Practical Data Mining

COMP-321B

Tutorial 4: Preprocessing

Shevaun Ryan
Mark Hall

August 1, 2006

1 Introduction

For this tutorial we will be using the **Preprocess** panel, the **Classify** panel and the **Select attributes** panel. At different stages of the tutorial we will be using and comparing three classification methods: the Naive Bayes model, the decision tree model J48 and the nearest neighbour algorithm IBk. All of these classification methods have been covered in previous tutorials.

In the 'Discretization' section of the tutorial, the supervised and unsupervised Discretize filters will be compared, as well as different settings in the unsupervised Discretize filter. WEKA's supervised Discretization filter used in the tasks is the 'weka.filters.supervised.Discretize' filter, and the unsupervised Discretization filter is 'weka.filters.unsupervised.Discretize' filter.

In the 'Feature Selection' section of the tutorial, two attribute evaluation methods will be tested: WrapperSubsetEval and CfsSubsetEval. In WEKA, the Wrapper attribute evaluation function is found under 'weka.attributeSelection.WrapperSubsetEval' and Cfs is found under 'weka.attributeSelection.CfsSubsetEval'.

In the 'Sampling' section of the tutorial, the 'Sampling With Replacement' filter is used in the task. It is found under 'weka.filters.unsupervised.instance.Resample'.

1.1 Introduction to the data sets used

All the datasets that are to be used are in WEKA's ARFF format, and have been selected especially for this tutorial. The measurement taken to compare classification tests is the accuracy of the classifier. All datasets come from the UCI data sets (See tutorial 2 for more information about UCI).

sick.arff The information in this dataset comes from thyroid disease records supplied by the Garavan Institute and J. Ross Quinlan, New South Wales Institute, Sydney, Australia.

vote.arff This dataset comes from the 1984 United States Congressional Voting Records Database

mushroom.arff Mushroom records drawn from The Audubon Society Field Guide to North American Mushrooms (1981).

letter.arff This file holds Letter Image Recognition Data, where the objective is to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters in the English alphabet.

2 Discretization

Discretization is the process of turning numeric attributes into nominal attributes by converting the numeric values into a small number of distinct ranges.

The main benefit of this is that some classifiers can only take nominal attributes as input, not numeric attributes. Another advantage is that some classifiers that can take numeric attributes can achieve improved accuracy if the data is discretized prior to learning. In the following task, we will find out whether the Naive Bayes is an example of such a classifier.

2.1 Exercise A: Apply the Discretize filter to the 'Sick' dataset

Task A1: Load the data set 'sick.arff' and browse the attribute information / details. How many of the attributes are numeric? Write down their attribute numbers.

Task A2: Change to the **Classify** panel and run the classification algorithm 'Naive Bayes' (weka.classifiers.bayes.NaiveBayes) using cross-validation to test its performance. The number of folds for cross-validation stay at their default value 10. Run the algorithm with its default settings and record the accuracy.

Task A3: Change back to the **Preprocess** panel. Choose the supervised Discretization filter (filters.supervised.attribute.Discretize) and apply (using default settings). Browse the details of the attributes you wrote down in Task A1. How are they different? How many distinct ranges have been created for each attribute?

Task A4: Undo the Discretize filter. Change the filter to the unsupervised Discretization filter (filters.unsupervised.attribute.Discretize) and set the 'bins' setting to 5 (filter settings are found by clicking on the box to the right of the 'Choose' button). Leave the other settings as default and click 'Apply'. Have a look at the attributes that you wrote down in Task A1. Undo the filter and redo it with the bins set to 10. What do you think the 'bins' setting affects?

Task A5: Undo the Discretize filter and go back to the **Classify** panel. Choose the FilteredClassifier (weka.classifiers.meta.FilteredClassifier) and set its classifier to Naive Bayes and its filter to (supervised) Discretize. Select Cross Validation from the test options, run the test and record the accuracy.

Task A6: Run the same test as in the previous task, but using the unsuperivsed Discretize filter (still within the FilteredClassifier). Run this test three times, changing the bins setting from 5 to 10 to 20. Record the accuracy of each test result.

Task A7: Compare all 5 accuracy readings. Which test had the highest accuracy rate? What can you say about discretization in general? Supervised discretization versus unsupervised discretization? What difference do the size of the bins make using unsupervised discretization?

3 Feature Selection

In this section we will be trying to improve the accuracy of three different classifiers (J48, IBk, NB) by identifying and removing irrelevant attributes from datasets before they are used to build the classification models. We will use two different models to evaluate the relevance of each attribute, the WrapperSubsetEval, which evaluates attribute sets by using a learning scheme, and the CfsSubsetEval, which evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them.

3.1 Exercise B: Use CfsSubsetEval to improve the J48, IBk and NB classification algorithms

Load the data set 'mushroom.arff'.

Task B1: Go to the **Classify** panel. Choose J48 as the classifier (weka.classifiers.trees.J48). Leave the classifier settings as default, and make sure that in the test options 'Cross Validation' is selected (10 folds). Press start and record the resulting accuracy in the 'J48' section of the answers. Now change the classifier to IBk (weka.classifiers.lazy.IBk) and press start (leave all other settings the same). Record the resulting accuracy in the 'Nearest Neighbour' section. Finally, change the classifier to Naive Bayes (weka.classifiers.bayes.NaiveBayes) and press start (leave all other settings the same). Record the resulting accuracy in the 'Naive Bayes' section.

