RMIT University
School of Computer Science and Information Technology
COSC2110/COSC2111 Data Mining

Tutorial Solutions Week 6

1. Filter Methods: In such methods, the subset selection procedure is independent of the learning algorithm and is generally a pre-processing step, for example only features that are highly correlated could be selected. This leads to a faster learning pipeline but it is possible for the criterion used in the pre-processing step to result in a subset that may not work very well downstream in the learning algorithm.

   Wrapper Methods: In these methods, the subset selection takes place based on the learning algorithm used to train the model itself. Subsets are generated by a search procedure and passed to the learning algorithm for evaluation. The most accurate subset is chosen. There may be many be many subsets to evaluate which could lead to long computation times.

2. Filter method: Run J48 and examine the tree. Select only the attributes that appear in the tree.

   Wrapper: Use J48 s the classifier.
3. IBK is a classifier and cannot be used in a filter method for feature selection because all attributes are needed for distance calculations, but is OK in wrapper.
4.

(a) Start with single attributes and keep only the best in each iteration.

Iteration 1: 100 candidate attribute sets: {A001}, {A002}, {A003}, … {A100}

Now apply the 1-nearest neighbor classifier. With {A001}, find the instances whose values of A001 are closest to the instances to be classified. Use the class label of the closest instance as the label for the test instance, and calculate the error rate. Keep only the candidate set with the lowest error rate. Here let us assume it is {A002}.

Iteration 2: Generate new candidate sets by adding a new attribute to the one we kept in the first iteration and repeat the classification process. Note that the distance function is assumed to be Euclidean distance in this example.

99 candidate sets: {A002, A001}, {A002, A003}, {A002, A004}, … {A002, A100} Again apply the classifier, choose the best candidate set, and go to the next iteration.

Stop: i) When the required accuracy has been achieved
ii) When the space has been exhausted

(b) Suppose there are 5 attributes, A1,A2,A3,A4,A5. In the first pass A3 is identified as the best. In the second pass only A3+A1, A3+A2, A3+A4, A3+A5 would be examined. If A3+A5 is the best, then in the third pass A3+A5+A1,A3+A5+A2,A3+A5+A4 would be examined.

It is clear that many combinations of an exhaustive search, for example, A1+A5, A1+A4+A5 will not be examined.

(c)Similar to (a) but start with the full set and eliminate the worst one each time.

(d) Multiple approaches exist. One feasible approach would be to do forward and backward selection in turns, but never remove the attributes selected by forward selection in backward selection, and never add the attributes removed by backward selection.

(e) If were trying to choose N features from a set of 100, then there are $^{100}C_N$ ways to do this. So if we want to try all possible sizes for the feature set (from N = 0 to N = 100), then the number of combinations will be

$^{100}C_0 + {}^{100}C_1 + {}^{100}C_2 + ... + {}^{100}C_{100} = 2^{100}$ (since were just summing a row of a Pascal triangle)

Another way to look at this: Each feature can be either included or excluded (2 possible values) so there are 2^{100} possible feature sets.

2^{100} is approximately $1.26 * 10^{30}$. There are $3.15569*10^7$ seconds in a year. Assuming one execution of the classifier in the wrapper takes 1 second, this would be about 10^{23} years.

f) The greedy algorithm at first glance looks like it might be n!, but its actually o(n**2). There are n choices for the first attribute. Then it is fixed and there are n-1 choices for the second attribute, ie n + (n-1) invocations of the wrapper [not n*(n-1)]. To go through all n attributes is then n +(n-1) + (n-2) ..... + 1 which is n*(n-1)/2.

The terminating condition for the greedy search would be to go through all the possibilities and return the one with the lowest error. For 100 attributes this would mean 100*99/2 = 4950 calls to the wrapper.

5. Apply OneR each time on the entire attribute set . OneR picks one attribute which is the best one. Then eliminate this attribute and add it to the list of selected attributes. Again run OneR on the rest of attributes to pick the second best attributes and add the second best to the list. And keep continue till the subset of attributes has been obtained.

6. It is not sensible to use OneR as the classifier in a wrapper. OneR picks the best single attribute. Any combination would degenerate to one attribute.