KRKPA7 problem in various data mining software
4IZ451 - Knowledge discovery in databases

Tomáš Maršálek

December 6, 2014

1 Introduction

The purpose of this work is to analyze database of KRKPA7 problem using four different data mining tools. Some of the chosen data mining programs are freely available (Weka) or freely available for non-commercial use (RapidMiner) or proprietary licensed (Enterprise Modeler, IBM PASW). The nature of problem for this assignment is suitable for classification and association modeling, but not for clustering.

2 KRKPA7 problem

KRKPA7 is a shorthand for King Rook versus King Pawn on A7, which is a chess endgame. The black player has a king and a pawn left, where the pawn is on A7, which is one turn away from turning his into a queen. The white player has a turn in this situation and has a king and a rook to play with.

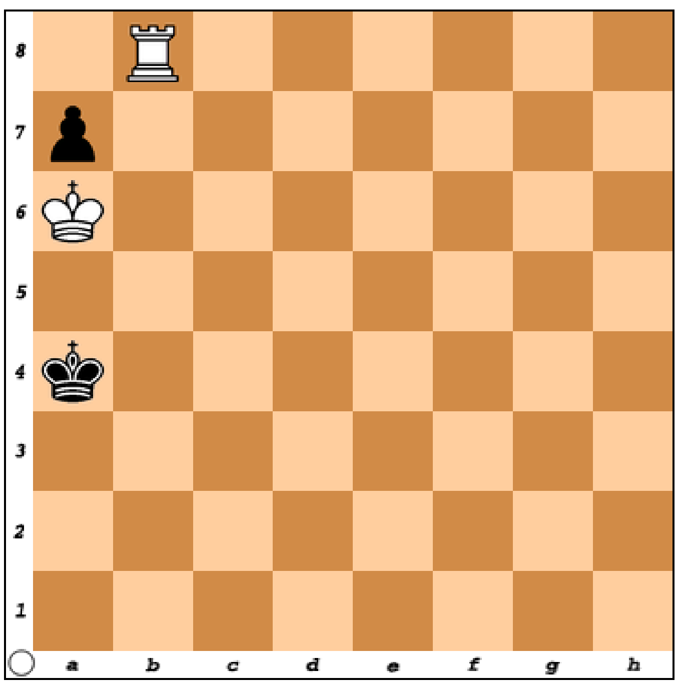

Figure 1: Example board setup

2.1 Data set

Data set contains 3196 setups of chess board, each one with different positions of both kings and the rook, the pawn is fixed on A7. The 37th column is a class column denoting **win** or **nowin** of white player. For our purposes we don't need to perform any preprocessing. The set is not as large for the algorithms to be unable to create desired models.

Each board setup is described by 36 attributes described in table 2.

Position	**Abbreviation**	**description**
1	bkblk	the BK is not in the way
2	bknwy	the BK is not in the BR's way
3	bkon8	the BK is on rank 8 in a position to aid the BR
4	bkona	the BK is on file A in a position to aid the BR
5	bkspr	the BK can support the BR
6	bkxbq	the BK is not attacked in some way by the pro- moted WP
7	bkxcr	the BK can attack the critical square (b7)
8	bkxwp	the BK can attack the WP
9	blxwp	B attacks the WP (BR in direction x = -1 only)
10	bxqsq	one or more Black pieces control the queening square
11	cntxt	the WK is on an edge and not on a8
12	dsopp	the kings are in normal opposition
13	dwipd	the WK distance to intersect point is too great
14	hdchk	there is a good delay because there is a hidden check
15	katri	the BK controls the intersect point
16	mulch	B can renew the check to good advantage
17	qxmsq	the mating square is attacked in some way by the promoted WP
18	r2ar8	the BR does not have safe access to file A or rank 8
19	reskd	the WK can be reskewered via a delayed skewer
20	reskr	the BR alone can renew the skewer threat
21	rimmx	the BR can be captured safely
22	rkxwp	the BR bears on the WP (direction x = -1 only)
23	rxmsq	the BR attacks a mating square safely
24	simpl	a very simple pattern applies
25	skach	the WK can be skewered after one or more checks
26	skewr	there is a potential skewer as opposed to fork
27	skrxp	the BR can achieve a skewer or the BK attacks the WP
28	spcop	there is a special opposition pattern present
29	stlmt	the WK is in stalemate
30	thrsk	there is a skewer threat lurking
31	wkcti	the WK cannot control the intersect point
32	wkna8	the WK is on square a8
33	wknck	the WK is in check
34	wkovl	the WK is overloaded
35	wkpos	the WK is in a potential skewer position
36	wtoeg	the WK is one away from the relevant edge

Table 1: Board positions attributes [1]

3 WEKA

WEKA (Waikato Environment for Knowledge Analysis) is an open source data mining software from University od Waikato, New Zealand.

