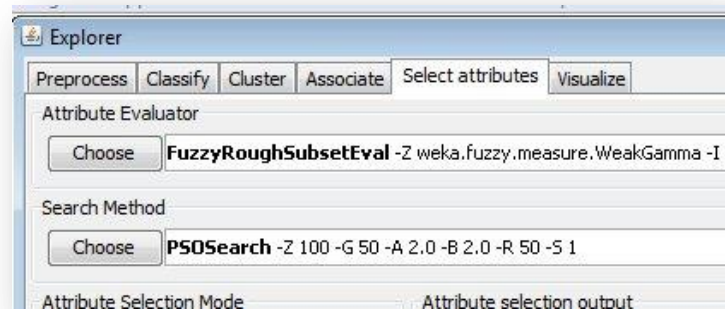


Fuzzy-rough data mining with Weka

Richard Jensen

This worksheet is intended to take you through the process of using the fuzzy-rough tools in Weka explorer and experimenter. It assumes no knowledge of Weka, so feel free to skip some of the initial steps if you are already familiar with it.



The fuzzy-rough version of Weka can be downloaded from:

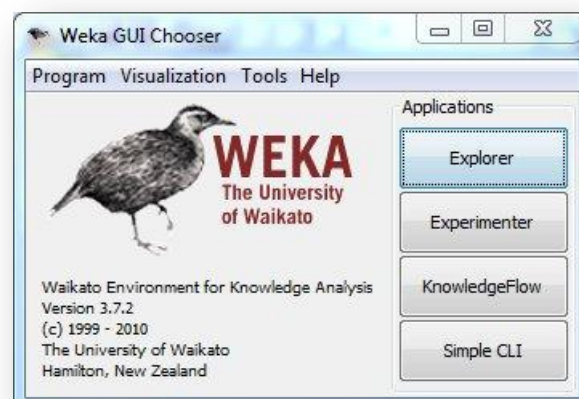
<http://users.aber.ac.uk/rkj/book/wekafull.jar>

And the datasets we'll use in this tutorial can be downloaded here:

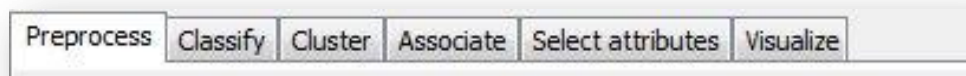
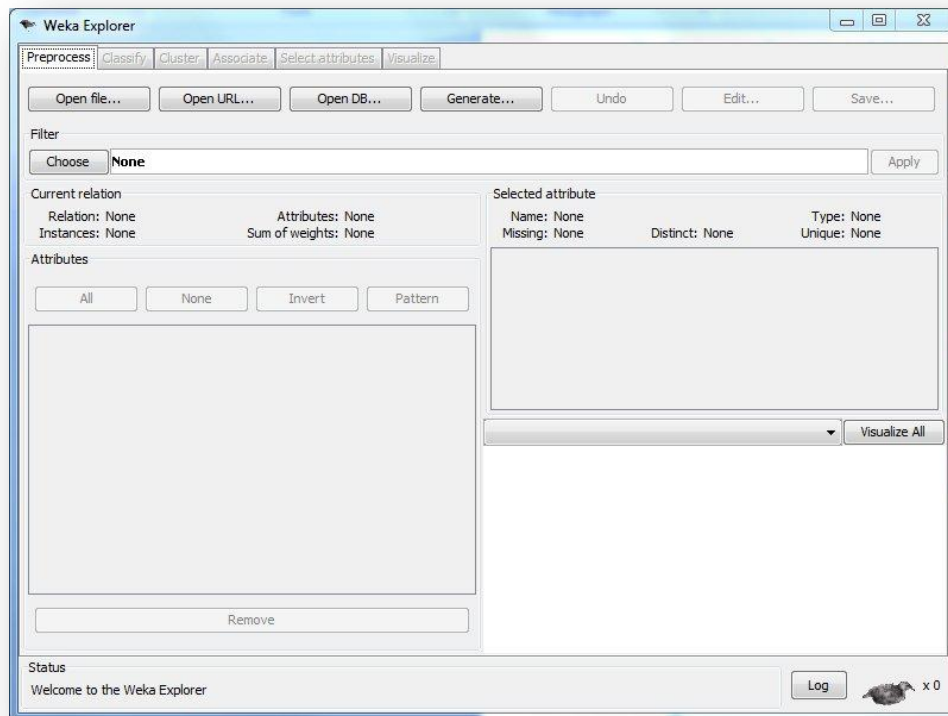
<http://users.aber.ac.uk/rkj/heart2.arff>

