Data mining: concepts and algorithms

Exam – Data mining

Name: Matteo Surname: Orlando

Student ID: 277013 Dataset: ecoli

Objective

Exploit data mining algorithms to analyze a real dataset using the RapidMiner machine learning tool.

Dataset

Download the zip http://dbdmg.polito.it/wordpress/wp-content/uploads/2020/01/DatasetsUCI.zip and select **the assigned dataset**. The dataset has a class attribute that will be used during the first part of the exam as label for the classification problem. If needed, a brief description of each dataset is available in the zip folder together with the dataset. The examination test is composed of two parts: the first one focuses on the classification problem and the second one the clustering problem.

Part I: Classification problem

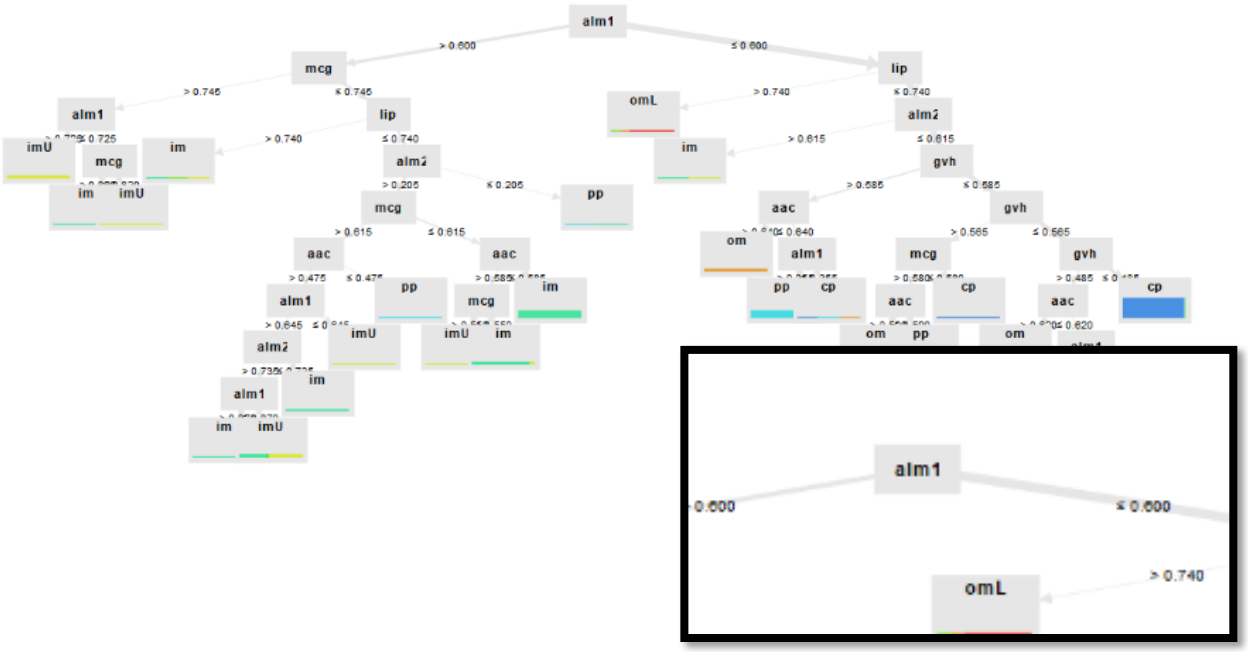
Analysts want to predict the class of new data, according to the already classified data. To this purpose, you must exploit two different classification algorithms: a decision tree (Decision Tree) and a distance-based classifier (K-NN) to create classification models based on the given dataset and give an answer to a set of questions in order to understand what is the best classification algorithm, in terms of accuracy, for the given problem/dataset.

To evaluate classification performance, different configuration settings have to be tested and compared with each other. A 10-fold Stratified Cross-Validation process must be used to validate classifier accuracy.

