A report on

INFO-H423 - Data Mining. Assignment – part 1.

By

Raymond Lochner (000443637)

Aleksei Karetnikov (000455065)

Aldar Saranov (000435170)

Description

Attribute selection procedure is done in “attribute-selection.rmp” file. The dataset preprocessing, data mining and model evaluation stages are described in a single “combined.rmp” file.

Data Preprocessing

The input file has been successfully loaded and the appropriate roles have been assigned. In order to detect and remove attributes which do not contribute to the prediction, we measure the information gain for each attribute. We find that certain attributes have negligible information gain (load, housing and day_of_week).

attribute	weight
loan	0.000
housing	0.000
day_of_week	0.000
marital	0.002
campaign	0.003
education	0.003
default	0.008
age	0.012
job	0.014
contact	0.017
previous	0.022
cons.price.idx	0.026
cons.conf.idx	0.032
month	0.038
poutcome	0.044
pdays	0.044
emp.var.rate	0.057
euribor3m	0.068
nr.employed	0.076

Table 1. Information gain rate of all attributes

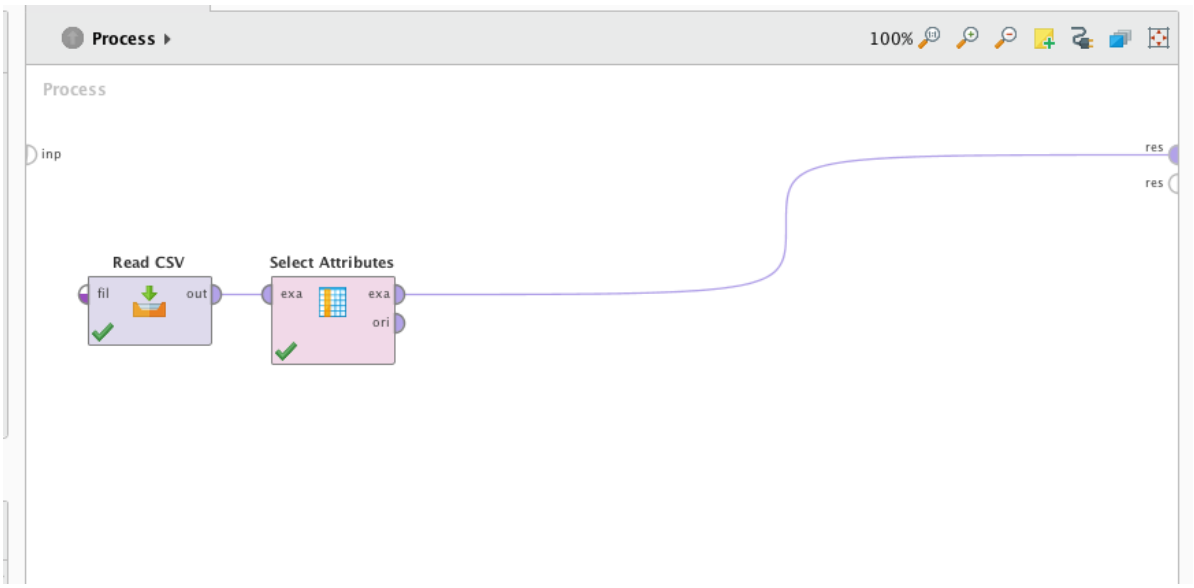

Figure 1. Attribute selection only stage

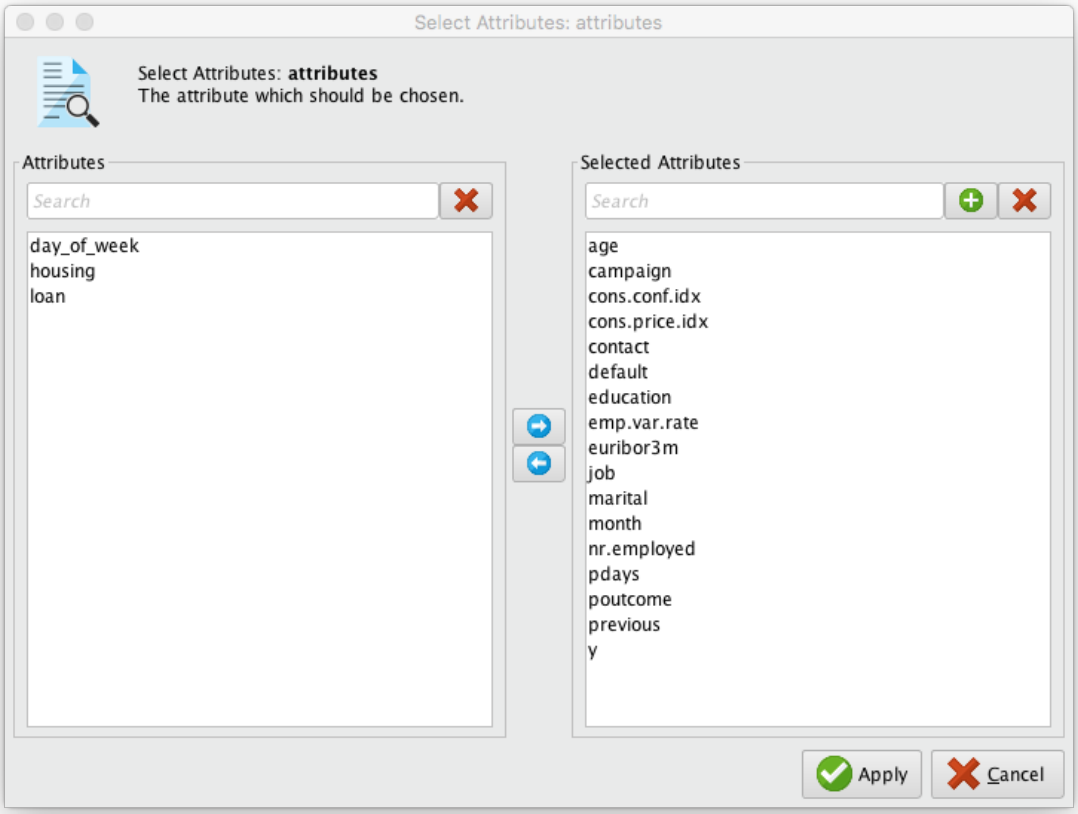

Figure 2. Attribute selection configuration

Then, we need to consider the “unknown” values. On condition that our dataset has 41.188 entries we suppose that it is enough to have a 5% significance level, which is about 2.000 entries. It is necessary to optimize the task without any significant influence on the result of the whole research. So, we can remove the entries, which has smaller number of “unknown” values and recovery the others. To recover the damaged values we will use the further method of the interpolation, which based on the correlation of values with values of other attributes:

1. We need to find the maximum absolute value of the correlation with other attributes;
2. Consider the sign of the correlation coefficient;
3. Find a pattern;
4. Replace these “unknown” values by the found pattern.

We need to recognize all the attributes with the possible “unknown” values, considering that attributes “load”, “housing” and “day_of_week” excluded of the research on the previous step.