3.1 Classification

In order to predict whether the white player wins or loses from board setup, we tested several classification algorithms implemented in WEKA. Classifiers were trained using k-fold cross-validation with $k = 10$.

ZeroR is a simple classifier which ignores every attribute and simply assigns dominating class to each sample. Uneffective in this case because the distribution of win and nowin class is roughly equal (52.2215% win and 47.7785% nowin).

OneR is a classifier which selects attribute with smallest classification error and uses that attribute as its classification rule.

NaiveBayes is a classifier constructed from naive Bayes probability model.

Decision tree algorithms construct a decision tree from the attributes of training set.

Algorithm	**Success rate**
ZeroR	52.2215%
OneR	66.4581%
NaiveBayes	87.8911%
J48 (decision tree)	99.4368%
SimpleCart (decision tree)	99.3742%
RandomForest (decision tree)	98.7797%

Table 2: WEKA Classifiers

3.2 Association

To find association rules in KRKPA7 data set we used **Apriori** algorithm. Best rules found:

1. thrsk=f 3060 → spcop=f 3060
2. skach=f thrsk=f 3049 → spcop=f 3049
3. hdchk=f thrsk=f 3045 → spcop=f 3045
4. reskd=f thrsk=f 3045 → spcop=f 3045
5. skach=f 3185 → spcop=f 3184
6. hdchk=f 3181 → spcop=f 3180
7. reskd=f 3170 → spcop=f 3169
8. hdchk=f skach=f 3170 → spcop=f 3169
9. reskd=f skach=f 3159 → spcop=f 3158
10. hdchk=f reskd=f 3155 → spcop=f 3154

4 Rapid Miner

RapidMiner is a software by company RapidMiner for commercial use of machine learning, data and text mining and predictive and business analytics.

4.1 Classification

Preprocessing steps for krkpa7 data set in RapidMiner require to only specify target column. To create a classifier in RapidMiner, we layout several components together starting from data input, then going through validation step, which makes all the partitioning, classification and model performance evaluation, and finishing in the output of the pipeline.

