StatPatternRecognition: A C++ Package for Multivariate Classification

Ilya Narsky, Caltech

What is StatPatternRecognition?

- C++ standalone package
- Optional external dependency on Root for data I/O. By default uses Ascii. No other external dependencies.
- Distributed under GPL off Sourceforge: https://sourceforge.net/projects/statpatrec
- Implements plenty of methods for supervised learning and related.
- Command-line interface. No GUI.
- Full collection of talks and references at http://www.hep.caltech.edu/~narsky/spr.html
- README is the user manual

SPR: Implemented Classifiers

- Decision split, or stump
- Decision trees (2 flavors: regular tree and top-down tree)
- Bump hunter (PRIM, Friedman & Fisher)
- LDA (aka Fisher) and QDA
- Logistic regression
- Boosting: discrete AdaBoost, real AdaBoost, and epsilon-Boost. Can boost any sequence of classifiers.
- Arc-x4 (a variant of boosting from Breiman)
- Bagging. Can bag any sequence of classifiers.
- Random forest (rndm selection of inputs for decision splits)
- Backprop neural net with a logistic activation function (original implementation)
- Multi-class learner (Allwein, Schapire and Singer)
- Interfaces to SNNS neural nets (without training): Backprop neural net, and Radial Basis Functions

Rectangular cuts for physicists

- Decision tree and bump hunter implemented in SPR can optimize any figure-of-merit supplied through an abstract interface
- Ten FOM's can be used out of the box. User can easily implement one of his own.
    - conventional "statistical" FOM's – Gini index, cross-entropy etc
    - FOM's for physics analysis – signal significance, potential for discovery, expected upper limit etc.
- Analysts mostly use this functionality with the bump hunter (set of rectangular cuts) to optimize the desired FOM.

Other tools

- Cross-validation
- Bootstrap
- Tools for variable selection
- Computation of data moments (mean, variance, covariance, kurtosis etc)
- Arbitrary grouping of input classes in two categories (signal and background)
- Variable matching – every trained classifier remembers variables used for training and can map them onto input variables if they are supplied in a different order
- Multivariate GoF method proposed by Friedman at Phystat 2003
- Combiner of classifiers: training a classifier in the space of outputs of several sub-classifiers. Can train any classifier on an arbitrary set of sub-classifiers.

Overlap with TMVA

- Common classifiers
  - LDA (Fisher)
  - neural net (only one implementation in SPR)
  - boosted decision trees (Discrete AdaBoost only in TMVA)
- Implementations of decision trees are substantially different:
  - TMVA – binned; divides the range for each variable into intervals of fixed length
  - SPR – unbinned; potential splits are halfway between nearest points
- Implementations of bagging are different:
  - TMVA – randomly generate weight for each event on [0,1]
  - SPR – the standard bootstrap aggregation (sample N out of N with replacement)
- The overlap between SPR and TMVA is small

How SPR works

- Train a classifier on input data. SPR sucks all input data into memory and converts into internal SPR format.
  - Every input is coded as double.
  - Given the amount of memory on your machine, you can estimate the max size of input data SPR can handle.
- After training is completed, full classifier configuration (e.g., decision tree structure) is stored into a file.
- To apply the trained classifier to new data, you load the stored configuration from the file. Almost all classifiers in the package allow to resume training starting with this configuration. (AFAIK, this store/load/resume functionality is missing in TMVA).
- SPR executables take input test data, compute classifier responses and copy these input data along with classifier responses into an output file:
  - if use Ascii, can analyze the output file with SprOutputAnalyzerApp
  - if use Root, gotta write your own Root macros for analysis

User tools

- SPR comes with ~15 executables for various purposes. Basically, one executable per classifier plus general utilities
- SprInteractiveAnalysisApp
  - Interactive selection of classifiers and comparison of their performance on test data. Starts a dialogue with the user.
  - Saves classifier output into disk cache. If you want to change just one classifier, you do not need to re-enter parameters and re-train all others.
  - Should be used only on small datasets in not too many dimensions
- SprOutputWriterApp
  - For reading stored classifier configurations and applying them to test data. Can handle any classifier and can read several classifiers at once.
- SprOutputAnalyzerApp
  - For people who don’t use Root. Prints out efficiency curves for data and classifier responses stored in Ascii format.

Installation

- Distributed as source tarball off Sourceforge
- Several known quirks documented in INSTALL
- Successful builds on 32-bit SL3, SL4, RH4, 64-bit Fedora and 64-bit Debian. Enthusiasts adapted SPR to Solaris, WinXP and MacOS.
- SPR is an extra package for Fedora 6 (ascii):
  - ‘yum install spr’
- More quirks for Root-dependent version:
  - SprAsciiWriter is replaced with SprRootWriter by configure script
  - Need to edit executable source code to replace SprAsciiReader with SprRootReader.
  - May need to adjust compiler flags

Who is using SPR?

- I wish I knew…
- Cases I know of:
  - took off really well in BaBar PID (lepton, kaon, proton selectors) and rare leptonic groups
  - used by people for D0, SNO, and MINOS analyses, searches for supernovae
  - a few use cases outside HEP
- Feedback:
  - complaints (typically about installation)
  - thanks
  - almost none on what I am really looking for – what problems were studied, what results were obtained, what features would be useful, how SPR experience compares with other packages and/or methods

Summary and outlook

- Work is continued… subject to my time and availability
- SURF working on integration of SPR into Clarens web service http://clarens.sourceforge.net/
- Wish list (need manpower):
  - benchmarking against other packages
  - improving install procedure
  - primitive GUI
  - integration into Root?