Attribute	“Unknown” values	Ratio of the “unknown” values
Job	330	0.008
Marital	80	0.002
Education	1731	0.042
Default	8597	0.209

Table 2. Unknown values’ statistics

Regarding the results, we can see the we need to work only with “default” attribute and remove other entries to optimize the task. Regarding this attribute, we can see that we have 32.588 “no” (79.25% of the all entries) and 3 “yes” (0.007%) values. Unfortunately, this set is not enough for building any model. So, we can assume that other “unknown” values could be “no”. Since, more than 99% of the “default” values are the same, we could also exclude this attribute, because it will be useless.

- Unknown: 20.9%
- No: 79.25%
- Yes: 0.007%

Finally, 4 attributes were excluded from the dataset - {day_of_week, housing, loan, default}.

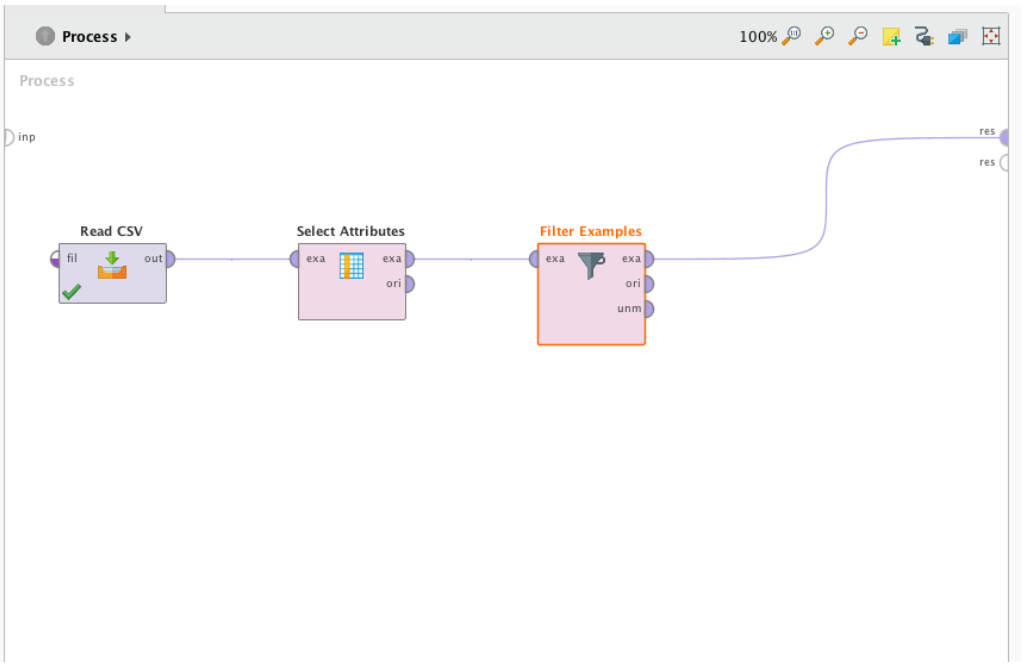

Figure 3. Selection with filtering

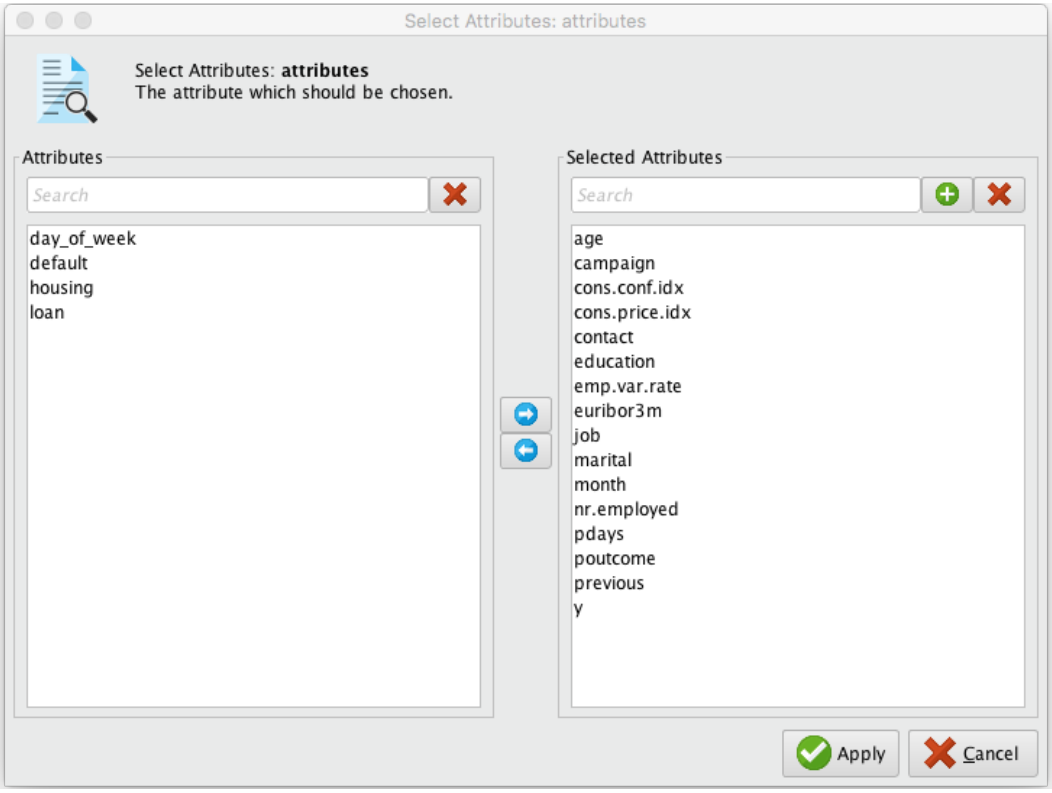

Figure 4. Attribute selection configuration

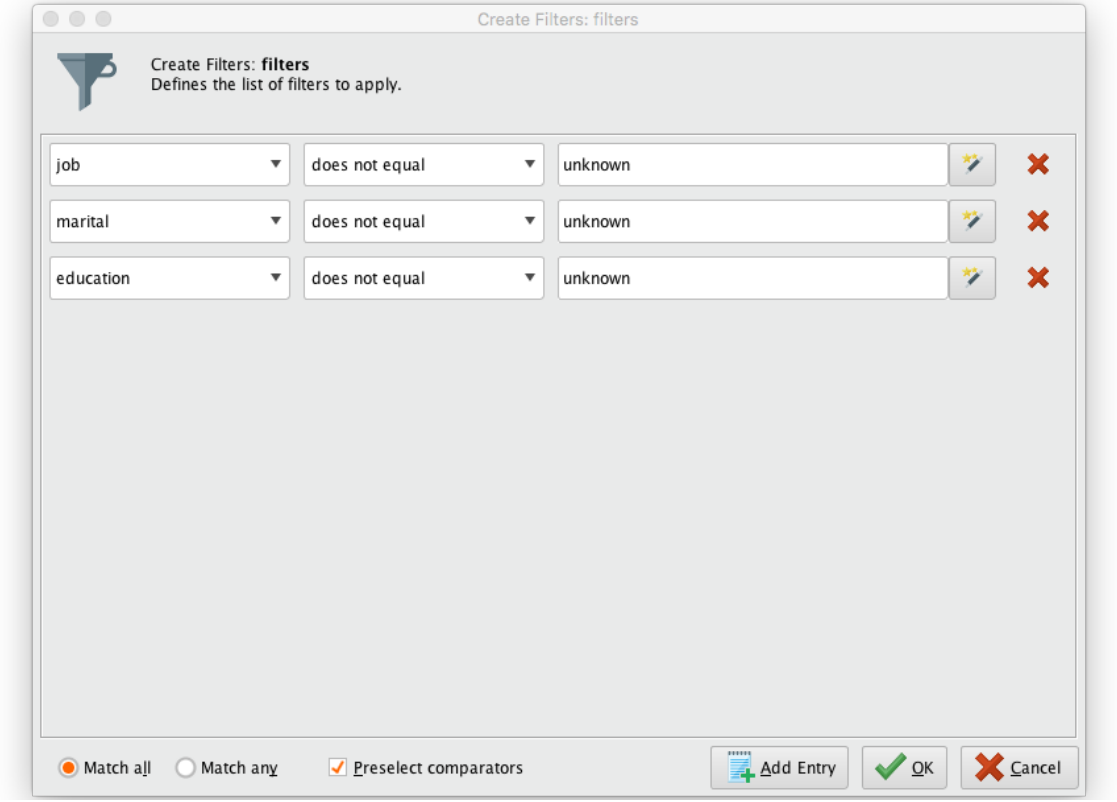

Figure 5. Filtering configuration

For the last data preprocessing stage, we have to replace all the nominal values by corresponding numerical to enable the use of the various classification techniques used later. In particular this is a necessary requirement for using SVM classification.

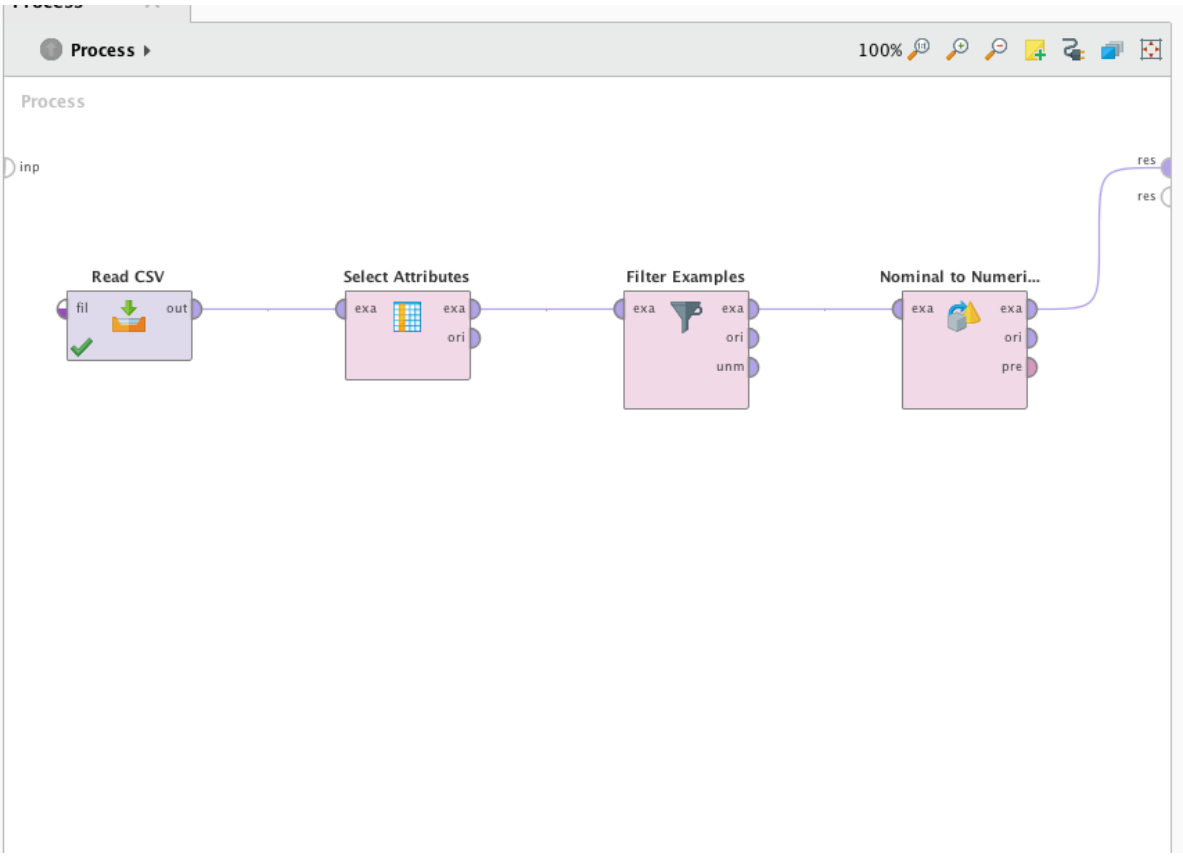

Figure 6. Attribute selection with filtering and "Nominal to Numerical" converting.

Dataset Preprocessing

We also use multiply operator in order to apply the same preprocessed set for all the required prediction methods.