Task B2: Go to the **Select attributes** panel. Set the Attribute Evaluator to CfsSubsetEval (weka.attributeSelection.CfsSubsetEval) and set the Search method to GreedyStepwise (weka.attributeSelection.GreedyStepwise). In the Attribute Selection Mode box, select 'Cross Validation 10 folds' and press 'Start'. Analyse the results window (a high number beside an attribute means that that attribute was selected in a lot of the folds, and is preferred by the algorithm). What do the results say about the attribute(s) with a score of 0%?

Record the attribute numbers of the **relevant** attribute(s) (those with a high score).

Task B3: Notice that the feature selection in the above task uses all of the data in order to evaluate the relevance of each attribute, whereas the classification in task 1 never uses all the data at one time, because of the cross validation. We can prevent the feature selection from having this unfair advantage by using a meta classifier to achieve feature selection and classification at the same time (i.e. both algorithms will be performed on each fold). To do this, go back to the **Classify** panel and select the AtrributeSelectedClassifier (weka.classifiers.meta.AttributeSelectedClassifier). Click on the name to change the settings; use the same classifier settings as in

the previous three tests (e.g. for the first test the settings will be: **classifier:** J48, **evaluator:** CfsSubsetEval, **search:** GreedyStepwise). Run all three tests and record the resulting accuracy in the appropriate section.

Task B4: Compare the before and after tests for each model, and comment on your findings (did the accuracy improve? Why/Why not?). Don't forget you can use information from the classifier output and the tree visualisation to help you analyse the results.

3.2 Exercise C: Use WrapperSubsetEval to improve the J48, IBk and NB algorithms

Load the data set 'vote.arff'.

Task C1: Go to the **Classify** panel. Choose J48 as the classifier (Weka.classifiers.trees.J48). Leave the classifier settings as default, and make sure that in the test options 'Cross Validation' is selected (10 folds). Press start and record the resulting accuracy in the 'J48' section of the answers. Now change the classifier to IBk (weka.classifiers.lazy.IBk) and press start (leave all other settings the same). Record the resulting accuracy in the 'Nearest Neighbour' section. Finally, change the classifier to Naive Bayes (weka.classifiers.bayes.NaiveBayes) and press start (leave all other settings the same). Record the resulting accuracy in the 'Naive Bayes' section.

Task C2: Go to the **Select attributes** panel. Set the Attribute Evaluator to WrapperSubsetEval (weka.attributeSelection.WrapperSubsetEval) and then click the word 'WrapperSubsetEval' to change its settings. Change the classifier to J48 (weka.classifiers.trees.J48) and leave the rest of the settings at default. In the main windows, change the Search Method to RankSearch (weka.attributeSelection.RankSearch) and then click the word 'RankSearch' to edit its settings. Change the Attribute Evaluator to InfoGainAttributeEval (weka.attributeSelection.InfoGainAttributeEval). Make sure the Cross Validation button is still selected in Attribute Selection Mode. Press Start and wait for the results. Record the attribute numbers of attributes that you think are **irrelevant**.

Task C3: Go back to the **Classify** panel and select the AtrributeSelectedClassifier (weka.classifiers.meta.AttributeSelectedClassifier). Click on the name to change the settings; use the same classification settings as in the previous three tests (e.g. for the first test the settings will be: **classifier:** J48, **evaluator:** WrapperSubsetEval, **search:** RankSearch, **attributeEvaluator:** InfoGainAttributeEval). Run all three tests and record the resulting accuracy in the appropriate section.

Task C4: Compare the before and after tests for each model and comment on your findings (did the accuracy improve? Why/Why not?). Don't forget you can use information from the classifier output and the tree visualisation to help you analyse the results.

4 Sampling

In this section, we are looking at Sampling as a means of reducing the size of very large datasets. Sometimes, the cost of using an entire dataset to build and test a classification model is too high, and so extracting a representative sample of the instances can reduce this cost. In the following exercise we are going to use the Resample filter, which produces a random subsample of a dataset using sampling with replacement.

4.1 Exercise D: Using Resample to take a sample of a large dataset

Load the data set 'letter.arff'.

Task D1: Stay in the **Preprocess** panel. Choose an attribute and record the **min**, **max**, **mean** and **standard deviation** of the attribute. How many instances are there in this dataset?

Task D2: Choose the Resample filter (filters.unsupervised.instance.Resample), and click on the name to edit its settings. Change the sampleSizePercent to 50.0. Apply the filter. How many instances have been removed from the dataset? Record the new **min**, **max**, **mean** and **standard deviation** of the attribute that you chose in Task D2.

Task D3: Compare the statistic values from Task D1 and Task D2. Have these values changed much? Why/Why not? Give a benefit of sampling a large dataset.

5 Answers

Answer A1: Number of numeric attributes .....
Attribute Nos. .....

Answer A2: Naive Bayes, before discretization .....

Answer A3:

Answer A4:

Answer A5: Naive Bayes, supervised discretization .....

Answer A6: Naive Bayes, unsupervised discretization: bins = 5 .....
bins = 10 ..... bins = 20 .....

Answer A7:

Answer B1: J48 ..... Nearest Neighbour .....
Naive Bayes .....

Answer B2: Attributes with a score of 0% are .....
Instance(s) .....

Answer B3: CfsSubsetEval: J48 ..... Nearest Neighbour .....
Naive Bayes .....

Answer B4:

Answer C1: J48 ..... Nearest Neighbour .....
Naive Bayes .....

Answer C2: Instance No. .....

Answer C3: WrapperSubset: J48 ..... Nearest Neighbour ..... Naive Bayes .....

Answer C4:

Answer D1: Attribute ..... min ..... max ..... mean ..... std. ..... No. of instances .....

Answer D2: No. of instances removed ..... min ..... max ..... mean ..... std. .....

Answer D3:
