Report

Lazy Learning with FCA toolbox

Undalov Nikolai

Data

Dataset: Statlog (Heart) (UCI Machine learning repository)
(https://archive.ics.uci.edu/ml/datasets/Statlog+%28Heart%29)
Dataset consist of 270 records with 13 mixed type features and one binary target (if there is a heart disease). Dataset has no missing values and already cleaned.
Features:

1. Age
2. Sex
3. Chest pain type
4. Resting blood pressure
5. Serum cholesterol
6. Fasting blood sugar
7. Resting electrocardiographic results
8. Maximum heart rate
9. Exercise induced angina
10. Oldpeak
11. The slope of peak exercise
12. Number of major vessels
13. Thal

Feature types:

- Real: 1, 4, 5, 8, 10, 12
- Ordered: 11
- Binary: 2, 6, 9

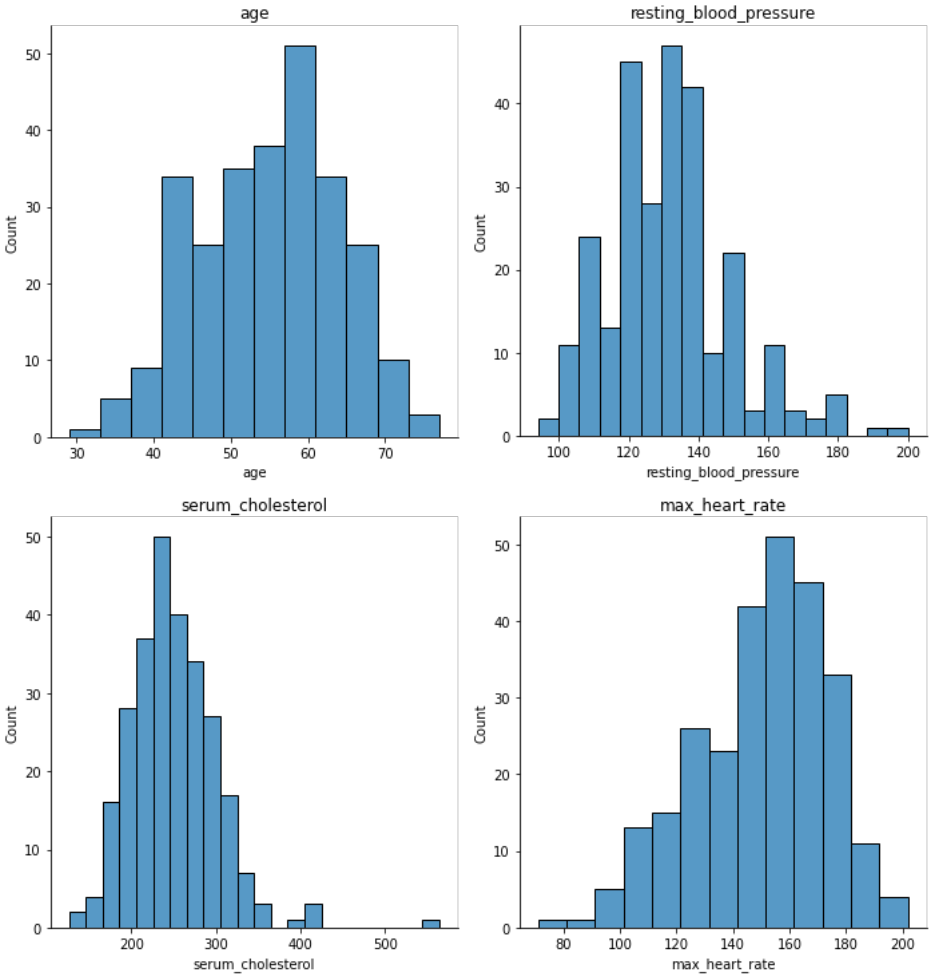
- Nominal 3, 7, 13

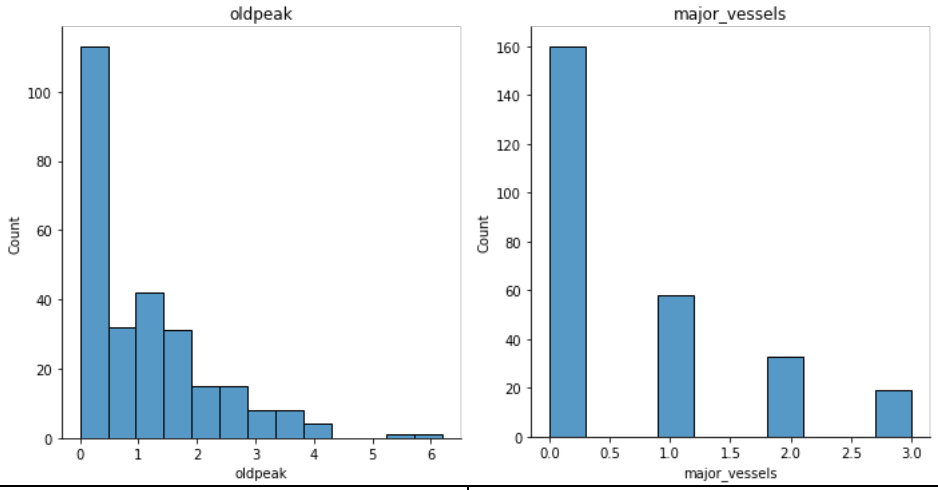
Target: Absence (0) or presence (1)

Data preparation:

All real valued features were converted to categorical using discretization.