Due to the lack of test set, as usual for data mining tasks, we have to take 70% of the prepared data as "training set" and 30% as a "test set". In order to achieve this, we need to use "Split Data" operator.

Data Mining Stage

Decision tree

The Decision tree method creates a collection of nodes to be used like a flowchart to decide on the outcome of a attribute. Each node can be interpreted as an if statement that looks at a specific attribute of the dataset. To obtain the nodes corresponding attributes in the tree, information gain techniques are used to find out which attribute results in the best split.

1. “Numerical to Polynomial”. We change the “y=yes” attribute of the dataset and rename it as “y”.
2. “Split Data” into 70% Training Data and 30% Testing Data.
3. For the decision Tree operator we use the “Set Role” on the “y” attribute to make it a label as it is required for the decision tree operator and “Remove y-labels” - removing the not needed “y = no” attribute in the training dataset.
4. Train the model with the training dataset
5. “Remove y-labels” - Removing the “y” and “y - no” attributes for the training set
6. Create the “Decision Tree” with criterion=”gain_ration”, maximal depth=20, confidence=0.25, confidence=0.25 and allow pruning.
7. “Apply Model” takes the trained Tree and executes it on the test data.
8. “Join” and “Set Role” joins he test data and classified test data and sets the role of the attribute “y” to label so it can be used by the “Performance Tree” operator. With it, we can see how accurate the “Decision Tree” operator was to test the data.