<http://users.aber.ac.uk/rkj/housing.arff>

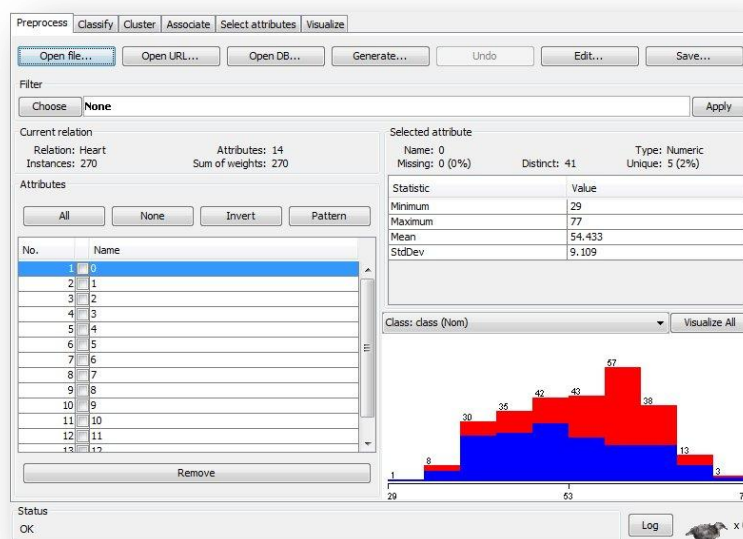
Step 1: Getting started



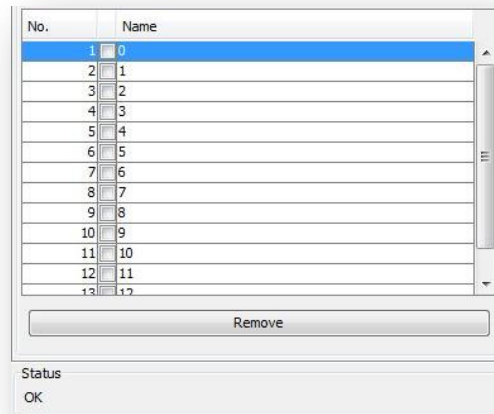
Double-click the 'wekafull.jar' icon. You should then see the Weka GUI chooser as above. Select 'Explorer'.



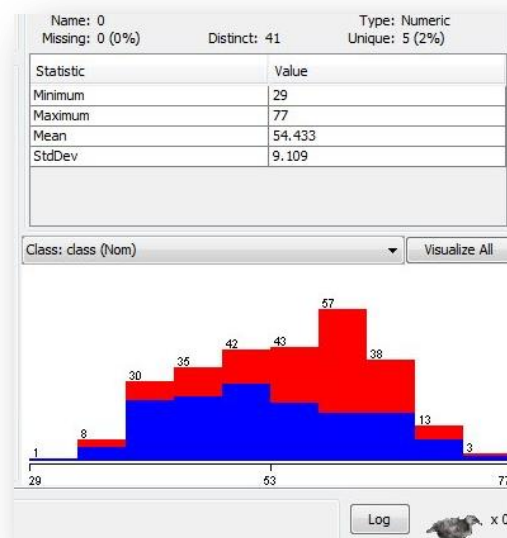
There are a number of tabs towards the top of the window which are activated when a dataset is loaded. We'll only be considering the 'Preprocess', 'Classify' and 'Select attributes' tabs in this tutorial. Click 'Open file...' and navigate to wherever the **heart2.arff** dataset is stored. Double-click on the file to load it into the Weka environment. You should see something similar to this:



Once the dataset has been loaded, you are presented with a range of information. This window is the 'Preprocess' tab, where the dataset can be processed, e.g. feature selection applied, instances removed, missing values removed etc. On the bottom-left of the screen there is a list of features in the dataset:



Options to manually remove features are present here as well, but we won't look at this. (It's fairly self-explanatory.) By clicking on one of the rows, statistics for that feature are displayed to the right:

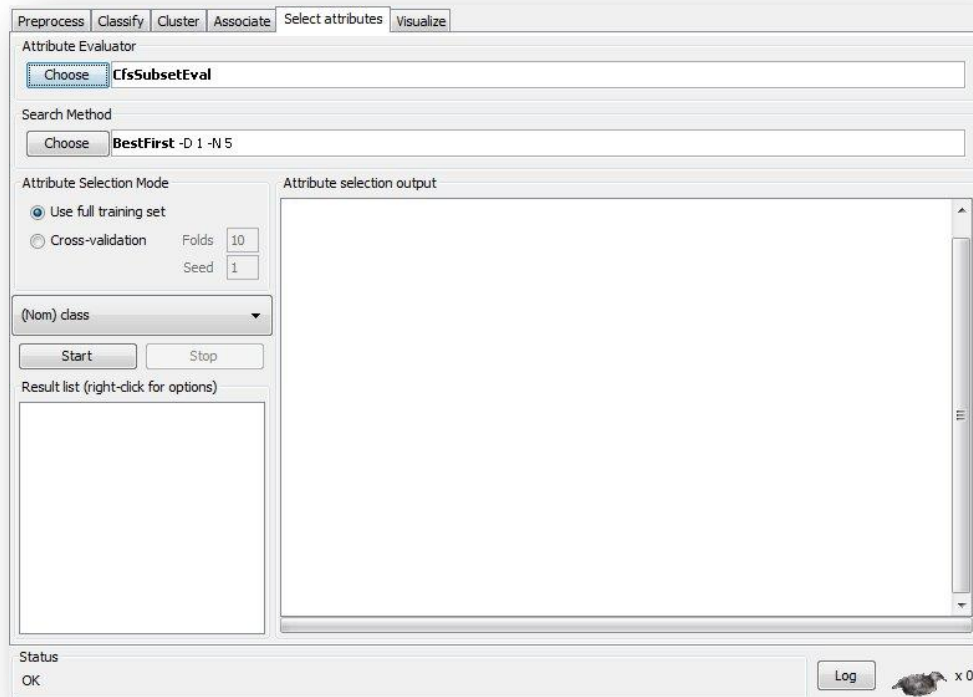


If the feature is discrete, then a bar chart of the frequencies of its values is displayed. If it's continuous (as in the figure above), then the distribution of values will be displayed. The 'Visualise All' button shows the visualisation for all features in the dataset.

Later on, we'll make use of the Filter part of this window.

Step 2: Fuzzy-rough feature selection

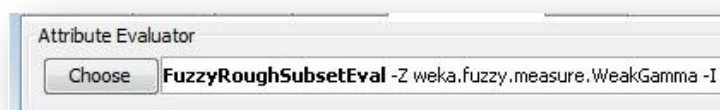
- Click on the 'Select attributes' tab. You should see something like the following:



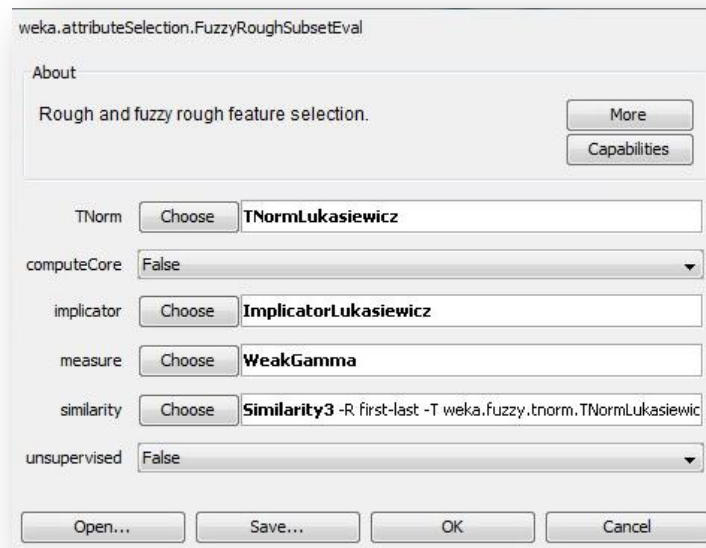
- Feature selectors are composed of two main components: an 'Attribute Evaluator' which evaluates the 'goodness' of feature subsets (sometimes just individual features only, depending on the evaluator chosen) and a 'Search Method' which searches through the space of feature subsets. We will look at both options.