Questions

Answer to the following questions:

1. Learn a Decision Tree using the whole dataset as training data and the default configuration setting for algorithm Decision Tree. Which attribute is deemed to be the most discriminative one for service class prediction?
    - Answer:
      alm1 is the most discriminative attribute



2. Use a 10-fold Stratified Cross-Validation approach to validate the accuracy of the generated classification model. What is the impact of the **minimal leaf size** parameter on the average accuracy achieved by the generated Decision Tree? Compare the confusion matrices achieved using different parameter settings (keep the default configuration for all the other parameters). Report the accuracy achieved for at least three values of the **minimal leaf size** parameter.
    - Answer:
      The overall accuracy decreases when increasing the minimum leaf size (MLS). Moreover when the MLS increase we can notice how the class recall for *cp*,*im* and *imU* remains quite unchanged. We can also notice how when increasing the MLS the decision tree start to "collapse" towards the calls **cp** whose classification precision decrease.

Min leaf size:1 accuracy: 85.13% +/- 6.80% (micro average: 85.12%)

	true cp	true pp	true im	true imS	true imL	true imU	true om	true omL	class precision
pred. cp	139	5	3	0	0	1	1	0	93.29%
pred. pp	2	43	3	1	0	2	2	0	81.13%
pred. im	0	1	64	1	1	12	0	0	81.01%
pred. imS	0	0	0	0	0	0	0	0	0.00%
pred. imL	0	0	0	0	0	0	0	0	0.00%
pred. imU	0	2	5	0	0	19	0	0	73.08%
pred. om	2	1	1	0	0	0	16	0	80.00%
pred. omL	0	0	1	0	1	1	1	5	55.56%
class recall	97.20%	82.69%	83.12%	0.00%	0.00%	54.29%	80.00%	100.00%	

Min leaf size:2 accuracy: 80.08% +/- 9.09% (micro average: 80.06%)

	true cp	true pp	true im	true imS	true imL	true imU	true om	true omL	class precision
pred. cp	139	14	3	1	0	1	7	0	84.24%
pred. pp	3	33	2	0	0	2	3	0	76.74%
pred. im	1	2	65	1	1	13	0	0	78.31%
pred. imS	0	0	0	0	0	0	0	0	0.00%
pred. imL	0	0	0	0	0	0	0	0	0.00%
pred. imU	0	1	6	0	0	18	0	0	72.00%
pred. om	0	2	0	0	0	0	9	0	81.82%
pred. omL	0	0	1	0	1	1	1	5	55.56%
class recall	97.20%	63.46%	84.42%	0.00%	0.00%	51.43%	45.00%	100.00%	

Min leaf size: 4 accuracy: 78.00% +/- 7.08% (micro average: 77.98%)

	true cp	true pp	true im	true imS	true imL	true imU	true om	true omL	class precision
pred. cp	142	20	4	1	1	2	9	1	78.89%
pred. pp	1	27	4	0	0	0	3	0	77.14%
pred. im	0	4	63	1	1	13	0	0	76.83%
pred. imS	0	0	0	0	0	0	0	0	0.00%
pred. imL	0	0	0	0	0	0	0	0	0.00%
pred. imU	0	0	5	0	0	19	0	0	79.17%
pred. om	0	1	0	0	0	0	7	0	87.50%
pred. omL	0	0	1	0	0	1	1	4	57.14%
class recall	99.30%	51.92%	81.82%	0.00%	0.00%	54.29%	35.00%	80.00%	

Min leaf size:8 accuracy: 70.22% +/- 5.82% (micro average: 70.24%)

	true cp	true pp	true im	true imS	true imL	true imU	true om	true omL	class precision
pred. cp	143	41	7	1	1	2	16	5	66.20%
pred. pp	0	7	0	0	0	0	0	0	100.00%
pred. im	0	3	67	1	1	16	2	0	74.44%
pred. imS	0	0	0	0	0	0	0	0	0.00%
pred. imL	0	0	0	0	0	0	0	0	0.00%
pred. imU	0	0	3	0	0	17	0	0	85.00%
pred. om	0	1	0	0	0	0	2	0	66.67%
pred. omL	0	0	0	0	0	0	0	0	0.00%
class recall	100.00%	13.46%	87.01%	0.00%	0.00%	48.57%	10.00%	0.00%	

