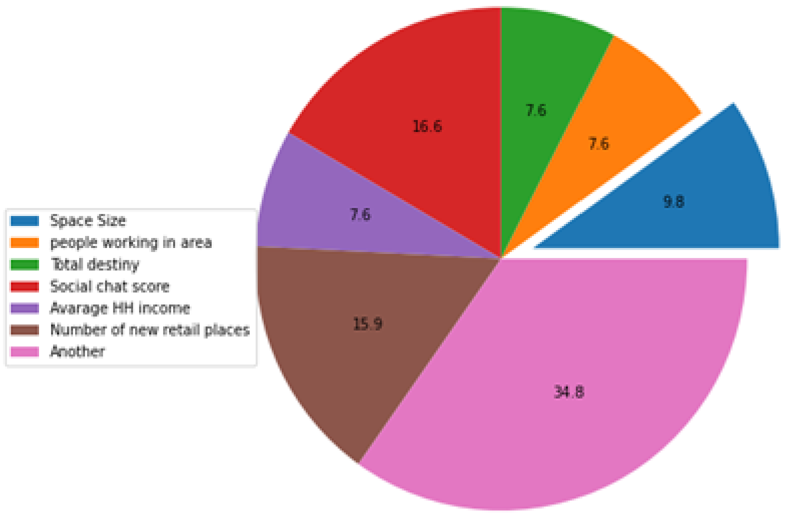

Figure 11: The most important features are determined by the Random Forest algorithm

Comparing these results, we can see that the Boruta algorithm shows the results, which is an extension of the results of RandomForest. Several things lead us to this conclusion:

The Boruta algorithm has retained all the important features found by its predecessor.

The Boruta algorithm has retained the proportionality of the importance of the selected features.

The Boruta algorithm found deeper connections between the features and added 2 new fetures to our result, and reduced the% content of unimportant features from 35% to 7%.

7. Comparison of the complexity of complete search and Boruta algorithm

Define n as the number of features in the data network, p is the number of iterations of the Boruta algorithm, and k is the number of forests in RandomForest.

Then the complexity of the full search will be evaluated as $O(2^\wedge n)$, and the complexity of the Boruta algorithm as $O(p * n)$. owever, the Boruta algorithm requires $O(n * k)$ of memory to store arbitrary trees generated by RandomForest.

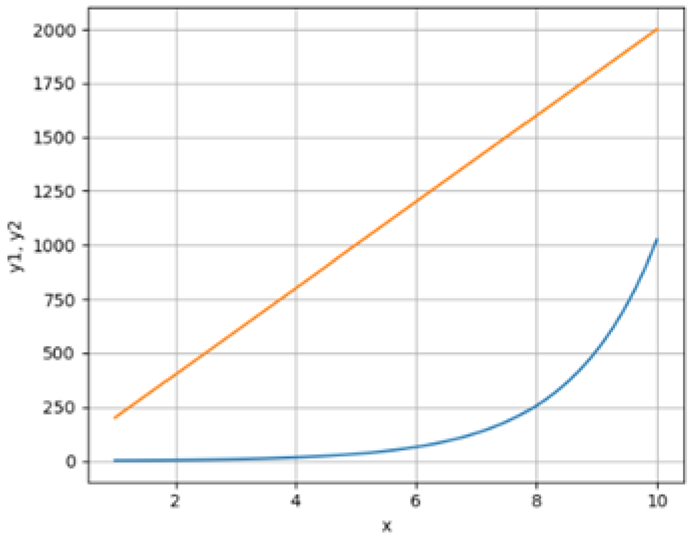

Figure 12: Comparison of complexity for n = 10

For n = 10, we can see that the number of operations for a complete search is 2 times less than for our chosen algorithm. However, an important fact is that the complexity of Boruta grows linearly depending on the number of selected iterations and the number of features in our data network. Instead, the full bust grows exponentially from n, and already for n = 11, catches up with Boruta, and for n = 15, leaves it far behind.

