QMAK: Interacting with Machine Learning Models and Visualizing Classification Process

Arkadiusz Wojna, Katarzyna Jachim, Łukasz Kosson, Łukasz Kowalski, Damian Mański, Michał Mański, Krzysztof Mroczek, Krzysztof Niemkiewicz, Robert Piszczatowski, Maciej Próchniak, Tomasz Romańczuk, Piotr Skibiński, Marcin Staszczyk, Michał Szostakiewicz, Leszek Tur, Damian Wójcik, and Maciej Zuchniak

Faculty of Mathematics, Informatics and Mechanics, University of Warsaw
wojna@mimuw.edu.pl

Abstract. In various classification problems beside high accuracy data analysts expect often understanding and certain insight into the process of classification. To help them understand why a trained model selects a particular decision, how much it is sure of the assigned decision, and to enable interactive improvement of trained models we present Qmak. The tool visualizes not only classification models but also the processes classifying individual objects. Five classical machine learning models and their classification process are visualized with Qmak: neural network, decision tree, k nearest neighbors, classifier based on principal component analysis (PCA) and rule-based classifier. Qmak provides also exemplary functions enabling users to modify trained models interactively.

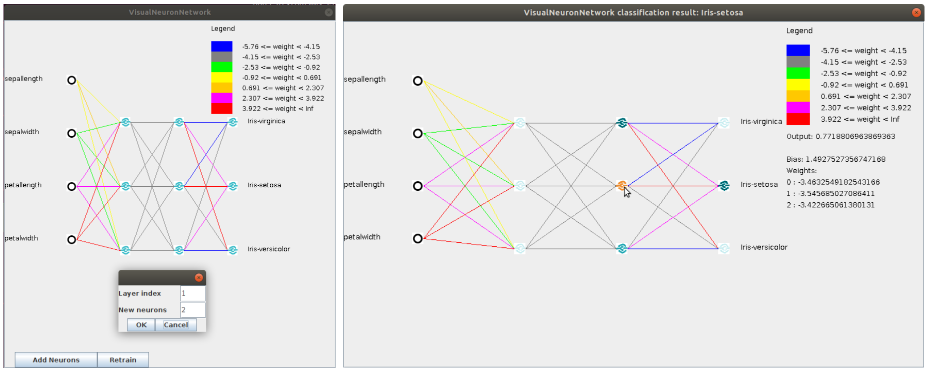
Keywords: explainable machine learning, classification visualization, interactive classifier.

1 Introduction

As the field of machine learning matured, beside a predicted value data analysts started to expect some insight into how a trained model made a decision. They expect explanation why decision was selected and how much the model is sure of its decision. Understanding why classification models classify incorrectly some cases can help both authors and analysts improve their models. Yet further step it is to provide the ability of modifying a trained model in an interactive process.

There are many visualization tools helpful in data mining but most of them use various charts to visualize data and classification results. Weka [1] and Graphviz can visualize classification models based on a graph or tree structure but only the models having one of these two structures can be visualized in this way. The tools visualizing classification models reflecting their specific structure are dedicated usually to a single model.

Qmak is a visualization and interactive platform gathering different classification models. It provides a framework not only for visualization and interaction with models but also for visualization of the classification process. It allows users to compare both the structure of different models and how they differ in the



Legend

🟦	-5.76 <= weight < -4.15
⬜	-4.15 <= weight < -2.53
🟩	-2.53 <= weight < -0.92
🟨	-0.92 <= weight < 0.691
🟧	0.691 <= weight < 2.307
🟪	2.307 <= weight < 3.922
🟥	3.922 <= weight < Inf

Layer index 1
New neurons 2

```
Output: 0.7718806963869363

Bias: 1.4927527356747168
Weights:
0 : -3.4632549182543166
1 : -3.545685027086411
2 : -3.422665061380131
```

Fig. 1. Visualization of neural network (left) and its classification process (right)

classification processes provided the same object to be classified. Qmak implements visualization of popular machine learning models and gives examples of how users may interact with models to improve their classification accuracy. The platform can be used also as a didactic tool during machine learning courses.