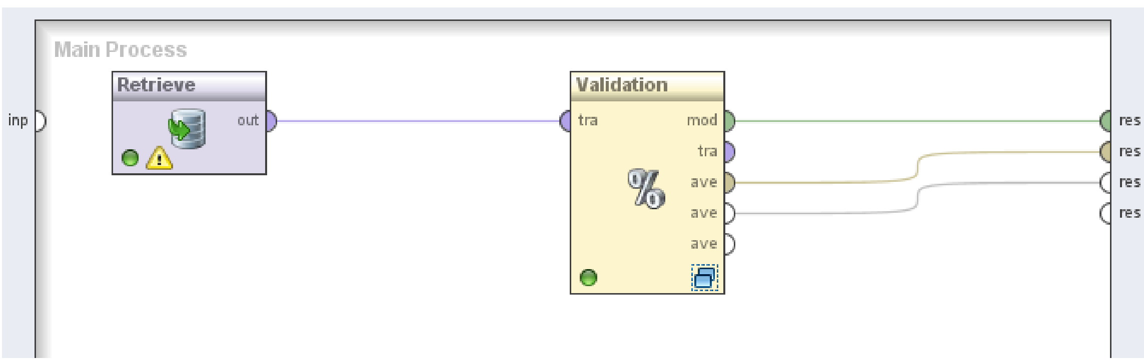

Figure 2: Pipeline setup

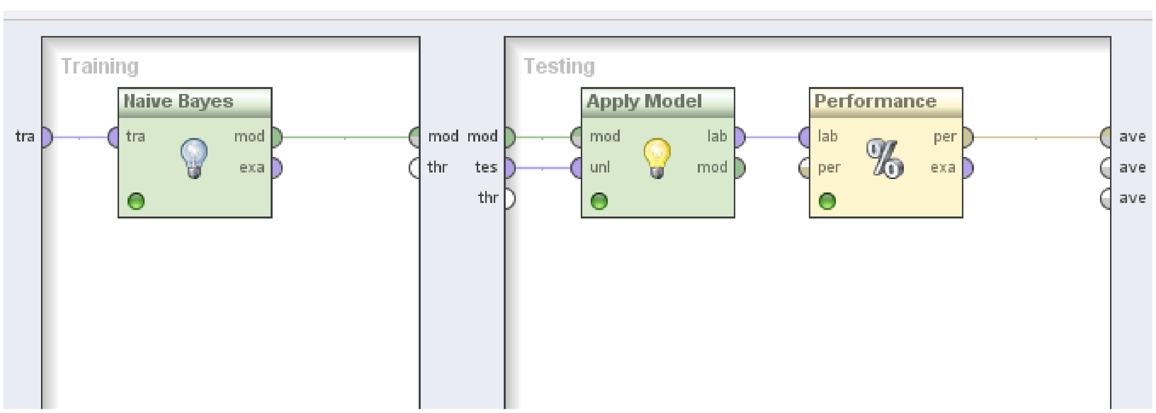

Figure 3: Naive bayes classifier component

If we use desicion tree classifier, we can get either graphical or textual representation of the resulting classifier.

```
Tree
rimmx = f
|   bxqsq = f
|   |   wknck = f
|   |   |   wkna8 = f
|   |   |   |   bkxbq = f
|   |   |   |   |   wkpos = f
|   |   |   |   |   |   katri = b
|   |   |   |   |   |   |   bkblk = f: nowin {won=0, nowin=15}
|   |   |   |   |   |   |   bkblk = t
|   |   |   |   |   |   |   |   hdchk = f: won {won=21, nowin=1}
|   |   |   |   |   |   |   |   hdchk = t: nowin {won=0, nowin=4}
|   |   |   |   |   |   katri = n
|   |   |   |   |   |   |   bkblk = f
|   |   |   |   |   |   |   |   wkcti = f: nowin {won=0, nowin=55}
|   |   |   |   |   |   |   |   wkcti = t
|   |   |   |   |   |   |   |   |   r2ar8 = f
|   |   |   |   |   |   |   |   |   |   reskr = f
|   |   |   |   |   |   |   |   |   |   |   dsopp = f: nowin {won=2, nowin=3}
|   |   |   |   |   |   |   |   |   |   |   dsopp = t: won {won=2, nowin=0}
|   |   |   |   |   |   |   |   |   |   reskr = t: nowin {won=0, nowin=6}
|   |   |   |   |   |   |   |   |   r2ar8 = t: nowin {won=0, nowin=14}
|   |   |   |   |   |   |   bkblk = t
|   |   |   |   |   |   |   |   hdchk = f: won {won=22, nowin=0}
|   |   |   |   |   |   |   |   hdchk = t: nowin {won=0, nowin=4}
|   |   |   |   |   |   katri = w: won {won=50, nowin=0}
|   |   |   |   |   wkpos = t
|   |   |   |   |   |   katri = b
|   |   |   |   |   |   |   bkblk = f: nowin {won=0, nowin=20}
|   |   |   |   |   |   |   bkblk = t
|   |   |   |   |   |   |   |   rxmsq = f: won {won=8, nowin=0}
|   |   |   |   |   |   |   |   rxmsq = t: nowin {won=1, nowin=1}
|   |   |   |   |   |   katri = n
|   |   |   |   |   |   |   rxmsq = f
|   |   |   |   |   |   |   |   dsopp = f: won {won=221, nowin=0}
|   |   |   |   |   |   |   |   dsopp = t
|   |   |   |   |   |   |   |   |   dwipd = g
|   |   |   |   |   |   |   |   |   |   skewr = f: won {won=8, nowin=0}
|   |   |   |   |   |   |   |   |   |   skewr = t
|   |   |   |   |   |   |   |   |   |   |   wtoeg = n: won {won=3, nowin=0}
|   |   |   |   |   |   |   |   |   |   |   wtoeg = t: nowin {won=0, nowin=2}