Attribute Evaluator

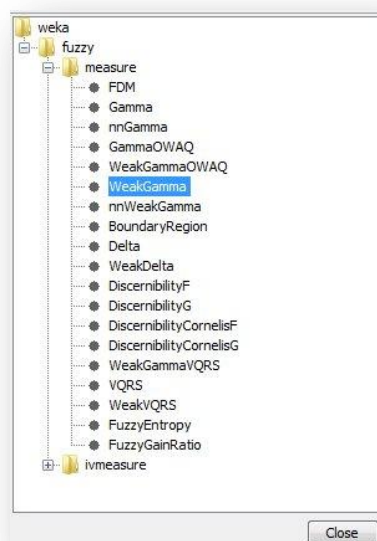
- First, we will choose the fuzzy-rough attribute selector. Click on 'Choose' under 'Attribute Evaluator' and you will be presented with a list of evaluators. Those ending in 'AttributeEval' will only allow features to be ranked; those ending in 'SubsetEval' will work for both feature ranking and feature subset selection. Click on 'FuzzyRoughSubsetEval' and you should see the following:



- Click anywhere along the line on the right next to 'Choose'. This opens up a dialog box containing a number of options that are used to configure FRFS (the default values are the typical ones used for feature selection).



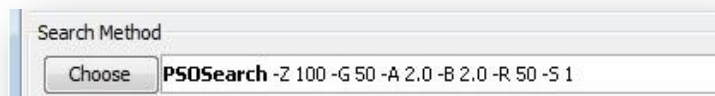
- The following options are available:
 - **TNorm**: the choice of t-norm (only used for measures that use the upper approximation).
 - **computeCore**: whether to compute the core of the dataset beforehand (i.e. calculate those features that should appear in every reduct and use this as a seed to the feature selection process).
 - **implicator**: the choice of implicator for the lower approximation calculation.
 - **measure**: the choice of measure to use (more on this below). The default is the (weak) fuzzy-rough dependency measure.
 - **similarity**: the choice of similarity relation (which also uses a t-norm to compose multiple relations).
 - **unsupervised**: a Boolean flag to set the method to be unsupervised (i.e. it ignores the decision feature and treats the rest of the data as an information system rather than a decision system).



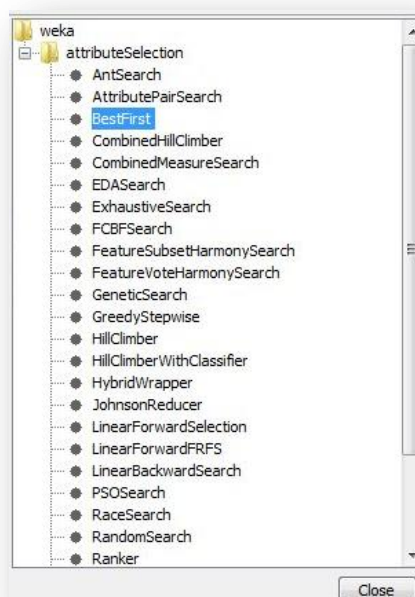
- If you select 'Choose' next to 'measure' you can see the fuzzy measures that are available for selection. Some of these are beyond the scope of this tutorial, but the main measures are:

- **Gamma/WeakGamma:** the fuzzy-rough dependency measures. WeakGamma is the 'weak' version of the positive region which is less computationally complex than Gamma and produces the same results for crisp decisions.
 - More information: <http://cadair.aber.ac.uk/dspace/handle/2160/2758> and <http://cadair.aber.ac.uk/dspace/handle/2160/3824>
- **BoundaryRegion:** uses the boundary region (the difference between the upper and lower approximations) to evaluate feature subsets. (Super)reducts have an empty boundary region.
 - More information: <http://cadair.aber.ac.uk/dspace/handle/2160/2758>
- **Delta/WeakDelta:** delta function. Similar to Gamma/WeakGamma but uses the min function instead of summation of positive region memberships.
 - More information: <http://cadair.aber.ac.uk/dspace/handle/2160/3824>
- **VQRS/WeakVQRS:** the vaguely-quantified approach to FRFS. Once selected, you can set its parameters (alpha and beta).
 - More information: <http://cadair.aber.ac.uk/dspace/handle/2160/581>
- **FDM:** the fuzzy discernibility matrix approach. Once selected, several options for this can be chosen. This generates fuzzy clauses (crisp clauses if the data is discrete) for use in either the JohnsonReducer or SATSearch search methods. If 'simplify' is set to true, then the clause database is reduced as the clauses are generated and this will significantly reduce memory consumption, but will be slower.
 - More information: <http://cadair.aber.ac.uk/dspace/handle/2160/4825>

Search Method



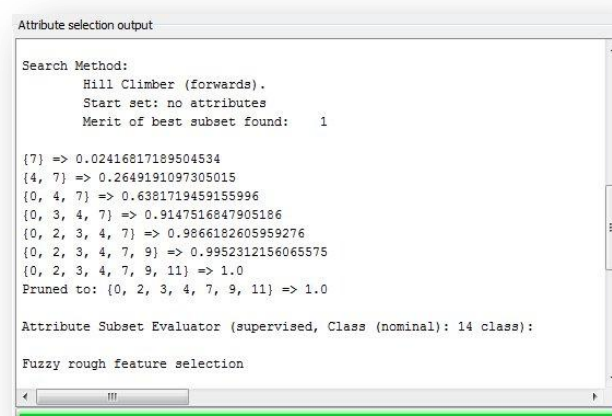
- By clicking on 'Choose' here, you can select from a number of search methods. Most of these are concerned with subset search rather than ranking individual features.



- The main search algorithms that you might want to use are:
 - **AntSearch**: Ant Colony Optimization-based search. Parameters to consider here include the number of ants, number of iterations, heuristic choice (used at the start of the search to weight each pair of features) and alpha and beta (for the probabilistic transition rule).
 - More information: <http://cadair.aber.ac.uk/dspace/handle/2160/488> and <http://cadair.aber.ac.uk/dspace/handle/2160/423>
 - **GeneticSearch**: uses a Genetic Algorithm.
 - **HillClimber**: performs greedy hill-climbing (equivalent to QuickReduct). This contains a couple of options that you might find useful. 'Prune' will quickly prune the final subset found (whilst maintaining the quality of the subset), as this search method often produces subsets that contain redundant features. 'searchBackwards' allows the algorithm to perform a backward rather than forward search (equivalent to ReverseReduct). The 'alpha' parameter is for finding fuzzy decision reducts.
 - More information: <http://cadair.aber.ac.uk/dspace/handle/2160/2758>
 - **JohnsonReducer**: for use with the FDM measure.
 - **PSOSearch**: uses Particle Swarm Optimization. This can be a very effective search method.
 - More information: <http://cadair.aber.ac.uk/dspace/handle/2160/382>
 - **SATSearch**: for use with the FDM measure. Performs a DPLL-style search for determining the 'optimal' (i.e. smallest) reduct (optimality is guaranteed).
 - More information: <http://cadair.aber.ac.uk/dspace/handle/2160/4825>

Task: Perform FRFS for the heart2 data.

Select the FuzzyRoughSubsetEval with the HillClimber search method and press 'Start'. After a few seconds, the search will have stopped and outputted those features appearing in the final subset. More information on the search process can be found by scrolling up in the Attribute selection output window:



```

Attribute selection output

Search Method:
  Hill Climber (forwards).
Start set: no attributes
Merit of best subset found: 1

{7} => 0.02416817189504534
{4, 7} => 0.2649191097305015
{0, 4, 7} => 0.6381719459155996
{0, 3, 4, 7} => 0.9147516847905186
{0, 2, 3, 4, 7} => 0.9866182605959276
{0, 2, 3, 4, 7, 9} => 0.9952312156065575
{0, 2, 3, 4, 7, 9, 11} => 1.0
Pruned to: {0, 2, 3, 4, 7, 9, 11} => 1.0

Attribute Subset Evaluator (supervised, Class (nominal): 14 class):

Fuzzy rough feature selection
  
```

In this example, you can see that the first feature selected is feature 7 (note that the index for the features starts at 0) and has a fuzzy-rough dependency of 0.024168.