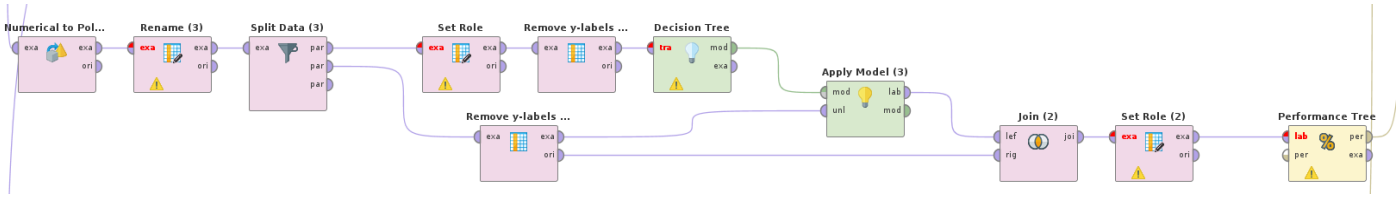

Figure 7. Decision tree data processing sequence

K-means

The K-means method is designed to conduct clusterization of the instances into predefined number of clusters with aim to minimize the within-cluster sum of squares (WCSS). The subtle moment of the algorithm is applying a clusterization method for classification purpose. In order to do that, one can apply 2-means method and associate one of the result clusters with “y=yes” value and “y=no” to another one.

1. “Split Data”. As it was told 70% of the instances go to training part (cluster definition) and 30% go as test instances.
2. “Clustering”. Applying K-means clustering with K=2, max runs = 10, BregmanDivergences as measure types, SquaredEuclideanDistances as divergence.
3. “Remove y-labels” is done since we need unlabeled test dataset.
4. “Apply Model” assigns the testset to one of two clusters defined in the “clustering”.

5. “Set Role (for Map)” resets attribute “cluster_0” role to regular in order to map the values of this attribute to value of “0” or “1” for sake of future comparing and performance measuring. The value assignment is done according to how it was decided to associate the label values and the obtained clusters.
6. “Set Role Prediction” sets back the role of “cluster” as prediction.
7. “Rename to y” renames “cluster” attribute to “prediction(y)”.
8. “Rename” renames “y = yes” attribute of the test dataset to “y”.
9. “Numerical to polynomial” converts numerical “y” attribute to corresponding polynomial attribute since we are performing classification (not regression).
10. “Set role label” sets “y” role as label.
11. “Join” merges the labeled test data with corresponding clusterization results.
12. “Performance” evaluates the efficiency of the model, calculates the values of accuracy, precision and recall.

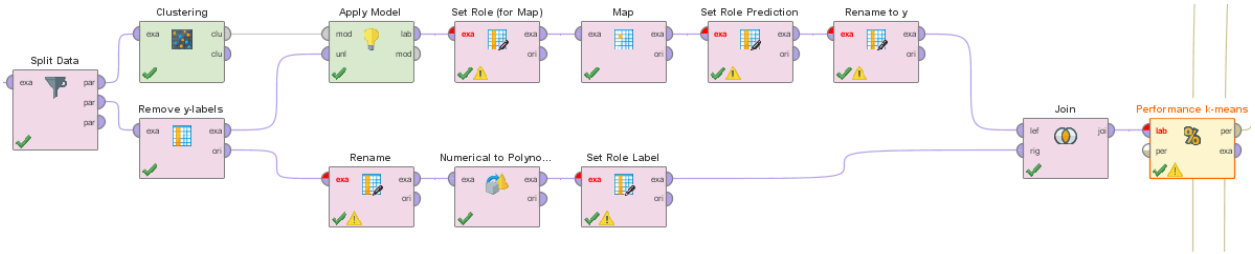

Figure 8. The K-means data processing sequence

K-medoids

The k-medoids methods provides us the opportunity to make a clusterization of the dataset with randomly selected its centers. Then, we have to remove the “y” attribute of the “test set” for further applying the trained model to this subset. We have to reduce number of steps for the operators’ runs and optimization steps because of the high computation costs. So, we will use 2 runs and 2 optimization steps. It is possible to increase number of steps for better accuracy in the future. In comparison with k-means method, the k-medoid provide us more balanced and distributed result.

Regarding the process, we will start from the previously prepared dataset, which had been divided into 2 subsets: “training set” and “test set”. So, we need to run the following operators:

1. “K-Medoids” clustering with the following parameters (all the parameters have the default value if it is not mentioned);
2. “Select Attributes” operator as a parallel action to remove the “y” attribute from the “test set”. After that, two output streams are going to the inputs of the “Apply model” and “Rename” operators to apply the calculated model for this dataset.

2.1. Then we have to “Apply model” to the whole dataset.

2.1.1. After that it is necessary to set role of the cluster with target role “regular”;

2.1.2. Then we need to apply the “Map” operator;

2.1.3. After that we changed the role of the cluster to “prediction”;

2.1.4. Finally, we set new name of the predicted value from “cluster” to “prediction(y)”;

2.2. As a parallel process, we need to rename the left attribute;

2.2.1. Then we changed all the numerical values back to nominal;

2.2.2. Finally, we set role of the “y” attribute as “label”;

3. Inner Join operator, which is aimed to join the predicted and original attributes;
4. Finally, we are recognizing the performance of the applied k-medoids method to estimate the accuracy of final data.

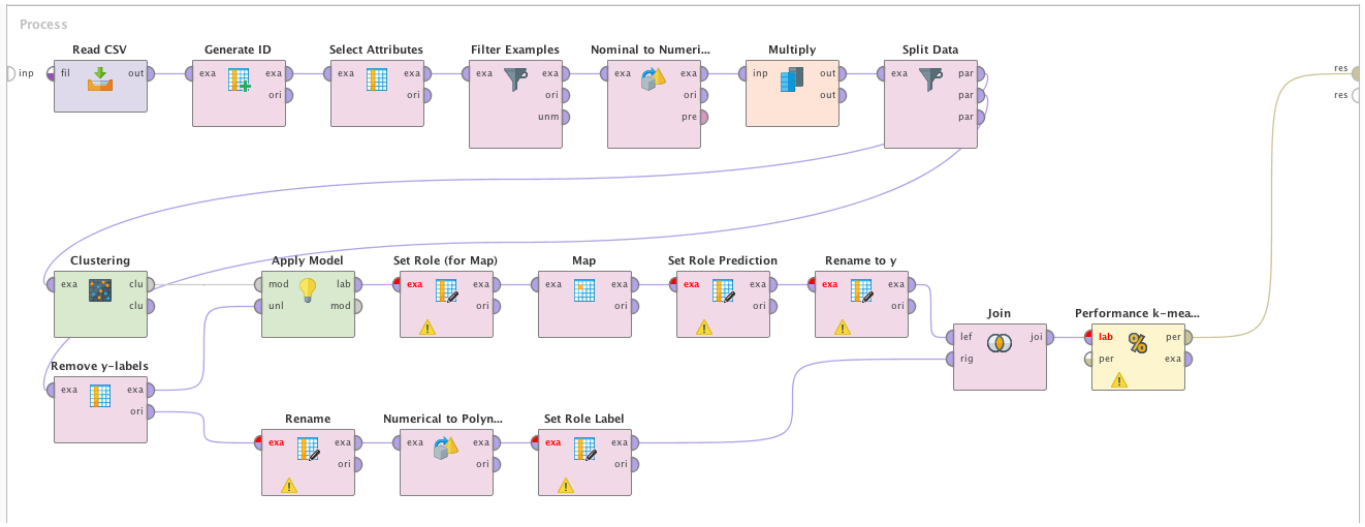

Figure 9. The whole data-mining process of K-medoids

SVM

Support Vector Machine is a method that can be used for solving classification/regression problems. It is based on the widely-used concept of defining an optimal hyperplane, that will separate the vector space in such way, that the margin value, dividing class areas is maximized. The output of the SVM training is a vector of weights that correspond to all the dataset attributes. Based on a linear combination of these weights with the attribute values, it is decided whether an instance belongs to one or another class (i.e. binary classification).

