FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION FOR HIGHER EDUCATION NATIONAL RESEARCH UNIVERSITY HIGHER SCHOOL OF ECONOMICS

Faculty of Computer Science

BIG HOME TASKS REPORT

On the course ORDERED SETD IN DATA ANALYSIS

Student: Burova Nadezhda

Group: мНОД21_ИССА

Supervisor: Strok Fedor

Moscow

2021

Contents

Introduction

The purpose of this home task is to implement lazy classification algorithm and check its effectiveness on different parameters sets. Also, comparison with other well-known classification methods is provided.

The idea of classification rule is following: if an object description coincide with some object from a positive context description in more than given number of positions (in terms of proportion over an overall number of attributes), return 1 (same for negative with 0). If this proportion is higher than given minimal threshold, calculate support in positive context. If it is greater than given number, return 1. If not, add support to overall support in plus-context, add support in negative class (counter-support) to the corresponding value, continue to intersect classifying object with objects from plus-context. When (if) finished with plus-context continue in the same way in negative. At the end if still not classified by other rules, compare proportions of counter-supports over supports (if the last not zeros), return class with smaller proportion. If one of supports is zero return class where not. If both are zeros return class with smaller proportion of counter-support over the size of context.

So, the parameters of classifier are: misup, which stands for support to be reached to be classified as class under consideration; min_conf which stands for a proportion of attributes that have to coincide with some example from class under consideration to be classified as this class; and aconf standing for proportion of attributes to coincide to take this intersection into consideration when calculation overall support and counter-support for classifying object.

Tic-tac-toe dataset

This "toy" dataset includes results of tic-tac-toe games. Each attribute shows whether there was 'x', 'o' or 'b' – blank space in a particular cell on gameboard. The target binary class is whether the first player (with 'x') won the game or not.

Binarization rule is: if attribute is 'x' put 1 else 0 (except the last attribute – class, of course).

The parameters are 0.9, 0.9, 0.7 for binarized data and 0.9, 0.9, 0.5 for original.

We see, here the accuracies are one of the highest. Two of methods overcoming lazy classification are involving ensembles of classifiers. One more is decision tree, which is known as the best-suitable for this data.

So, the accuracies are actually high.

Method	**Accuracy on cross-validations**
Naive Bayes – Bernoulli version	0.75
Naive Bayes – Gaussian version	0.72
Random forest	**0.99**
Gradient boosting	**0.96**
K-nearest-neighbours classifier	0.89
Linear discriminant analysis	0.73
Logistic regression	0.73
Linear support vector classifier	0.73
Decision tree	**0.97**
Lazy classification on binarized data	**0.97**
Lazy classification on non-binarized data	**0.95**

The following metrics were assessed:

True Positive

True Negative

False Positive

False Negative

True Positive Rate

True Negative Rate

Negative Predictive Value

False Positive Rate

False Discovery Rate

Accuracy

Precision

Recall

Metrics for binarized version:

```
{'FN': 0,
 'FP': 3,
 'TP': 62,
 'TN': 29,
 'A': 0.96,
 'TPR': 1,
 'TNR': 0.9,
 'NPV': 1,
 'PPV': 0.9,
 'FPR': 0.09,
 'FDR': 0.05,
 'P': 0.97,
 'R': 0.95}
```

For non-binarized:

```
{'FN': 4,
 'FP': 0,
 'TP': 58,
 'TN': 33,
 'A': 0.95,
 'TPR': 0.93,
 'TNR': 1,
 'NPV': 0.88,
 'PPV': 1,
 'FPR': 0,
 'FDR': 0,
 'P': 0.94,
'R': 0.96}
```

The values that must be high in case of good classifier are high, and those which must be low are low. So, we can conclude that classifier works good relying not only on its accuracy.

As for parameters selection, here are results for binarized data:

minconf	minsup	aconf	FN	FP	TP	TN	A	TPR	TNR	NPV	PPV	FPR	FDR	P	R
0,8	0,8	0,5	0	29	62,5	4,2	0,696961	1	0,126381	1	0,683153	0,873619	0,316847	0,841576	0,563191
0,8	0,8	0,7	0	29	62,5	4,2	0,696961	1	0,126381	1	0,683153	0,873619	0,316847	0,841576	0,563191
0,8	0,9	0,5	0	29	62,5	4,2	0,696961	1	0,126381	1	0,683153	0,873619	0,316847	0,841576	0,563191
0,8	0,9	0,7	0	29	62,5	4,2	0,696961	1	0,126381	1	0,683153	0,873619	0,316847	0,841576	0,563191
0,9	0,8	0,5	0	4,4	62,5	28,8	0,953848	1	0,867023	1	0,935532	0,132977	0,064468	0,967766	0,933512
0,9	**0,8**	**0,7**	**0**	**3,7**	**62,5**	**29,5**	**0,961205**	**1**	**0,888235**	**1**	**0,944791**	**0,111765**	**0,055209**	**0,972396**	**0,944118**
0,9	0,9	0,5	0	4,4	62,5	28,8	0,953848	1	0,867023	1	0,935532	0,132977	0,064468	0,967766	0,933512
0,9	**0,9**	**0,7**	**0**	**3,7**	**62,5**	**29,5**	**0,961205**	**1**	**0,888235**	**1**	**0,944791**	**0,111765**	**0,055209**	**0,972396**	**0,944118**

We get the highest value of accuracy on the parameter set which were used earlier. The same is on 0.9 0.8 0.7, so minsup is not the main parameter here. The accuracy drops slightly if we decrease aconf and dramatically when minconf. So the parameters can be ordered by significance as follows: minconf, aconf, minsup.