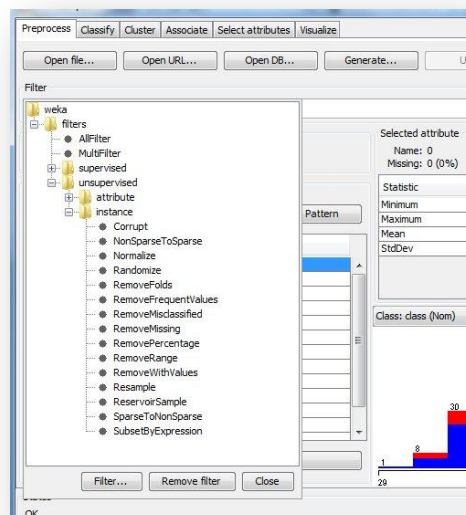
Things to try:

- Perform a backward search by setting the searchBackwards flag in HillClimber to true.
- Use different search methods, e.g. PSOSearch, GeneticSearch.
- Use different connectives and similarity relations for FRFS and see what impact they have.
- Use different measures, e.g. WeakVQRS, WeakGammaOWA.
- Compute the core for the dataset by setting computeCore to true and running (forward) HillClimbing. The core information is outputted in the 'Attribute selection output' pane

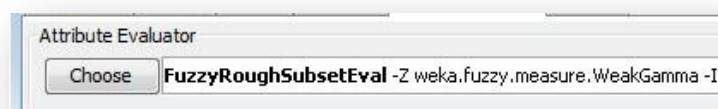
The 'Select attributes' tab also has the option of using cross-validation to look at the frequencies of appearance of features in subsets across the different folds.

Step 3: Handling missing values

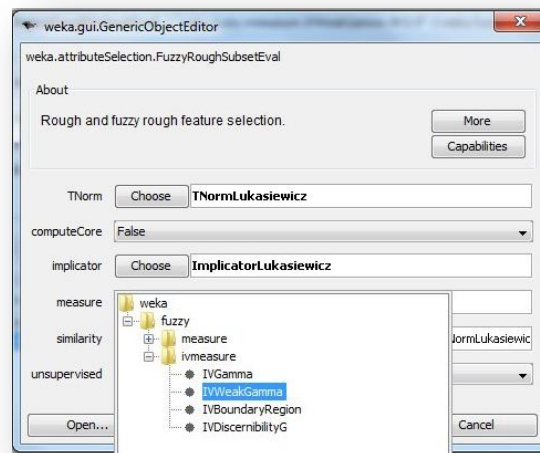
- Another problem encountered with real world data is that some of this data may be missing (i.e. there are some objects in the data that have missing values). Sometimes this is through measurement error or human error, or just a lack of information when a sample is recorded.
- There are three main ways of dealing with such data when using FRFS:
 - Remove instances with missing values using Weka's existing instance filters. There is more information on the use of filters in Weka in the next section on instance selection. The FRFS version of Weka contains a simple filter (RemoveMissing) that removes any instances with missing data; this filter is accessed via the Preprocess tab and is under unsupervised>instance.



- Replace missing data values with either the mean (for real-valued features) or the mode (for nominal features) using the ReplaceMissingValues filter, found under unsupervised>attribute.
- Use the interval-valued approach, IVFRFS. This is accessed via the 'Select attributes' tab, using the FuzzyRoughSubsetEval as the Attribute Evaluator. To select the interval-valued approach, open the options for FuzzyRoughSubsetEval by clicking anywhere on the line next to the 'Choose...' box:



From the 'measure' menu open the 'ivmeasure' folder, which will then display the fuzzy-rough measures that can be used for this type of data:



More information on the interval-valued approach can be found here:
<http://cadair.aber.ac.uk/dspace/handle/2160/2871>

Task: Dealing with missing values

There is a filter in *unsupervised>instance* called *Corrupt* which scans through the dataset (except the decision values) and makes a proportion of the data missing based on a supplied probability. Choose this filter and then set the mutation probability to a small value, e.g. 0.01. Apply the filter and see the impact of this by looking at the statistics for a few of the features (the % missing can be seen in the 'Selected attribute' pane. Save this modified dataset using the 'Save...' button, change the filename to avoid overwriting the original dataset. You can then use this dataset to

Apply the *RemoveMissing* and *ReplaceMissingValues* filters described above to see their results. If you are already familiar with Weka you might like to see how the results are affected for various classification algorithms for this dataset with and without missing values

Try the *IVFRFS* algorithm. Follow the procedure above to setup the *IVFRFS* algorithm, choose the *IVWeakGamma* measure select the *HillClimber* search method and click 'Start'. Compare the results with those obtained for the ordinary *WeakGamma* for the original dataset with no missing values.