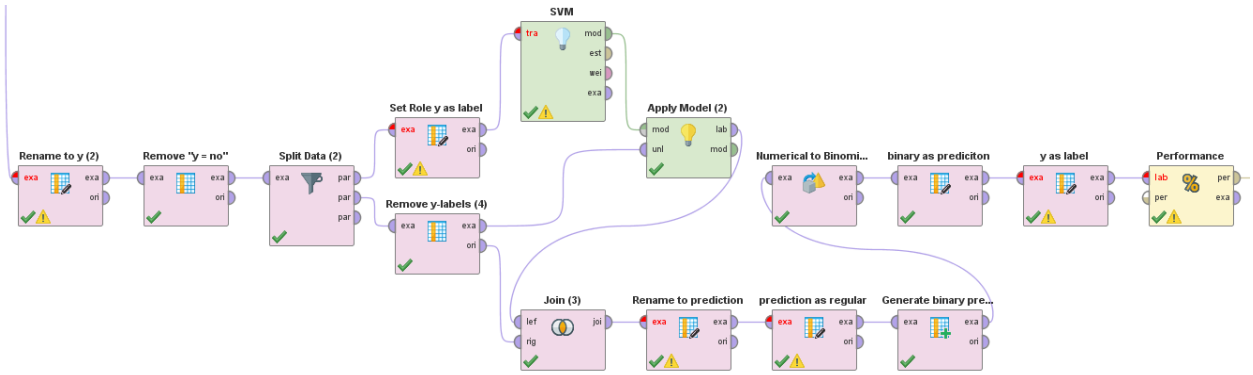

Figure 10. SVM data mining sequence

1. "Rename to y(2)" renames "y = yes" attribute to "y".
2. "Remove y = no" removes "y = no" attribute since only one dataset label attribute will be required for applying the SVM method.
3. "Split Data (2)" splits data into 70% trainset and 30% testset.
4. "Set Role y as label" sets y attribute as label attribute.
5. "SVM" trains an SVM model with the following parameters. Kernel type="dot", C="10.0", L_pos = P_neg = 1.0. Higher C parameters correspond to higher fitting to the input trainset and, however, therefore it may lead to overfitting. L_pos and P_neg define the importance of recalling the two result classes. Since we have no initial preference, we have set them both as 1.0.
6. "Remove y-labels (4)" removes y labels since they are not needed for the model applying.
7. "Apply model(2)" applies the obtained SVM model for the specified testset.
8. "Join (3)" matches the predicted values to their original labelled instances.
9. "Rename to prediction" renames "prediction(y)" to "prediction". It is done because a name with brackets will not be possible to use for conditional values reassignment in future.
10. "prediction as regular" sets role of "prediction" attribute to regular.
11. "Generate binary prediction" is the final detail of SVM application. Since we have a linear combination of the attribute weights with their values, after its computation we have to define the border-value of this linear combination that will separate the two classes. This operator creates an attribute "binary" having value 0 or 1 depending whether the "prediction" value if higher or not than the predefined border-value. This is implemented by inserting an expression "if(prediction > 10, 1, 0)" as the substituting value.

12. “Numerical to Binomial” makes the “prediction” and “binary” attributes binomial in order to estimate the efficiency of the classification.
13. “binary as prediction” sets “binary” attribute role as prediction.
14. “y as label” sets “y” attribute role as label.
15. “Performance” calculates the efficiency of the classification.

An important step, that affects the prediction accuracy is defining of the border-value. Problem is that we face a dilemma of preference between a good recall for classes “0” and “1”. Low border-values (as -1) will favor predicting “yes” rather than “no” and vice versa. Ideally one should draw the ROC-curve in order to illustrate the dilemma, but we will simplify this by considering the following table.

Border-value	Accuracy	“no” recall	“yes” recall
-1	67.13	66.23	74.11
0	71.95	72.73	65.61
1	79.31	82.05	57.11
2	83.52	87.69	49.85
3	86.48	91.97	42.04
4	87.75	93.84	38.56
5	88.3	94.77	35.94
6	88.64	95.56	32.69
7	89	96.2	30.83

8	89.26	96.69	29.13
9	89.37	97.26	25.58
10	89.52	97.67	23.65

Table 3. Support Vector Machine manual tuning results

We will select a border-value that will have relatively high accuracy (>75%) and satisfactory recall for both classes (>50%). Performing a manual tuning with different border-values from -1 to 10 we can see that the only SVM model performing more and less well is the one that has +1.0 as its border-value.

Models evaluation

To estimate the data, we will use the next metrics (we have no prescribed values, so we will estimate the method by comparing its results with the results of other recognized method):

- Accuracy;
- Precision coefficient (both classes);
- Recall coefficient (both classes);
- Run time;

Decision Tree

We obtain the following Decision Tree:

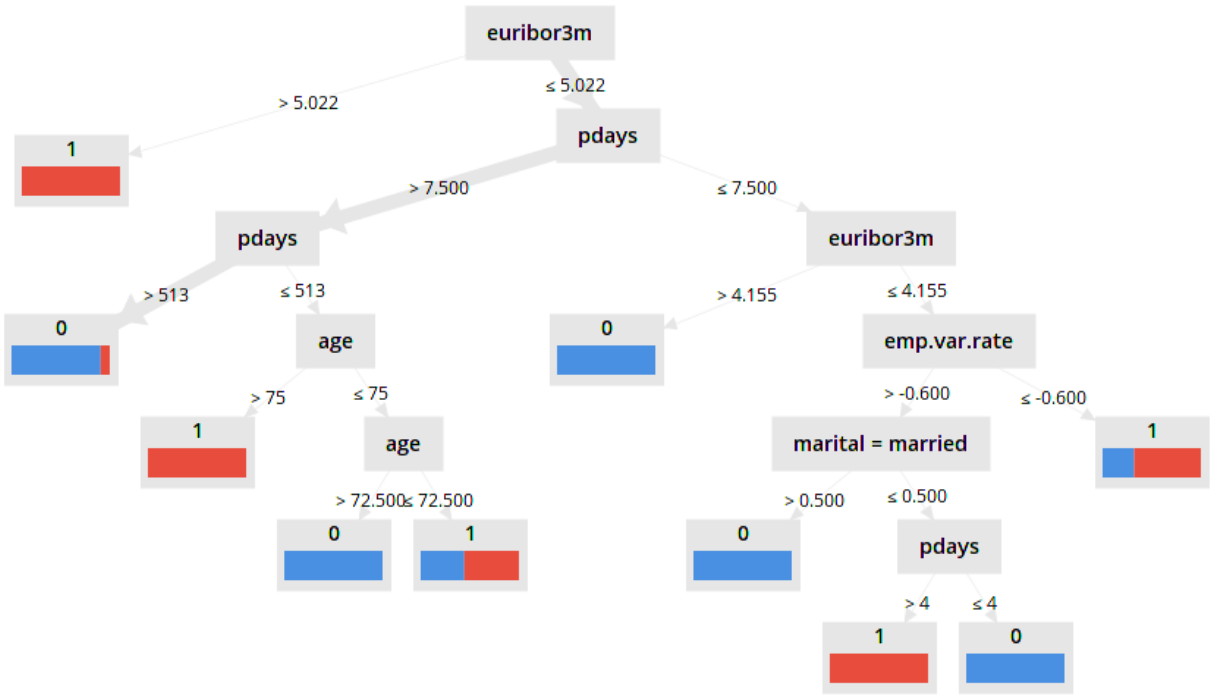

Figure 11. Decision tree illustration

With the following accuracy:

accuracy: 89.83%

	true 0	true 1	class precision
pred. 0	10319	1052	90.75%
pred. 1	144	242	62.69%
class recall	98.62%	18.70%	

Table 4. Decision tree evaluation results

The computation of the Tree method is rather fast with 1 second. We obtain an accuracy of 98.62% for “no” and 18.70% for “yes”. Indeed, the small number of True “yes”es do have a negative effect on the Decision Tree method.

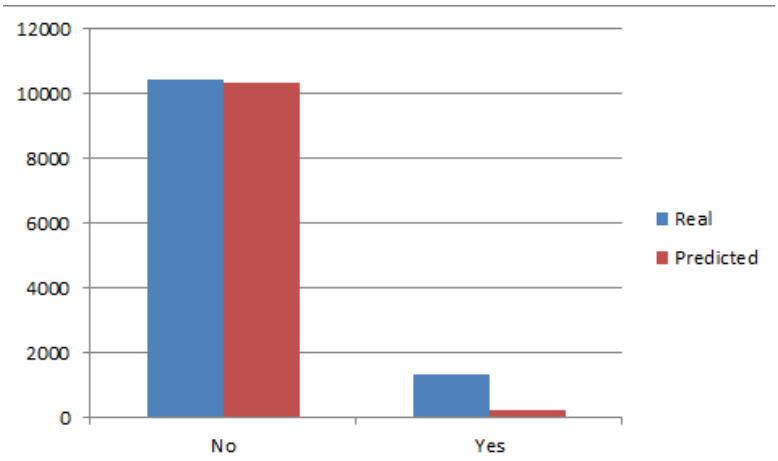

Figure 12. Real/predicted instance populations comparison.

K-means

As the result of the clusterization process, we obtain the clusters as follows:

Attribute	cluster_0	cluster_1
job = housemaid	0.02615982786606772	0.030752916224814422
job = services	0.09912800573779774	0.05090137857900318
job = admin.	0.25438828281303083	0.3075291622481442
job = blue-collar	0.23004039107621457	0.0911983032873807
job = technician	0.16756634328639916	0.15588547189819724
job = retired	0.038201653391717945	0.12513255567338283
job = management	0.07225095315390133	0.07529162248144221
job = unemployed	0.024272394398097467	0.043478260869565216
job = self-employed	0.0349175191574497	0.02545068928950159
job = unknown	0.0	0.0
job = entrepreneur	0.03631421992374769	0.021208907741251327
job = student	0.016760409195575855	0.07317073170731707
marital = married	0.6107357215658148	0.5387062566277837
marital = single	0.27681099241251744	0.36585365853658536
marital = divorced	0.11245328602166774	0.09544008483563096
marital = unknown	0.0	0.0

education = basic.4y	0.10660224227095995	0.09968186638388123
education = high.school	0.24185572458570836	0.23223753976670203
education = basic.6y	0.05994488694273527	0.030752916224814422
education = basic.9y	0.1552602770752331	0.08589607635206786
education = professional.course	0.13170510739496433	0.13997879109225875
education = unknown	0.0	0.0
education = university.degree	0.3043297723755238	0.41039236479321317
education = illiterate	3.0198935487524064E-4	0.0010604453870625664
contact = telephone	0.37325884262579745	0.07529162248144221
contact = cellular	0.6267411573742026	0.9247083775185578
month = may	0.3410969763315843	0.15906680805938495
month = jun	0.12977992525763468	0.10816542948038176
month = jul	0.17775848401343852	0.08589607635206786
month = aug	0.14903174663093127	0.15906680805938495
month = oct	0.01366501830810464	0.09968186638388123
month = nov	0.10131742856064324	0.11664899257688228
month = dec	0.0034728775810652676	0.02863202545068929
month = mar	0.011551092823977954	0.05938494167550371
month = apr	0.06277603714469066	0.07953340402969247
month = sep	0.009550413347929486	0.10392364793213149
poutcome = nonexistent	0.8960401645841984	0.0
poutcome = failure	0.10395983541580159	0.08589607635206786
poutcome = success	0.0	0.9141039236479321
y = no	0.9080064927711298	0.36585365853658536
y = yes	0.09199350722887019	0.6341463414634146
age	39.76135291230984	42.09756097560975
campaign	2.5931070929749724	1.8738069989395547
pdays	999.0	6.07847295864263
previous	0.1166056396512023	1.6246023329798516

emp.var.rate	0.1626061681329598	-2.092576882290551
cons.price.idx	93.58123098408538	93.34030646871597
cons.conf.idx	-40.646008078206286	-38.41919406150552
euribor3m	3.7204697821908796	0.9926977730646903
nr.employed	5172.520399379121	5030.281230116641

Table 5. 2-means clustering result

Analyzing the obtained clusters, one can associate cluster0 with “y = no” and cluster1 with “y = yes”.

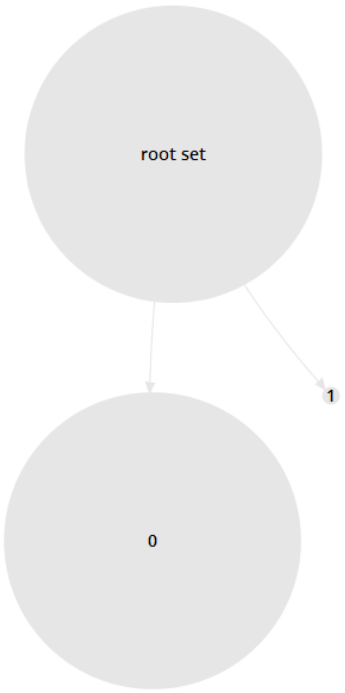

Figure 13. Clusters visualisation

As a result of the evaluation process we have the following table:

accuracy: 89.80%

	true 0	true 1	class precision
pred. 0	10267	1034	90.85%
pred. 1	165	291	63.82%
class recall	98.42%	21.96%	

Table 6. K-means evaluation results

The computation process takes roughly 7 seconds to execute.

89.8% accuracy can be considered as a positive performance of the data mining method.

Class prediction precisions are 90.85% and 63.82% which can be considered satisfactory. The recall of the cluster0 is 98.42%, however it is likely to be caused by a very large number of actual “no” answers.