And for non-binarized:

minconf	minsup	aconf	FN	FP	TP	TN	A	TPR	TNR	NPV	PPV	FPR	FDR	P	R
0,8	**0,8**	**0,5**	**3,9**	**0,1**	**58,6**	**33,1**	**0,958308**	**0,937711**	**0,997059**	**0,898607**	**0,998276**	**0,002941**	**0,001724**	**0,948441**	**0,967385**
0,8	0,8	0,7	48,6	0	13,9	33,2	0,492094	0,222248	1	0,405976	1	0	0	0,702988	0,611124
0,8	**0,9**	**0,5**	**3,9**	**0,1**	**58,6**	**33,1**	**0,958308**	**0,937711**	**0,997059**	**0,898607**	**0,998276**	**0,002941**	**0,001724**	**0,948441**	**0,967385**
0,8	0,9	0,7	48,6	0	13,9	33,2	0,492094	0,222248	1	0,405976	1	0	0	0,702988	0,611124
0,9	**0,8**	**0,5**	**3,9**	**0,1**	**58,6**	**33,1**	**0,958308**	**0,937711**	**0,997059**	**0,898607**	**0,998276**	**0,002941**	**0,001724**	**0,948441**	**0,967385**
0,9	0,8	0,7	48,6	0	13,9	33,2	0,492094	0,222248	1	0,405976	1	0	0	0,702988	0,611124
0,9	**0,9**	**0,5**	**3,9**	**0,1**	**58,6**	**33,1**	**0,958308**	**0,937711**	**0,997059**	**0,898607**	**0,998276**	**0,002941**	**0,001724**	**0,948441**	**0,967385**
0,9	0,9	0,7	48,6	0	13,9	33,2	0,492094	0,222248	1	0,405976	1	0	0	0,702988	0,611124

Here we get a lot of sets with coinciding accuracies. They all have aconf=0.5. Thus, it is the main parameter in this case and other two do not make an impact if changed by 0.1. However, precision and recall are not so bad even in worst cases but they are less objective characteristics than accuracy. Still, if we have a reason to rely on some other than accuracy metrics, some of them have their best values on not-the-highest-accuracy set of parameters.

Adult dataset

This dataset is based on real data on census. It contains of people's characteristics linked with their biological (like sex and race) or social (education, workclass, etc) status. The target class attribute is whether some person's income is greater than 50k$ or not.

Here the results are not so brilliant as for tic-tac-toe example. Accuracies for both binary and non-binary cases are the lowest ones.

Actually, the main reason for this could be that not all data was used when making classification. This was done in such way because the dataset contains about 50k samples and so working on a whole dataset takes enormous times.

So, we can conclude from this that the algorithm are to be optimised in order not to take so many time. After that, new evaluations on this dataset could be done to make the results more reliable.

Method	**Accuracy**
Naive Bayes – Bernoulli version	0.76
Naive Bayes – Gaussian version	0.79
Random forest	0.85
Gradient boosting	0.86
K-nearest-neighbours classifier	0.73
Linear discriminant analysis	0.84
Logistic regression	0.79
Linear support vector classifier	0.73
Decision tree	0.81
Lazy classification on binarized data	0.62
Lazy classification on non-binarized data	0.65

Conclusions

Lazy classification method in proposed in this h/t form has both advantages and disadvantages. The main plus is that it can work with categorical data without its binarization or enumeration (the last is not always possible; only if categories are ordered). But the minus is that we need to calculate a support for each intersection and if we don't exit because of any other rule, this support will be calculated N times where N is a size of a whole dataset. And each support+counter-support calculation requires N*K iterations over the whole cells in dataset, actually (K – number of attributes).

Actually, in well-separated data first two rules of exit should help to overcome this problem. Still, if a lot of values occur frequently both in positive and negative contexts, they may not help and even spoil the classification if parameters are not specified correctly.

Thus, on the one hand, lazy classification helps in overcoming pre-computation of classifier and does not need any additional pre-processing of data (except some basic thing like deleting missing values, etc). This allows on-line using with adding new examples in context. On the other hand, it depends strongly on data on input and requires careful finding of parameters. In addition, on practice it takes a lot of time to provide a classification. However, the last may be the drawback of implementation.

About what can be done next to improve this work: optimise all used operations via making use of python features; compare with other FCA-based approaches (namely, JSM). Make more deep research on how the parameters affect the results.

References

1. *Adult*. (n.d.). Retrieved from UCI Machine Learning Repository: https://archive-beta.ics.uci.edu/ml/datasets/adult
2. Aha, D. (n.d.). *Tic-Tac-Toe Endgame*. Retrieved from UCI Machine Learning Repository: https://archive-beta.ics.uci.edu/ml/datasets/tic+tac+toe+endgame