Introduction to Data Mining

Lab 3 – Simple Classifiers

## **Name: Phạm Đức Đạt**

**ID Student: ITITIU20184**

## Simplicity first!

In the third class, we are going to learn how to examine some data mining algorithms on datasets using Weka. (See the lecture of class 3 by Ian H. Witten, [1][[1]](#footnote-1))

In this section, we learn how **OneR** (one attribute does all the work) works. Open weather.nominal.arff, run OneR, look at the classifier model, how is it?

Getting a rule that is: branch on "outlook"; if it's "sunny" then choose "no", "overcast" choose "yes", and "rainy" choose "yes". It gets 10 out of 14 instances correct on the training set.

The success rate is 43%, worse then ZeroR (as using cross-validation for a small dataset).

- Remarks:

Use OneR to build decision tree for some datasets. Compared with ZeroR, how does OneR perform?

|  |  |  |
| --- | --- | --- |
| **Dataset** | **OneR -** **accuracy** | **ZeroR - accuracy** |
| weather.nominal | 0.43 | 0.64 |
| Supermarket | 0.67 | 0.64 |
| iris | 0.92 | 0.33 |
| glass | 0.58 | 0.36 |
| diabetes | 0.71 | 0.65 |

Excepting weather.nominal, OneR works better for the other datasets to build decision trees.

## Overfitting

What is “overfitting”? - **overfitting** occurs when a [statistical model](http://en.wikipedia.org/wiki/Statistical_model) describes [**random error**](http://en.wikipedia.org/wiki/Random_error) or **noise** instead of the underlying relationship, b/c of complex model, noise/error in the data, or unsuitable applied criterion, 🡪 poor prediction. To avoid this, use cross-validation, or pruning... [ref: <http://en.wikipedia.org/wiki/Overfitting>]

Follow the instructions in [1], run OneR on the weather.numeric and diabetes dataset…

Write down the results in the following table: (cross-validation used)

|  |  |  |
| --- | --- | --- |
| **Dataset** | **OneR** | **ZeroR** |
| weather.numeric | Classifier model: outlook  sunny -> no  overcast -> yes  rainy -> yes  Accuracy: 0.43 | Classifier model: class value “yes”  Accuracy: 0.64 |
| weather.numeric w/o outlook att. | Classifier model: humidity  < 82.5 -> yes  >= 82.5 -> no  Accuracy: 0.5 | Classifier model: class value “yes”  Accuracy: 0.64 |
| diabetes | Classifier model:  plas:  < 114.5 -> tested\_negative  < 115.5 -> tested\_positive  < 127.5 -> tested\_negative  < 128.5 -> tested\_positive  < 133.5 -> tested\_negative  < 135.5 -> tested\_positive  < 143.5 -> tested\_negative  < 152.5 -> tested\_positive  < 154.5 -> tested\_negative  >= 154.5 -> tested\_positive  Accuracy: 0.71 | Classifier model: class value: tested\_negative  Accuracy: 0.65 |
| Diabetes w/ minBucketSize 1 | Classifier model: pedi, branching on every single one  Accuracy: 0.57 |  |

MinBucketSize? – it affects to how the model branching.

Remark? -

## Using probabilities

Lecture of Naïve Bayes: [1]

* All attributes contribute equally and independently 🡪 no identical attributes

Follow the instructions in [1] to exame **NaiveBayes** on *weather.nominal*

|  |  |
| --- | --- |
| **Classifier model** | **Performance** |
| Naive Bayes Classifier  Class  Attribute yes no  (0.63) (0.38)  =============================  outlook  sunny 3.0 4.0  overcast 5.0 1.0  rainy 4.0 3.0  [total] 12.0 8.0  temperature  hot 3.0 3.0  mild 5.0 3.0  cool 4.0 2.0  [total] 12.0 8.0  humidity  high 4.0 5.0  normal 7.0 2.0  [total] 11.0 7.0  windy  TRUE 4.0 4.0  FALSE 7.0 3.0  [total] 11.0 7.0 | (how many percent of total instances are classified correctly?)  57.14% |

## Decision Trees

Lecture of decision trees: [1]

How to calculate entropy and information gain?

Entropy measures the impurity of a collection.

Information Gain measures the Expected Reduction in Entropy.

*Info. Gain = (**Entropy of distribution before the split) – (Entropy of distribution after the split)*

Values(*A*) is the set of all possible values for attribute *A* and *Sv* is the subset of *S* for which attribute *A* has value.

Build a decision tree for the weather data step by step:

|  |  |
| --- | --- |
| **Compute Entropy and Info. Gain** | **Selected attribute** |
| Entropy(S) = 0.94  Entropy(sunny) = 0.97  Entropy(overcast) = 0  Entropy(rainy) = 0.97  Gain(S,A) = 0.94 – (5/14\*0.97 + 5/14\*0.97) = 0.247 | outlook |
| Entropy(S) = 0.94  Entropy(false) = 0.81  Entropy(true) = 1  Gain(S,A) = 0.94 – (8/14\*0.81 + 6/14\*1) = 0.048 | windy |
| Entropy(S) = 0.94  Entropy(high) = 0.98  Entropy(normal) = 0.59  Gain(S,A) = 0.94 – (7/14\*0.98 + 7/14\*0.59) = 0.155 | humidity |
| Entropy(S) = 0.94  Entropy(hot) = 1  Entropy(mild) = 0.92  Entropy(cool) = 0.81  Gain(S,A) = 0.94 – (4/14\*1 + 6/14\*0.92 + 4/14\*0.81) = 0.029 | temperature |
| outlook = sunny  | humidity = high: no (3.0)  | humidity = normal: yes (2.0)  outlook = overcast: yes (4.0)  outlook = rainy  | windy = TRUE: no (2.0)  | windy = FALSE: yes (3.0) | *Final decision tree* |

Use Weka to examine J48 on the weather data.

## Pruning decision trees

Follow the lecture of pruning decision tree in [1] …

Why pruning? - Prevent overfitting to noise in the data.

In Weka, look at the J48 leaner. What are parameters: minNunObj, confidenceFactor?

* minNumObj is the minimum number of instances per leaf
* confidenceFactor is the confidence factor used for pruning

Follow the instructions in [1] to run J48 on the two dataset, then fill in the following table:

|  |  |  |
| --- | --- | --- |
| **Dataset** | **J48 (default, pruned)** | **J48 (unpruned)** |
| diabetes.arff | 73.8% accuracy, tree has 20 leaves, 39 nodes | 72.7% accuracy, tree has 22 leaves, 43 nodes |
| breast‐cancer.arff | 75.5% accuracy, tree has 4 leaves and 6 nodes | 69.6% accuracy, tree has 152 leaves and 179 nodes |

## Nearest neighbor

Follow the lecture in [1]

“Instance‐based” learning = “nearest‐neighbor” learning

What is k‐nearest‐neighbors (K-NN)? – is a method to classify unknown data point, by using its nearest neighbors, then choose the majority class among those.

Follow the instructions in [1] to run lazy>IBk on the *glass* dataset with k = 1, 5, 20, and then fill its accuracy in the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Dataset** | **IBk, k =1** | **IBk, k =5** | **IBk, k =20** |
| Glass | 70.6% | 67.8% | 65.4 |

1. http://www.cs.waikato.ac.nz/ml/weka/mooc/dataminingwithweka/ [↑](#footnote-ref-1)