Java---- Machine- Learning

Published on *Java Machine Learning Library (Java-ML)* (http://java-ml.sourceforge.net)

By *Thomas Abeel*
Created *12/16/2008 - 12:01*

Feature selection

This article provides a brief introduction to the concepts of feature selection. In the other subsections of this chapter we address several topics related to feature selection.

There are three main types of features selection: (i) feature scoring, (ii) feature ranking and (iii) feature subset selection. Feature scoring is the most general method and can be converted in the latter two, while feature ranking can only be turned into feature subset selection methods.

Any of these three types of feature selection can be converted to an ensemble feature selection method. Currently Java-ML only provides an ensemble of feature rankers, but in the future the other two will be supported as well.

Feature scoring

All feature scoring algorithms implements the following method. Higher scores are better.

```
. public double score(int attIndex);
```

This method will return the score for the supplied feature index.

Typical usage of a feature scoring algorithm is shown in the snippet below

```
. /* Load the iris data set */
. Dataset data = FileHandler.loadDataset(new File [1]("iris.data"), 4, ",");
. /* Create a feature scoring algorithm */
. GainRatio ga = new GainRatio();
. /* Apply the algorithm to the data set */
. ga.build(data);
. /* Print out the score of each attribute */
. for (int i = 0; i < ga.noAttributes(); i++)
.     System [2].out.println(ga.score(i));
```

[Documented source code] [3]

Feature ranking

All feature ranking algorithms provide the following method to determine the rank of a features. Lower ranks are better.

```
. public int rank(int attIndex);
```

The method will return the rank of the feature with the provided index.
Typical usage of a feature ranking algorithm is very similar to the use of the feature scoring algorithms.

```
. /* Load the iris data set */
. Dataset data = FileHandler.loadDataset(new File [1]("iris.data"), 4, ",");
. /* Create a feature ranking algorithm */
. RecursiveFeatureEliminationSVM svmrfe = new RecursiveFeatureEliminationSVM(0.2);
. /* Apply the algorithm to the data set */
. svmrfe.build(data);
. /* Print out the rank of each attribute */
. for (int i = 0; i < svmrfe.noAttributes(); i++)
.     System [2].out.println(svmrfe.rank(i));
```

[Documented source code] [4]

Feature subset selection

Subset selection algorithms differ with the scoring and ranking methods in that they only provide a set of features that are selected without further information on the quality of each feature individually.
Subset selection algorithms provide the method

```
. public Set<Integer> selectedAttributes();
```

which will return a set of feature indices that have been selected by the algorithm.

The basic use of a feature subset selection algorithm is depicted in the snippet below.

```
.  /* Load the iris data set */
. Dataset data = FileHandler.loadDataset(new File [1]("iris.data"), 4, ",");
. /* Construct a greedy forward subset selector */
. GreedyForwardSelection ga = new GreedyForwardSelection(1, new PearsonCorrelation
. /* Apply the algorithm to the data set */
. ga.build(data);
. /* Print out the attribute that has been selected */
. System [2].out.println(ga.selectedAttributes());
```

[Documented source code] [5]

This examples create a greedy forward selection algorithm that will select one ('the best') feature. To determine the quality of the feature in this example the Pearson correlation is used.

Ensemble feature ranking

The ensemble feature ranking algorithm is another form of feature ranking and as such provides the following method to determine the rank of a feature. Lower ranks are better.

```
. public int rank(int attIndex);
```

The method will return the rank of the feature with the provided index.
Typical usage of an ensemble feature ranking algorithm is very similar to the use of a single feature ranking algorithm.

```
. /* Load the iris data set */
. Dataset data = FileHandler.loadDataset(new File [1]("devtools/data/iris.data"), 4
. /* Create a feature ranking algorithm */
. RecursiveFeatureEliminationSVM[] svmrfes = new RecursiveFeatureEliminationSVM[10
. for (int i = 0; i < svmrfes.length; i++)
.     svmrfes[i] = new RecursiveFeatureEliminationSVM(0.2);
. LinearRankingEnsemble ensemble = new LinearRankingEnsemble(svmrfes);
. /* Build the ensemble */
. ensemble.build(data);
. /* Get rank of i-th feature */
. int rank=ensemble.rank(i)
```

[Documented source code] [6]

Weka attribute selection

This article provides a brief introduction to the concepts of
using Weka attribute selection through the Java-ML feature selection interfaces.

In Weka, attribute selection searches through all possible combination of attributes in the data to
find which subset of attributes works best for prediction. It employs
two objects which include an attribute evaluator and and search method.

Any combination of these attribute evaluator and search algorithms can be used to determine the score and rank attribute in a data set. Currently, only
Java-ML feature scoring and feature ranking are available through the wrapper, subset selection is not yet implemented.

Typical use of the Weka feature selection wrapper is shown in the snippet below:

```
. /* Load the iris data set */
. Dataset data = FileHandler.loadDataset(new File [1] [1]("iris.data"), 4, ",");
. /*Create a Weka AS Evaluation algorithm */
. ASEvaluation eval = new GainRatioAttributeEval();
. /* Create a Weka's AS Search algorithm */
. ASSearch search = new Ranker();
. /* Wrap WEKAs' Algorithms in bridge */
. WekaAttributeSelection wekaattrsel = new WekaAttributeSelection(eval,search);
. /* Apply the algorithm to the data set */
. wekaattrsel.build(data);
. /* Print out the score and rank  of each attribute */
. for (int i = 0; i &lt; wekaattrsel.noAttributes(); i++)
.     System [2] [2].out.println("Attribute  " +  i +  "  Ranks  " + wekaattrsel.ra
```


Source URL (retrieved on *06/08/2009 - 06:26*): http://java-ml.sourceforge.net/content/feature-selection

Links:
[1] http://www.google.com/search?hl=en&q=allinurl:file java.sun.com&btnI=I'm Feeling Lucky
[2] http://www.google.com/search?hl=en&q=allinurl:system java.sun.com&btnI=I'm Feeling Lucky
[3] http://java-ml.sourceforge.net/src/tutorials/featureselection/TutorialFeatureScoring.java
[4] http://java-ml.sourceforge.net/src/tutorials/featureselection/TutorialFeatureRanking.java
[5] http://java-ml.sourceforge.net/src/tutorials/featureselection/TutorialFeatureSubsetSelection.java
[6] http://java-ml.sourceforge.net/src/tutorials/featureselection/TutorialEnsembleFeatureSelection.java