The recall of cluster1 has poor performance, which may be caused by low ratio of the actual “yes” answers.

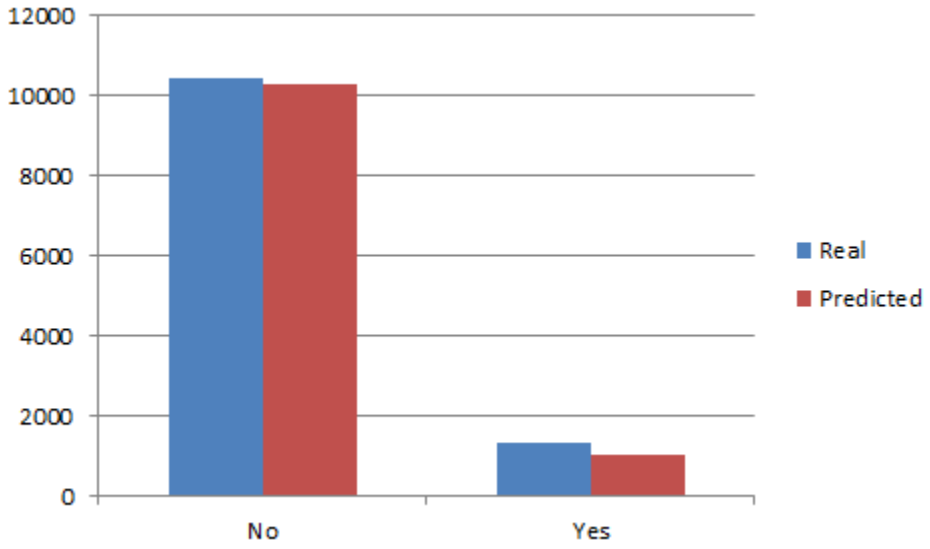

Figure 14. Real/predicted instance populations comparison.

Finally we can regard the efficiency of the method as relatively satisfactory, however one must remember that K-means method only manages to find the locally-optimal cluster definition. Thus, the efficiency will strongly depend on the actual datasets and the way the initial clusters are seeded.

K-medoids

After the calculations, we have received two cluster, which were selected by k-medoids method and received the confusion matrix after the performance estimation.

After 2 runs with 2 optimizations we have received only 10.24% of accuracy, which means that it is necessary to increase number of runs to receive better result. It happened because of the k-medoids method principals, which are based on the randomly selected medoids.

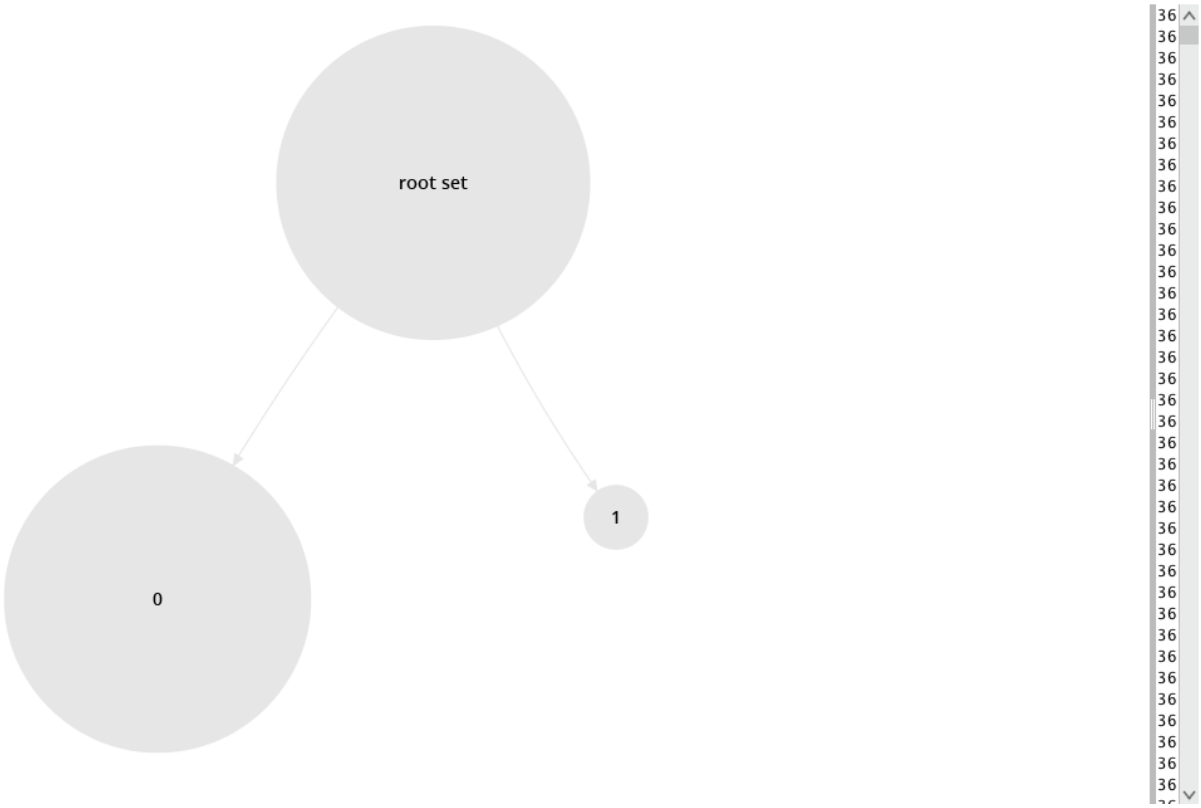

Figure 15. Clusters visualisation

Attribute	cluster_0	cluster_1
job = housemaid	0	0
job = services	0	0
job = admin.	1	0
job = blue-collar	0	0
job = technician	0	0
job = retired	0	1
job = management	0	0
job = unemployed	0	0
job = self-employed	0	0
job = unknown	0	0
job = entrepreneur	0	0
job = student	0	0
marital = married	0	1
marital = single	0	0
marital = divorced	1	0
marital = unknown	0	0
education = basic.4y	0	0
education = high.school	0	0

Table 7. Predicted conditions of data for two clusters

Attribute	cluster_0	cluster_1
education = basic.6y	0	0
education = basic.9y	0	0
education = professional.course	1	1
education = unknown	0	0
education = university.degree	0	0
education = illiterate	0	0
contact = telephone	1	0
contact = cellular	0	1
month = may	1	0
month = jun	0	0
month = jul	0	0
month = aug	0	0
month = oct	0	0
month = nov	0	1
month = dec	0	0
month = mar	0	0
month = apr	0	0
month = sep	0	0

Table 8. Predicted conditions of data for two clusters

age	45	74
campaign	1	3
pdays	999	999
previous	1	1
emp.var.rate	-1.800	-1.100
cons.price.idx	92.893	94.767
cons.conf.idx	-46.200	-50.800
euribor3m	1.270	1.028
nr.employed	5099.100	4963.600

Table 9. Predicted conditions of data for two clusters

According to the results, we can see that from the whole population of “test set”, which includes 11.575 records, the system predicted that value of our recognized attribute “y” as “no” in 153 records, while “yes” in 10310 records. Similarly, we can see that in 243 cases the system predicted the “y” answer as “n”, and 1051 other records. We can see that the result is better in comparison with the negative answers. So, we can see the cost-sensitive measures of our model. We have received 1.46% of class recall coefficient for the negative answers and 81.22% for positive ones respectively. Also, we can see the precision evaluation of the selected method, which has 38.64% of precision for all the predicted negative answers and 9.25% for positive ones.