3. Considerer the K-Nearest Neighbor (K-NN) classification algorithm and use a 10-fold Stratified Cross-Validation approach to validate the accuracy of the generated classification models. What is the impact of parameter **k** on the classifier performance (i.e., accuracy)? Compare the confusion matrices and the accuracy achieved using different values of **k**. Report the accuracy achieved for at least three values of **k**. Does K-NN perform on average better or worse than Decision Tree on the analyzed data?
    - Answer: We can notice how that the accuracy seems to increase when k increase and probably it would reach an optimum when 5<k<15. We can notice how with the exclusion of the class omL the class precision and the recall are quite stable when changing the value of k. We can also notice how the classes with high number of elements are the ones with the best values of precision and recall. However the class om despite the low amount of elements with respect to other classes reaches

always good performance values, so maybe we can assume that there is a feature that has a strong importance in identifying element of this class (the decision tree seems to indicate the feature is *aac*) With respect with the decision tree we can notice that k-nn has in general better performance than the decision trees . However neither of the 2 algorithm was able to correctly identify the smallest classes (i.e. imS, imL) and both of them had low performance for the class imU

K=2 accuracy: 81.56% +/- 3.81% (micro average: 81.55%)

	true cp	true pp	true im	true imS	true imL	true imU	true om	true omL	class precision
pred. cp	134	7	4	0	0	1	0	0	91.78%
pred. pp	5	42	1	1	0	1	3	0	79.25%
pred. im	4	1	56	0	1	10	0	0	77.78%
pred. imS	0	0	1	0	0	1	0	0	0.00%
pred. imL	0	0	1	0	0	1	0	0	0.00%
pred. imU	0	0	14	1	0	21	0	0	58.33%
pred. om	0	2	0	0	0	0	16	0	88.89%
pred. omL	0	0	0	0	1	0	1	5	71.43%
class recall	93.71%	80.77%	72.73%	0.00%	0.00%	60.00%	80.00%	100.00%	

K=5 accuracy: 86.30% +/- 3.20% (micro average: 86.31%)

	true cp	true pp	true im	true imS	true imL	true imU	true om	true omL	class precision
pred. cp	141	3	5	0	0	1	0	0	94.00%
pred. pp	2	46	1	1	0	0	2	0	88.46%
pred. im	0	1	60	1	0	12	0	0	81.08%
pred. imS	0	0	0	0	0	0	0	0	0.00%
pred. imL	0	0	0	0	0	0	0	0	0.00%
pred. imU	0	0	10	0	0	21	0	0	67.74%
pred. om	0	2	0	0	0	0	17	0	89.47%
pred. omL	0	0	1	0	2	1	1	5	50.00%
class recall	98.60%	88.46%	77.92%	0.00%	0.00%	60.00%	85.00%	100.00%	

K=10 accuracy: 86.93% +/- 5.21% (micro average: 86.90%)

	true cp	true pp	true im	true imS	true imL	true imU	true om	true omL	class precision
pred. cp	141	4	6	0	0	1	0	0	92.76%
pred. pp	2	47	1	1	0	0	2	0	88.68%
pred. im	0	1	60	1	0	11	0	0	82.19%
pred. imS	0	0	0	0	0	0	0	0	0.00%
pred. imL	0	0	0	0	0	0	0	0	0.00%
pred. imU	0	0	9	0	0	22	0	0	70.97%
pred. om	0	0	0	0	0	0	17	0	100.00%
pred. omL	0	0	1	0	2	1	1	5	50.00%
class recall	98.60%	90.38%	77.92%	0.00%	0.00%	62.86%	85.00%	100.00%	