Things to try:

- Repeat the above with different mutation probabilities to see their effect on *IVFRFS* (and other methods available in Weka). Probabilities over 0.1 will corrupt too much of the data, i.e. over 10% of all the data will be missing after application of the filter!

Step 4: Handling noisy data

- In this version of Weka, there are a couple of filters that add noise to data in order to test the robustness of algorithms in the presence of noise. The filters are *AddConditionalNoise* (to add noise to the conditional features) and is located in *unsupervised>attribute*; and *AddClassNoise* (to add noise to the decision feature only) and is located in

supervised>attribute. For both filters, there is a threshold which determines the number of values that are affected, $100 * \text{threshold\%}$ of values will have noise applied. For nominal valued features, a different value will be randomly chosen; for real-valued features, Gaussian noise will be applied to the original value.

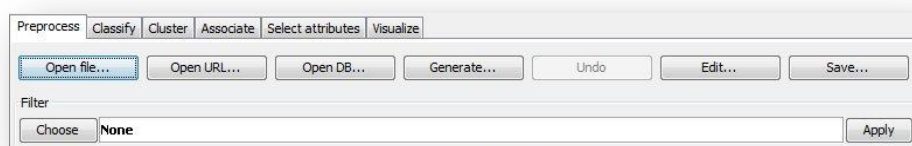
- The Vaguely Quantified Rough Set (VQRS) approach to feature selection intends to handle noise in a better way than traditional FRFS via the use of fuzzy quantifiers. This approach can be selected by choosing the FuzzyRoughSubsetEval attribute evaluator and setting its 'measure' to either VQRS or WeakVQRS (which are analogous to Gamma and WeakGamma described previously). The parameters for the quantifiers (alpha and beta) can be set, although the default settings should be appropriate.
 - o More information: <http://cadair.aber.ac.uk/dspace/handle/2160/581>
- Another approach for handling noise is via Ordered Weighted Average based fuzzy-rough sets. This can be selected via the GammaOWAQ or WeakGammaOWAQ measures in FuzzyRoughSubsetEval.
 - o More information: <http://cadair.aber.ac.uk/dspace/handle/2160/5808>

Task: Using VQRS to perform feature selection with noisy data

Load the original data. Apply FRFS with WeakVQRS and the HillClimber, note the subset found. Now use the AddConditionalNoise filter to add noise to the data values using various settings for the threshold, then apply FRFS with WeakVQRS and the HillClimber and see what happens. Repeat with WeakGammaOWAQ.

Note that to actually *reduce* the data in Weka, you need to apply a feature selection method as a filter in the 'Preprocess' tab. The filter to use is called AttributeSelection and can be found in supervised>attribute.

Step 5: Perform fuzzy-rough instance selection



- The fuzzy-rough instance selection methods are implemented as filters. They can be found in the filters list under supervised>instance. Three algorithms are available, two within InstanceSelectionBasic and one within InstanceSelection. For FRIS-I, use InstanceSelectionBasic with default settings; for FRIS-II, use InstanceSelectionBasic and setting the 'iterative' flag to true; for FRIS-III use InstanceSelection with default settings.
 - o More information on these methods:
<http://cadair.aber.ac.uk/dspace/handle/2160/4824>
- Although the traditional fuzzy-rough dependency measure is used in the paper at the URL above, different measures can be used, selectable under the 'measures' option for these algorithms. Note that the method requires the Similarity4 similarity relation, which contains the 'alpha' parameter – this must be tuned depending on the data used. Some discussion of this is given in the paper.