The whole data mining process has taken 3 hours for this stage only, so, it is a poor result in comparison with the rest of the classification and clustering techniques.

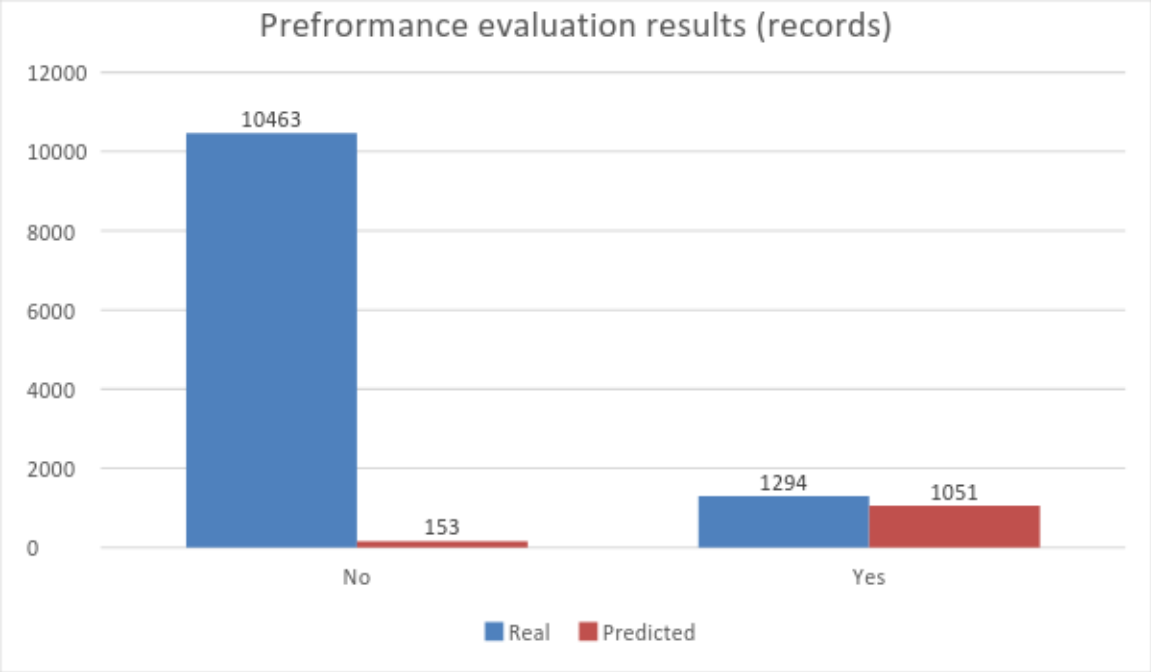

Figure 16. Instance polutation comparison

Table View Plot View

accuracy: 10.24%

	true 0	true 1	class precision
pred. 0	153	243	38.64%
pred. 1	10310	1051	9.25%
class recall	1.46%	81.22%	

Table 10. Decision tree evaluation results

PerformanceVector

```
PerformanceVector:
accuracy: 10.24%
ConfusionMatrix:
True:   0       1
0:      153     243
1:      10310   1051
```

Figure 17. Estimation of the process

Finally, this method provides us an opportunity to divide the data into two clusters (in this case) which are more normalized in comparison with k-means. Unfortunately, this method takes significantly more time to calculate the result, so, it is cannot be applied in time-dependent tasks. We have estimated this method with 2 runs and 2 optimization steps, which is not enough (according to the result). As a result, we received non-acceptable value of accuracy. Moreover, it is important that this method takes significantly more time to calculate than others, so, it is not the best method for given task.

SVM

The algorithm performs everything is 3 seconds.

accuracy: 82.15%

	true false	true true	class precision
pred. false	9009	685	92.93%
pred. true	1414	649	31.46%
class recall	86.43%	48.65%	

Table 11. Decision tree evaluation results

As it was said in the data mining stage, with the specified border-value the algorithms renders a satisfactory efficiency. However as it was described earlier, the performance strongly depends on the tuning stage. One can favor better recall for one of these classes at expense of the recall of another one.

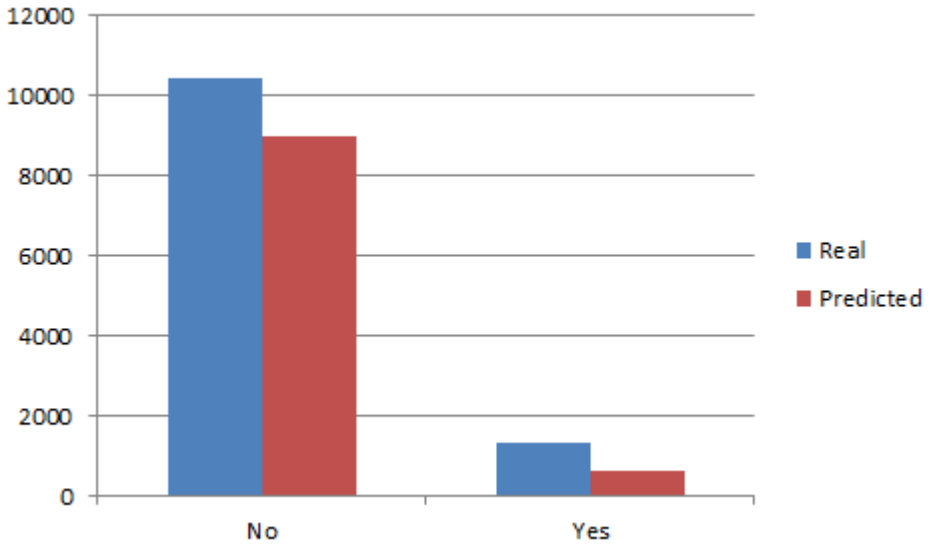

Figure 18. Real/predicted instance populations comparison.

Overall comparison

Several measures will be used to make the overall comparison. The best performance per measure is marked by green color.

Method\measure	Accuracy, %	"no" recall, %	"yes" recall, %	"no" precision, %	"yes" precision, %	Runtime, seconds
Decision tree	89.83	98.62	18.70	90.75	62.69	1
K-means	89.80	98.42	21.96	90.85	63.82	7
K-medoids	10.24	1.46	81.22	38.64	9.25	~10800
SVM	82.15	86.42	48.65	92.93	31.46	3

Table 12. Overall measures comparison

The measure comparison does not define a single "winner" method in this process. Every of the methods has a better performance than the others at some measure, and thus, is non-dominated. Nonetheless decision tree has more well-performing measures. Once again, some methods can be tuned in order to elevate some measures by means of the others.