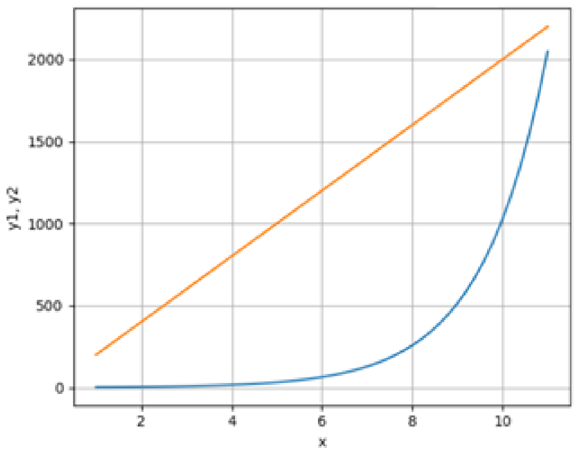

Figure 13: Comparison of the number of operations for *n* = 11

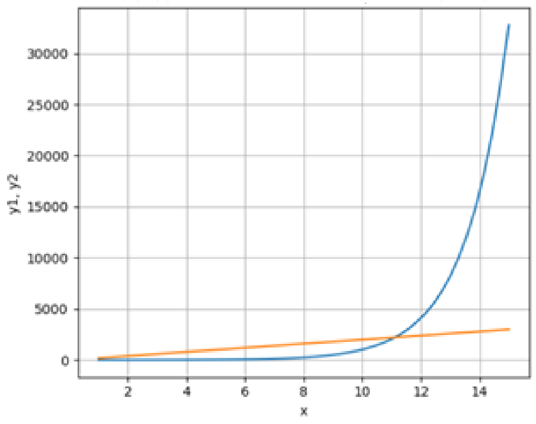

Figure 14: Comparison of the number of operations for *n* = 15

Based on these graphs, we can say that for small data you can use a full search to obtain the optimal set of features, but for large, there is a so-called "combinatorial explosion", which puts the Boruta algorithm in 1st place.

8. Conclusion

Selection of features is an important stage in the construction of machine learning algorithms. This step is needed to get rid of redundant features and thus improve the quality and reduce the execution time of algorithms. The conducted experiments confirm that the algorithms for selecting features using RandomForest effectively cope with their task. Thus, the solution of this problem can significantly reduce the time spent on calculating the matrix chain. This solution can be obtained by a complete overhaul: it is necessary to consider all possible sequences of calculations and choose from them the one that occupies the smallest number of scalar products when calculating the chain. However, it should be borne in mind that this algorithm itself requires exponential calculation time [2, 15], so that for long matrix chains the gain from calculating the chain in the most efficient way (optimal strategy) can be completely lost during the stay of this strategy [4].

9. References

[1] M.F. Yakovlev, T.O. Gerasymova, A.N. Nesterenko, Characteristic feature of the solving both of non-linear systems and systems of ordinary differential equations on parallel computer, In Proceedings of international symposium “Optimization problems of computations” (OPC - XXXV). Glushkov Institute of cybernetics of NAS of Ukraine, 2009. Kyiv, Vol. 2. P. 435-439.

[2] M.F.Yakovlev, A.N. Nesterenko, V.N. Brusnikin, Problems of the efficient solving of non-linear systems on multi-processor MIMD-architecture computers. Mathematical machines and systems. (4), 2014. P. 12-17.

[3] A.N. Khymych, Y.N.Molchanov, A.V. Popov, Parallel algorithms for solving problems of computational mathematics. Kiev: Scientific Opinion, 2008, 248 pp.

[4] K. Autar, Nonlinear Equations - Newton-Raphson Method-More Examples, Civil Engineering. August 7, 2009, 4 pp

[5] K.K. Lai, J.W.M. Chan, Developing a simulated annealing algorithm for the cutting stock problem, Computers & Industrial Engineering, vol. 32, 1997, pp. 115-127.

[6] D. Kakhaner, K. Mouler, S. Nэsh, Numerical methods and software. Mir, 1998, 575 pp.

[7] M. Voss,: OpenMP Share Memory Parallel programming. Toronto, Kanada, 2003.

[8] B. Chapman, G. Jost, Ruud van der Pa, Using OpenMP: portable shared memory parallel programming (Scientific and Engineering Computation). Cambridge: The MIT Press, 2008.

[9] R. Chandra, R. Menon, L. Dagum, D. Kohr, D. Maydan, J. McDonald, Parallel Programming in OpenMP. Morgan Kaufinann Publishers, 2000.

[10] G. Ananth, G. Anshul, , G. Karypis, V. Kumar, Introduction to Parallel Computing» Addison Wesley, ISBN- 0-201-64865-2, 2003, 856 p.

[11] P. Larra Naga, C.M.H. Kuijpers, R.H.Murga, I. Inza, S. Dizdarevic, Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators,. Artificial Intelligence Review, Kluwer Academic Publishers. Netherlands, Vol. 13(2), 1999, pp. 129–170.

[12] J. Majumdar, A.K. Bhunia, Genetic algorithm for asymmetric traveling salesman problem with imprecise travel times. Journal of Computational and Applied Mathematics, Elsevier, Vol. 235, Issue 9, 2011, pp. 3063-3078.

[13] O. E. Semenkina, E. A. Popov, O. E. Semenkina, Self-configuring evolutionary algorithms for travelling salesman problem. Journal of Siberian State Aerospace University named after academician M. F. Reshetnev , Vol. 4(50), 2013, pp. 134-139.

[14] M. Mitchell, An Introduction to Genetic Algorithms. USA, UK: MIT Press, 1999.

[15] N. Boyko, L. Mochurad, U. Parpan, O. Basystiuk, Usage of Machine-based Translation Methods for Analyzing Open Data in Legal Cases, International Workshop on Cyber Hygiene (CybHyg-2019) co-located with 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks (CyberConf 2019), Kyiv, 2019, pp. 328-338.

[16] N. Boyko, L. Mochurad, I. Andrusiak, Yu. Drevnytskyi, Organizational and Legal Aspects of Managing the Process of Recognition of Objects in the Image, Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019), Kyiv, Ukraine, November 30, 2019, pp. 571-592.

[17] G. Sokolov, Yu. Syerov, ”Building a Computer Model of an Acoustic Signal Recognition Device”. CEUR Workshop Proceedings. Vol-2654: Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019). Kyiv, Ukraine, November 30, 2019. pp. 471-491. http://ceur-ws.org/Vol-2654/paper37.pdf

[18] S. Fedushko, Yu. Syerov, O. Tesak, O. Onyshchuk, N. Melnykova, ”Advisory and Accounting Tool for Safe and Economically Optimal Choice of Online Self-Education Services”. Proceedings of the International Workshop on Conflict Management in Global Information Networks (CMiGIN 2019), Lviv, Ukraine, November 29, 2019. CEUR-WS.org, Vol-2588, 2020, pp. 290-300. http://ceur-ws.org/Vol-2588/paper24.pdf

[19] N. Boyko, L. Mochurad, O. Chervinska, Cost-Effective Use of Mathematical Methods for the Organization and Management of Data in the Cloud, Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019), Kyiv, Ukraine, November 30, 2019, pp. 444-457.

[20] A. Shabalov, E. Semenkin, P. Galushin, Automatized Design Application Of Intelligent Information Technologies for Data Mining Problems. Joint IEEE Conference, The 7th International Conference on Natural Computation & The 8th International Conference on Fuzzy Systems and Knowledge Discovery, Shanghai, China, 2011, pp. 2659–2662.

[21] E. Semenkin, M. Semenkina, Self configuring genetic algorithm with modified uniform crossover operator, Advances in Swarm Intelligence, Part 1, Springer, 2012, pp. 414.–421.

[22] X. Chen, P. Zhang, G. Du, F. Li, Ant colony optimization based memetic algorithm to solve bi-objective multiple traveling salesmen problem for multi-robot systems, 2018. DOI: 10.1109/ACCESS.2018.2828499