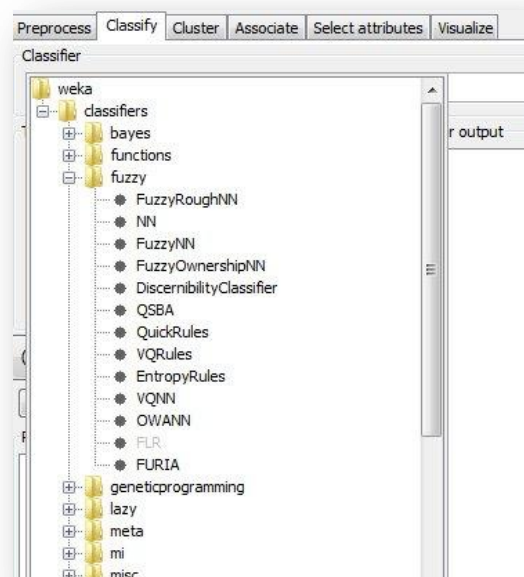
Task: Applying instance selection to data

Load the original data. Go to the 'Classify' tab and select a classifier of your choice, e.g. IBk in the fuzzy folder. Use 10-fold cross-validation (the default) and click 'Start'. Now select InstanceSelectionBasic in the 'Preprocess' tab, and set the similarity relation to Similarity4. The default value for alpha is 0.5, so we'll use this. Apply the filter and note the number of instances removed. Run the same classification experiment, but with this reduced data. Compare the performance (but bear in mind that the experiments concern different numbers of instances!). Repeat this for different values of alpha.

Step 6: Perform fuzzy-rough classification

Finally, we'll look at a couple of the fuzzy-rough classification (and prediction) methods we've developed for Weka.

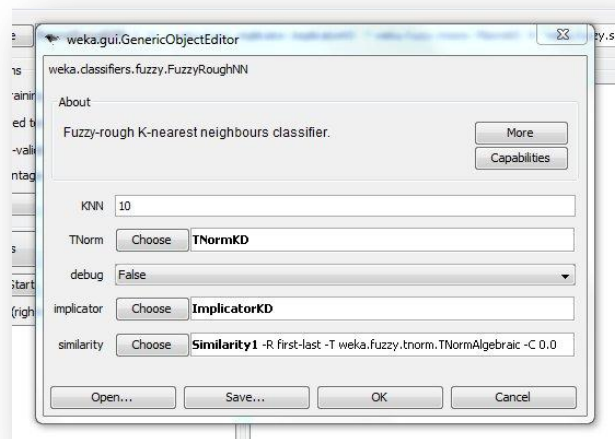
- The algorithms for fuzzy-rough classification can be found in the 'Classify' tab – click 'Choose' underneath 'Classifier' and then navigate to the 'fuzzy' folder:



- This tutorial will focus on just three of the algorithms contained in this folder: FuzzyRoughNN, VQNN and QuickRules.
- FuzzyRoughNN is the fuzzy-rough nearest neighbour approach, VQNN is the vaguely quantified approach

- o More information: <http://cadair.aber.ac.uk/dspace/handle/2160/5866>

The options for FuzzyRoughNN are below – you can set the t-norm, implicator and similarity relation used here. There is also the option for setting the number of nearest neighbours (KNN), but this will not affect FuzzyRoughNN for classification tasks (but will affect it for prediction).



- VQNN has a similar setup (and *is* affected by choice of KNN). It also contains options for setting the parameters for the quantifiers for ‘some’ and ‘most’, but these can be left at their default values.
- QuickRules is the hybrid feature selection/rule induction method. Again, the connectives and similarity can be set. There is also an option to set the level of pruning required (default is none) and an option to use the weak method (i.e. use the WeakGamma approach).
 - o More information: <http://cadair.aber.ac.uk/dspace/handle/2160/2872>

Task: Apply the classification methods

Load the original data. Select the FuzzyRoughNN method and use 10-fold cross-validation to evaluate the classifier. Note its performance and try VQNN and QuickRules using the same experimental setup. Now try selecting other similarity relations for FuzzyRoughNN and VQNN. Compare this with the performance of some of Weka’s other classifiers, e.g. J48, JRip etc.

If you have time left over you can repeat all previous steps with the housing data. (This has a real-valued decision feature.) Also, the Experimenter in Weka gives more flexibility when conducting experiments, as well as better options for the statistical analysis of the results, so if you have time you can try comparing some of these data mining algorithms using the Experimenter instead.