Number of bins determined based on domain knowledge and data distribution.





Feature	Bins borders
age	[30.0, 40.0, 50.0, 60.0, 70.0]
resting_blood_pressure	[100.0, 120.0, 140.0, 160.0, 180.0, 200.0]
serum_cholesterol	[200.0, 300.0, 400.0, 500.0]
max_heart_rate	[80.0, 100.0, 120.0, 140.0, 160.0, 180.0, 200.0]
oldpeak	[0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0]
major_vessels	[0.0, 1.0, 2.0, 3.0]

Features after binarization represented as index of corresponding bin. All features except already binary ones transforms to binary. Prepared dataset has 53 binary features.

Dataset is divided to train and test parts. Test part contains 27 records (10%) and train part contains 243 records (90%).

Classification

Naïve Bayes (Gaussian) algorithm is used as baseline. In the work sklearn implementation of this algorithm is used. Parameters are rest default.

FCA lazy classification algorithms are based on generator framework. In the work different step functions are explored.

Algorithm:

Suppose we are given Formal Context $C = (G, M, I)$, G- set of objects, M– set of features, $I \subseteq G \times M$– set of relations, $L(g), g \in G$ – label of object g (+ or -), $'$– prime operator. To classify object g_t we need:

1. For each object $g_i \in G_+$ ($G_+ = \{g \in G | L(g) = +\}$) we calculate intersection of test and current objects' features $d_{i+} = {g'}_i \cap {g'}_t$. And count how many times this intersection present in objects of G_- : $fe_{i+} = |\{g \in G_- | d_{i+} \subseteq g'\}|$.
2. Do same thing for $g_i \in G_-$ vice versa, calculating d_{i-} and fe_{i-}
3. Apply classification rule based on $|d_{i-}|, |d_{i+}|, fe_{i-}, fe_{i+}$ to get label for g_t

Classification rule 1 (V1)

1. For label + and – sort in increasing oreder corresponding statistics $p_{i+} = (|d_{i+}|, fe_{i+}), p_{i-} = (|d_{i-}|, fe_{i-})$ with composite key $\left(fe_{i+}, \frac{1}{|d_{i+}|+1}\right)$ and $\left(fe_{i-}, \frac{1}{|d_{i-}|+1}\right)$ (comparing is performed from left to right until first inequality).
2. From beginning of pairs p_{i+} and p_{i-} compare , fe_{i+} and , fe_{i-} if , $fe_{i+} > fe_{i-}$ then classify as -, if $fe_{i+} < fe_{i-}$ then classify as +, if $fe_{i+} = fe_{i-}$ then : if $|d_{i+}| > |d_{i-}|$ then classify as +, if $|d_{i-}| > |d_{i+}|$ then classify as -, if $|d_{i-}| = |d_{i+}|$ move to the next pair. If all pairs are considered (cuts by minimal length of list) then return + as default.

Classification rule 2 (V2)

Method has one parameter top_k which controls size of final set of appropriate candidates to make prediction.

1. For label + and – sort in increasing oreder corresponding statistics $p_{i+} = (|d_{i+}|, fe_{i+}), p_{i-} = (|d_{i-}|, fe_{i-})$ with composite key $\left(fe_{i+}, \frac{1}{|d_{i+}|+1}\right)$ and $\left(fe_{i-}, \frac{1}{|d_{i-}|+1}\right)$ (comparing is performed from left to right until first inequality).
2. Define $n_+ = \min(top\_k, |G_+|)$
3. Caclulate $s_+ = \frac{1}{n_+}\sum_{i=1}^{n_+} \frac{|d_{(i)+}|}{fe_{(i)+}+1}$ ((i) – index in sorted list) and similarly s_-.
4. If $s_+ > s_i$ classify as + else as –

Results

Baseline algorithm, and versions of FCA lazy classificators was scored on test part of dataset given train part as context. Following list of metrics is used:

- 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
- F1

Results presented in table below.

Metric	**Naïve Bayes**	**V1**	**V2** $(top\_k = 10)$	**V2** $(top\_k = 3)$	**V2** $(top\_k = 30)$
TP	10	**11**	10	10	8
TN	13	12	**14**	13	13
FP	2	3	**1**	2	2
FN	2	**1**	2	2	4
TPR	0.83	**0.92**	0.83	0.83	0.67
TNR	0.87	0.8	**0.93**	0.87	0.87
NPV	0.87	0.92	**0.88**	0.87	0.76
FPR	0.13	0.2	**0.07**	0.13	0.13
FDR	0.17	0.21	**0.09**	0.17	0.2
Accurancy	0.85	0.85	**0.89**	0.85	0.78
Precision	0.83	0.79	**0.91**	0.83	0.8
Recall	0.83	**0.92**	0.83	0.83	0.67
F1	0.83	0.85	**0.87**	0.83	0.73

From measured metrics we can conclude that lazy classifiers works better than Naïve Bayes. The most accurate classifier by most of metrics is V2 with $top\_k = 10$. Comparing versions of V2 algorithm we can see that too small or too high values of top_k leads to poor performance.