|   |   |   |   |   |   |   |   |   dwipd = l: won {won=23, nowin=0}
|   |   |   |   |   |   |   rxmsq = t
|   |   |   |   |   |   |   |   qxmsq = f: nowin {won=0, nowin=15}
|   |   |   |   |   |   |   |   qxmsq = t: won {won=27, nowin=0}
|   |   |   |   |   |   katri = w: won {won=49, nowin=0}
|   |   |   |   bkxbq = t: won {won=599, nowin=0}
|   |   |   wkna8 = t
|   |   |   |   bknwy = f: nowin {won=0, nowin=106}
|   |   |   |   bknwy = t
|   |   |   |   |   r2ar8 = f
|   |   |   |   |   |   simpl = f: won {won=3, nowin=0}
|   |   |   |   |   |   simpl = t: nowin {won=0, nowin=5}
|   |   |   |   |   r2ar8 = t: nowin {won=0, nowin=9}
|   |   wknck = t
|   |   |   r2ar8 = f
|   |   |   |   bkxcr = f
|   |   |   |   |   skrxp = f
|   |   |   |   |   |   mulch = f
|   |   |   |   |   |   |   thrsk = f
|   |   |   |   |   |   |   |   bkona = f
|   |   |   |   |   |   |   |   |   reskr = f
|   |   |   |   |   |   |   |   |   |   blxwp = f
|   |   |   |   |   |   |   |   |   |   |   bkon8 = f
|   |   |   |   |   |   |   |   |   |   |   |   skach = f: won {won=38, nowin=0}
|   |   |   |   |   |   |   |   |   |   |   |   skach = t: nowin {won=0, nowin=2}
|   |   |   |   |   |   |   |   |   |   |   bkon8 = t: nowin {won=0, nowin=2}
|   |   |   |   |   |   |   |   |   |   blxwp = t: nowin {won=0, nowin=3}
|   |   |   |   |   |   |   |   |   reskr = t: nowin {won=0, nowin=6}
|   |   |   |   |   |   |   |   bkona = t: nowin {won=0, nowin=6}
|   |   |   |   |   |   |   thrsk = t: nowin {won=0, nowin=7}
|   |   |   |   |   |   mulch = t: nowin {won=0, nowin=18}
|   |   |   |   |   skrxp = t: nowin {won=0, nowin=34}
|   |   |   |   bkxcr = t: nowin {won=0, nowin=77}
|   |   |   r2ar8 = t
|   |   |   |   wkovl = f
|   |   |   |   |   bkxcr = f
|   |   |   |   |   |   bkona = f
|   |   |   |   |   |   |   mulch = f
|   |   |   |   |   |   |   |   bkon8 = f: won {won=8, nowin=0}
|   |   |   |   |   |   |   |   bkon8 = t: nowin {won=0, nowin=3}
|   |   |   |   |   |   |   mulch = t: nowin {won=0, nowin=11}
|   |   |   |   |   |   bkona = t: nowin {won=0, nowin=19}
|   |   |   |   |   bkxcr = t: nowin {won=0, nowin=41}
|   |   |   |   wkovl = t: nowin {won=0, nowin=295}
|   bxqsq = t: nowin {won=0, nowin=743}
rimmx = t: won {won=584, nowin=0}
```


