Chapter 7
Rule Induction

Peter Flach* and Nada Lavrač**
*University of Bristol, UK and **Jožef Stefan Institute, Slovenia

7.1. Introduction

Machine learning is an important form of intelligent data analysis. It is common in machine learning to distinguish symbolic and sub-symbolic approaches. Symbolic approaches employ some kind of description language in which the learned knowledge is expressed. The machine learning methods described in this chapter construct explicit symbolic classification rules that generalise the training cases, and are thus instances of symbolic machine learning. This area of machine learning is called *classification rule induction* [376, 378, 381]. Non-symbolic learning approaches are covered in other chapters. One of the main attractions of rule induction is that the rules are much more transparent and easier to interpret than, say, a regression model or a trained neural network.

The classification rule learning task can be defined as follows: Given a set of training examples (instances for which the classification is known), find a set of classification rules that can be used for prediction or classification of new instances, i.e., cases that haven't been presented to the learner before. A more formal definition of the classification rule learning task has to take into account the restrictions imposed by the language used to describe the data (data description language) and the language used to describe the induced set of rules (hypothesis description language). The *language bias* refers to the restrictions imposed by the languages defining the format and scope of data and knowledge representation.

A generic classification rule learning task can now be defined, given a binary classification problem of classifying instances into classes named *positive* and *negative*. The task of learning a set of rules defining the class *positive* is defined as follows.

Given:

- a data description language, imposing a bias on the form of data
- training examples, i.e., a set of classified instances described in the data description language
- a hypothesis language, imposing a bias on the form of induced rules
- a coverage function, defining when an instance is covered by a rule

Find:

- a hypothesis as a set of rules described in the hypothesis language, that is
  - consistent, i.e., does not cover any negative example
  - complete, i.e., covers all positive examples

This definition distinguishes between the terms *examples* and *instances*. Examples usually refer to instances labelled by a class label, where instances themselves bear no class label. This is how we will use the two terms in this chapter. Another term appearing in the definition is *hypothesis*, used to denote the output of learning. Due to the hypothetical nature of induction, the output of inductive learning can be falsified by new evidence presented to the learner. In this chapter, the induced hypotheses will always be represented as sets of rules.

Generally speaking, a rule is an expression of the form $Head \leftarrow Body$, where $Body$ describes the conditions under which the rule 'fires', and $Head$ is typically a class label. (In the simplified learning setting above we learn rules for only one class, so there the head of the rule is strictly speaking redundant.) An instance is *covered* by such a rule if it satisfies the conditions in $Body$. An example can be either correctly covered (if it is covered and the class label in $Head$ corresponds with the class label of the example), incorrectly covered ($Head$ assigns a different class), or not covered.

A simple extension of this definition allows us to deal with multi-class learning problems. Suppose that training instances are labelled with three class labels: c_1, c_2, and c_3. The above definition of the learning task can be applied if we form three different learning tasks. In the first task, instances labelled with class c_1 are treated as the positive examples, and instances labelled c_2 and c_3 are the negative examples. In the next run, class c_2 will be considered as the positive class, and finally, in the third run, rules for class c_3 will be learned. Due to this simple transformation of a multi-class learning problem into several binary classification problems, the rest of this chapter will mostly deal with binary classification tasks, unless explicitly stated otherwise. Notice that this procedure could also be applied in the case of binary classification problems to learn an explicit definition of both the positive and the negative class.

Generally speaking, consistency and completeness – as required in the above task definition – are very strict conditions. They are unrealistic in learning from large datasets which may contain errors. *Noise* refers to random errors in the data, either due to incorrect class labels or errors in instance descriptions. It is also possible that the hypothesis language is not expressive enough to allow a complete and consistent hypothesis, in which case the target class needs to