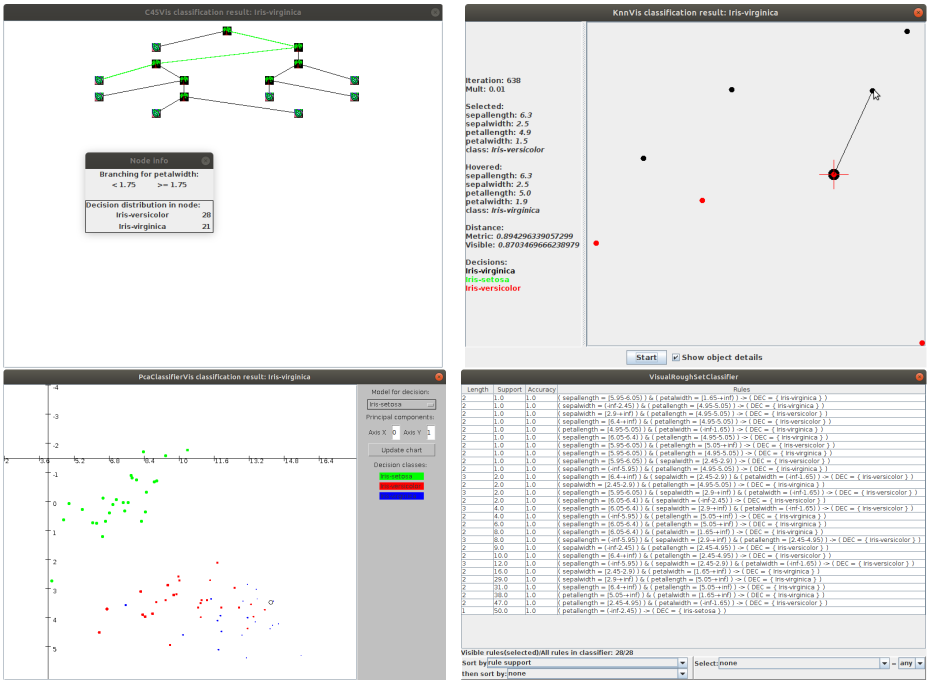
2 System Overview

Qmak is a graphical tool providing the following features: visualization of data, classifiers and single object classification, interactive classifier modification by a user, classification of test data with presentation of misclassified objects and experiments comparing classification accuracy of classifiers with various test types. The tool and its demo are available at `http://rseslib.mimuw.edu.pl/qmak`.

Qmak uses Rseslib library [2,3] as the source of classification models. The version 3.2.1 provides visualization of 5 classifiers: neural network, decision tree, k nearest neighbors, classifier based on principal component analysis (PCA) and rule-based classifier. Users can implement new classifiers and their visualization and add them easily to Qmak. It does not require any change in Qmak itself. A new classifier can be added using menu or in the configuration file.

Neural network in Qmak is trained with the classical backpropagation algorithm and sigmoid activation functions. Visualization of a neural network presents the neurons and the connections between them (see Figure 1). The neurons from the last layer correspond to decisions. The color of a connection represents its weight. A user can select a neuron to display the exact weights of its input connections and its bias. One can modify the network by adding new neurons in hidden layers and retraining the network. Visualization of classification presents also the strength of the output signal from each neuron with intensity of its color and the exact value of the signal when a node is selected.

Decision tree is visualized by presenting the structure of the tree (see Figure 2). After selection of a node the decision distribution of its training objects is displayed and the branching condition for an inner node or the assigned decision



Branching for petalwidth:
< 1.75 >= 1.75

Decision distribution in node:

Iris-versicolor	28
Iris-virginica	21

Iteration: 638
Mult: 0.01

Selected:
sepallength: *6.3*
sepalwidth: *2.5*
petallength: *4.9*
petalwidth: *1.5*
class: *Iris-versicolor*

Hovered:
sepallength: *6.3*
sepalwidth: *2.5*
petallength: *5.0*
petalwidth: *1.9*
class: *Iris-virginica*

