Laboratory 3 – Decision rules

1. Decision rules are the most popular symbolic representation of knowledge derived from data. It is a natural and easy form of representation, which allows possible inspection by human and their interpretation. It is more comprehensive than any other knowledge representation.
2. Standard form of rule is: *if P then Q*, where P is the conditional part of the rule and Q is the decision part of the rule. Conditional part P of a rule is a conjunction of elementary conditions and is represented in the form of: P=(condition_1)˅...˅(condition_l), where l is the number of conditions known as the length of the rule. A single elementary condition_i (selector) is represented as: at_i rel v_i, where at_i is a conditional attribute i, v_i is a value from the domain of attribute at_i and rel is a relation operator from the set of relations {=,≠,<,≤,>,≥,in}.
3. Rule covers an example when attributes of the example match the rule's conditions. Rules can cover both positive and negative examples. Rule is discriminant or certain, when it covers only positive examples (no negative examples covered). Thanks to this the rule distinguishes examples belonging to the class indicated by the rule's decision part. A discriminant rule is minimal, if removing of one of its selectors results in negative examples being covered.
4. Typical algorithms, which induce decision rules, are based on the scheme of a sequential covering and heuristically generate a minimal set of rule covering examples. A strategy for generating a rule set directly from data examples in each turn find a rule set that covers all examples for each decision class. The main procedure is iteratively repeated for each value of the decision class. For a given class it sequentially creates rules that, in the end, cover all positive examples without covering negative examples. Each of the rule covers some of the positive examples (next these examples are skipped from consideration for the next rules). Each rule is created in a stepwise way from general to specific guided by the evaluation measures.

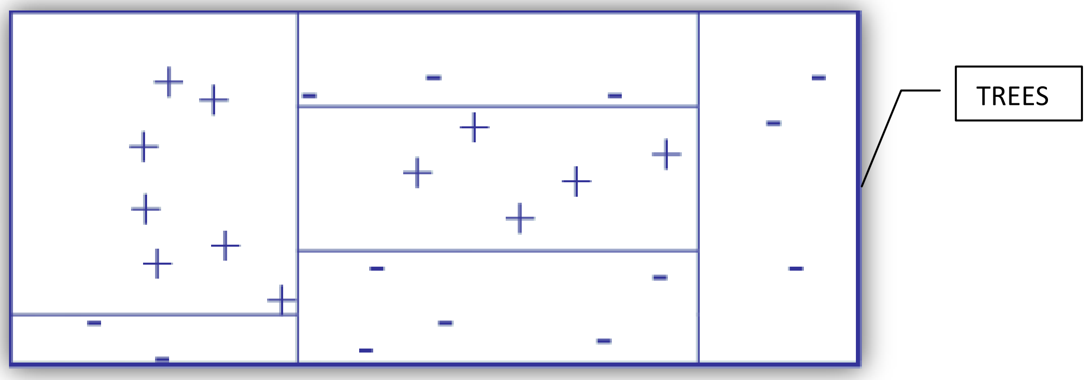
Algorithm Sequential Covering algorithm

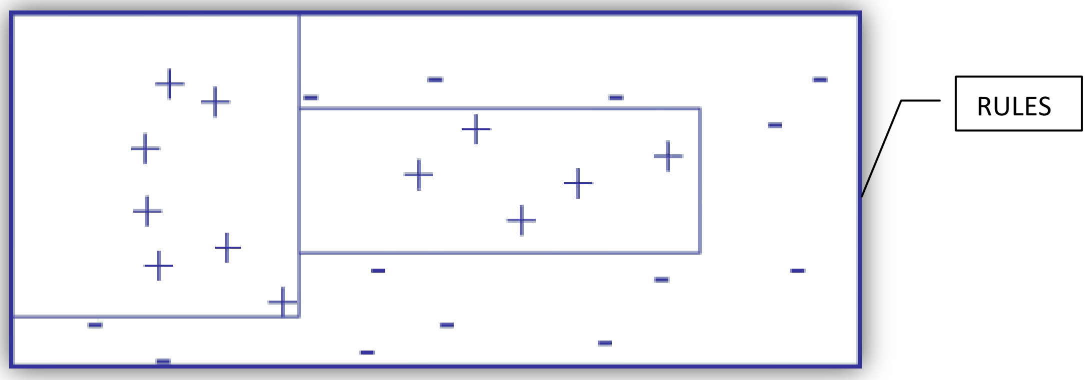
Input : U—a set of learning examples;
A—conditional attributes
Output: RS—a set of induced rules

$RS = \emptyset$;
foreach *different* $concept_i$ **do**
  $U_i = U$;
  while *all examples for* $concept_i$ *from* U_i *are not covered* **do**
    $r = LearnSingleRule(concept_i, A, U_i)$;
    $RS = RS \cup r$;
    $U_i = U_i \setminus [RS]$;
Return RS

5. Induced set of decision rules can be used for classification of new incoming examples. It is based on matching the description of the new object to the conditional part of a decision rule. Two main matching types can be distinguished:
   - full matching - all elementary conditions of a rule match the example's attributes
     - unique matching - matching to one or more rules from the same class
     - multiple matching - matching more rules from different classes
   - partial matching - there exist at least one elementary condition of a rule that does not match the new object's description

   In case of multiple matching and partial matching proper solution strategy is necessary.
6. Decision rules vs. decision trees: Trees split the data space, where rules cover parts of the space.





7. Approaches to avoid overfitting:
   - pre-pruning - stop learning the rule before it perfectly covers only positive examples (satisfy minimum purity threshold)
   - post-pruning - learn discriminant set of decision rules (overfitted ones), then prune them on the separate pruning set until the evaluation measure does not fall below minimum threshold (generalization)