k=15 accuracy: 85.41% +/- 3.59% (micro average: 85.42%)

	true cp	true pp	true im	true imS	true imL	true imU	true om	true omL	class precision
pred. cp	141	4	8	0	0	1	0	0	91.56%
pred. pp	2	47	1	1	1	0	2	3	82.46%
pred. im	0	1	57	1	1	11	0	0	80.28%
pred. imS	0	0	0	0	0	0	0	0	0.00%
pred. imL	0	0	0	0	0	0	0	0	0.00%
pred. imU	0	0	11	0	0	23	0	0	67.65%
pred. om	0	0	0	0	0	0	18	1	94.74%
pred. omL	0	0	0	0	0	0	0	1	100.00%
class recall	98.60%	90.38%	74.03%	0.00%	0.00%	65.71%	90.00%	20.00%	

Part II: Clustering problem

Goal

Analysts want to split the available data objects in groups. To this purpose, you must exploit two different clustering algorithms: a k-Means clustering algorithm (**K-Means**) and a density-based algorithm (**DBScan**) and select the most appropriate one for the given problem. Consider **only numerical attributes** for the clustering task.

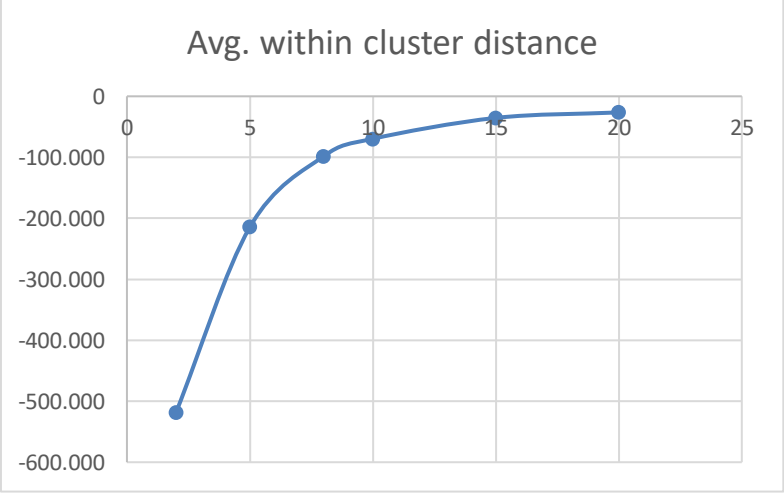
Questions

Answer to the following questions:

1. Apply the **k-Means** algorithm to cluster the given data and analyze the impact of parameter **k** (number of generated cluster) on the generated clusters. More specifically, perform an empirical analysis by using the average within cluster distance measure to evaluate the impact of the value of k on the quality (in terms of Cluster Cohesion) of the generated clusters. What is the impact of k on the cluster cohesion (average within cluster distance)?
    - Answer:

      We can notice how increasing k improves the performance, however as also the plot suggest the improvement we obtain reduces its magnitude. What it seems stable is the percentual improvements (it seems that doubling k improves the performance of around 30%)

K	Avg. within cluster distance
2	-517.969
5	-214.308
8	-98.253
10	-69.566
15	-35.390
20	-26.394



2. Consider the **DBScan algorithm** (a density based algorithm) and compare its performance with that of the **k-Means** algorithm. What is the impact of parameter **min points** on the performance of DBScan? Does **DBScan** perform better than **k-Means** (in terms of average within cluster distance)?
    - Answer:
      Increasing ***min_points*** brings and improvement on the average cluster distance however the algorithm seems to reach an optimum around ***min_points***=10. We can notice how in general DBScan has worse performance w.r.t. k-Means in terms of average cluster within distance.

Min points	Avg. within cluster distance
2	-753.174
5	-689.901
8	-309.184
10	-305.907
16	-310.285
20	-341.660