Distance:
Metric: *0.894296339057299*
Visible: *0.8703469666238979*

Decisions:
Iris-virginica
Iris-setosa
Iris-versicolor

Model for decision: Iris-setosa
Principal components:
Axis X 0 Axis Y 1

Length	Support	Accuracy	Rules
2	1.0	1.0	(sepallength = [5.95-6.05)) & (petalwidth = [1.65-+inf)) -> (DEC = { Iris-virginica })
2	1.0	1.0	(sepalwidth = (-inf-2.45)) & (petallength = [4.95-5.05)) -> (DEC = { Iris-virginica })
2	1.0	1.0	(sepalwidth = [2.9-+inf)) & (petallength = [4.95-5.05)) -> (DEC = { Iris-versicolor })
2	1.0	1.0	(sepallength = [6.4-+inf)) & (petallength = [4.95-5.05)) -> (DEC = { Iris-versicolor })
2	1.0	1.0	(petallength = [4.95-5.05)) & (petalwidth = (-inf-1.65)) -> (DEC = { Iris-virginica })
2	1.0	1.0	(sepallength = [6.05-6.4)) & (petallength = [4.95-5.05)) -> (DEC = { Iris-virginica })
2	1.0	1.0	(sepallength = [5.95-6.05)) & (petallength = [5.05-+inf)) -> (DEC = { Iris-versicolor })
2	1.0	1.0	(sepallength = [5.95-6.05)) & (petallength = [4.95-5.05)) -> (DEC = { Iris-virginica })
2	1.0	1.0	(sepallength = [5.95-6.05)) & (sepalwidth = [2.45-2.9)) -> (DEC = { Iris-versicolor })
2	1.0	1.0	(sepallength = (-inf-5.95)) & (petallength = [4.95-5.05)) -> (DEC = { Iris-virginica })
3	2.0	1.0	(sepallength = [6.4-+inf)) & (sepalwidth = [2.45-2.9)) & (petalwidth = (-inf-1.65)) -> (DEC = { Iris-versicolor })
2	2.0	1.0	(sepalwidth = [2.45-2.9)) & (petallength = [4.95-5.05)) -> (DEC = { Iris-virginica })
3	2.0	1.0	(sepallength = [5.95-6.05)) & (sepalwidth = [2.9-+inf)) & (petalwidth = (-inf-1.65)) -> (DEC = { Iris-versicolor })
2	2.0	1.0	(sepallength = [6.05-6.4)) & (sepalwidth = (-inf-2.45)) -> (DEC = { Iris-versicolor })
3	4.0	1.0	(sepallength = [6.05-6.4)) & (sepalwidth = [2.9-+inf)) & (petalwidth = (-inf-1.65)) -> (DEC = { Iris-versicolor })
2	4.0	1.0	(sepallength = (-inf-5.95)) & (petallength = [5.05-+inf)) -> (DEC = { Iris-virginica })
2	6.0	1.0	(sepallength = [6.05-6.4)) & (petallength = [5.05-+inf)) -> (DEC = { Iris-virginica })
2	8.0	1.0	(sepallength = [6.05-6.4)) & (petalwidth = [1.65-+inf)) -> (DEC = { Iris-virginica })
3	8.0	1.0	(sepallength = (-inf-5.95)) & (sepalwidth = [2.9-+inf)) & (petallength = [2.45-4.95)) -> (DEC = { Iris-versicolor })
2	9.0	1.0	(sepalwidth = (-inf-2.45)) & (petallength = [2.45-4.95)) -> (DEC = { Iris-versicolor })
2	10.0	1.0	(sepallength = [6.4-+inf)) & (petallength = [2.45-4.95)) -> (DEC = { Iris-versicolor })
3	12.0	1.0	(sepallength = (-inf-5.95)) & (sepalwidth = [2.45-2.9)) & (petalwidth = (-inf-1.65)) -> (DEC = { Iris-versicolor })
2	16.0	1.0	(sepalwidth = [2.45-2.9)) & (petalwidth = [1.65-+inf)) -> (DEC = { Iris-virginica })
2	29.0	1.0	(sepalwidth = [2.9-+inf)) & (petallength = [5.05-+inf)) -> (DEC = { Iris-virginica })
2	31.0	1.0	(sepallength = [6.4-+inf)) & (petallength = [5.05-+inf)) -> (DEC = { Iris-virginica })
2	38.0	1.0	(petallength = [5.05-+inf)) & (petalwidth = [1.65-+inf)) -> (DEC = { Iris-virginica })
2	47.0	1.0	(petallength = [2.45-4.95)) & (petalwidth = (-inf-1.65)) -> (DEC = { Iris-versicolor })
1	50.0	1.0	(petallength = (-inf-2.45)) -> (DEC = { Iris-setosa })