Figure 4: Decision tree of KRKPA7 in RapidMiner

5 IBM SPSS Modeler

Originally named Clementine, also known as SPSS Clementine, PASW Modeler is a data mining and text analytics software by IBM.

5.1 Classification

In the pipeline schema we first retrieve data from a file. Unlike in WEKA and RapidMiner, we need to provide a csv file instead of more suitable arff file, which contains more detailed definitions of data types. We run the data through auto data preparation step and feed the result to feature selector. After the selector is trained in the first run of the pipeline, we use the resulting component in the next step of pipeline and feed the refined data to a partition step so that we get two sets used for training and testing of the upcoming models. In this case we used c5 decision tree and neural network classifiers. Same as in the feature selector component, we need first to build the models and use the newly created components down in the pipeline. To evaluate results we feed the outputs from the classifiers to error matrices for each of the models.

class	**nowin**	**win**
nowin	1510	17
won	95	1574

Table 3: Error matrix for c5 decision tree in PASW

class	**nowin**	**win**
nowin	1495	32
won	83	1586

Table 4: Error matrix for neural network in PASW

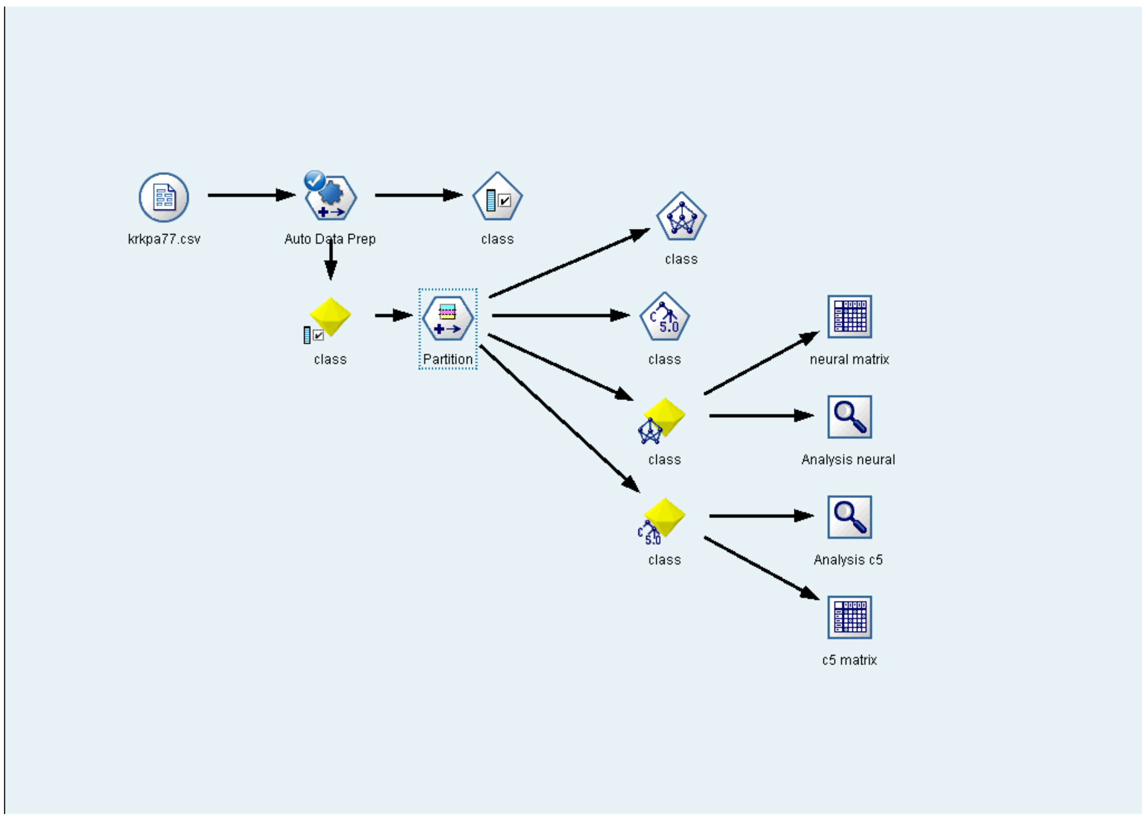

Figure 5: Pipeline setup of classifiers in PASW

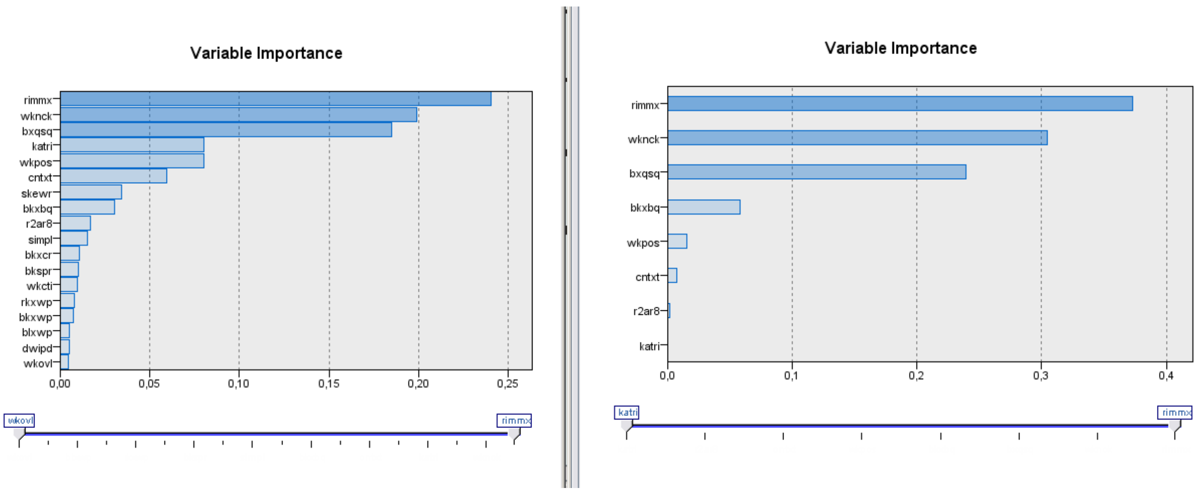

Figure 6: Result of PASW Models

```
Results for output field  class
Comparing $C- class  with  class
'Partition'             1_Training            2_Testing
Correct                      1 450   96,09%       1
634   96,86%
Wrong                           59    3,91%          53     3,14%
Total                        1 509                1 687
```

Figure 7: Analysis of c5 decision tree in PASW

```
Results for output field  class
Comparing $N- class  with  class
'Partition'             1_Training            2_Testing
Correct                      1 457   96,55%       1
624   96,27%
Wrong                           52    3,45%          63     3,73%
Total                        1 509                1 687
```

Figure 8: Analysis of neural network in PASW

6 Enterprise Modeler

References

[1] MICHIE, D. Consciousness as an Engineering Issue, Part. *Journal of Consciousness Studies*. 1995, 2, 1, s. 52–66.