Visible rules(selected)/All rules in classifier: 28/28
Sort by rule support
then sort by: none
Select: none = any

Fig. 2. Decision tree (upper left), k nearest neighbors (upper right), PCA classifier (lower left) and rule-based classifier (lower right) in Qmak

for a leaf. A user can cut off the subtree of any inner node and convert the node to a leaf. Visualization of classification presents a decision tree with the path from the root to a leaf corresponding to a classified object highlighted in green.

K nearest neighbors classifier projects all training objects on 2-dimensional area of the classifier window using different colors for different decisions. It displays the process of searching for placement of the objects reflecting best the true distances between them in the induced metric. A user can select one object and hover another one to display their attribute values and the true distance between them. Visualization of classification by a k-nn classifier projects the object to be classified and its k nearest neighbors also trying to reflect the true distances between them. As in model visualization neighbors from different decision classes have different colors and Qmak shows their attribute values and the distances between them.

Classifier based on principal component analysis (PCA) projects all training objects on the plane spanned by a pair of principal components of the model for one of decision classes. Different colors are used for objects with different decisions. The objects close to the plane are represented with bigger dots, the more distant the objects are from the plane the smaller dots represent them. Selection of the model and the principal components used to project objects can be changed by a user. Visualization of classification by a PCA classifier marks additionally the projection of the object to be classified.

Visualization of a rule-based classifier presents its decision rules with their length, support and accuracy. A user can filter and sort the rules by attribute occurrence, attribute values, rule length, support and accuracy. Visualization of classification shows the rules matching a classified object enabling the same types of filtering and sorting criteria as visualization of the classifier.

3 System Usage

Qmak can help users understand why a particular decision was selected. For example, the classification path for an object in a decision tree shows the attributes and the conditions on these attributes determining the decision. K nearest neighbors model shows the training objects identified as the most similar to a classified object and used to select the decision. Rule-based classifier shows the decision rules matching a classified object.

Using Qmak users can also find out how much a classifier is sure of the assigned decision. In a neural network it is indicated by the difference between the strength of the signals from the winning neuron and from other output neurons. In k nearest neighbors one can look at the proportion of the nearest neighbors with the winning decision to the neighbors with other decisions.

Users can also find out which element of a classifier needs to be improved and in some cases they can modify interactively the existing classifier. Qmak implements some examples of such interaction. Similar weights of the connections between pair of layers in a neural network like between the hidden layers in Figure 1 may indicate the preceding of the two layers was given too few neurons. A user can either train a new network with a different structure or add new neurons to hidden layers of the existing network and retrain it. If a branch in a decison tree misclassifies test objects the branch can be the result of overfitting to the training set. A user can prune such a branch in Qmak. In a k nereast neighbors model one can see whether another number of neighbors or another method of voting by neighbors will fix classification of misclassified objects. The number of neighbors and the voting method can be changed without retraining the classifier. One can also recognize that many misclassifications of a k-nn classifier is caused by the same small set of training objects. A user can consider removal of such objects from the training set.

References

1. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witen, I.: The weka data mining software: An update. SIGKDD Explorations 11(1), 10–18 (2009)
2. Wojna, A., Latkowski, R.: Rseslib 3: Library of rough set and machine learning methods with extensible architecture. LNCS Transactions on Rough Sets XXI 10810, 301–323 (2019)
3. Wojna, A., Latkowski, R., Kowalski, Ł.: RSESLIB: User Guide, `http://rseslib.mimuw.edu.pl/rseslib.pdf`