How to Find More Supernovae with Less Work:

Object Classification Techniques for Difference Imaging

Stephen Bailey for the Nearby Supernova Factory

Nearby Supernova Factory

Lawrence Berkeley National Laboratory
G. Aldering
C. Aragon
S. Bailey
S. Bongard
M. J. Childress
S. Loken
P. Nugent
S. Perlmutter
R. Romano
K. Runge
R. Scalzo
R. C. Thomas
B. A. Weaver

Yale University
C. Baltay
A. Bauer
D. Herrera
D. Rabinowitz

Kavli Institute for Cosmological Physics
R. Kessler

Laboratoire de Physique Nucleaire et de Haute Energies de Paris
P. Antilogus
S. Gilles
R. Pain
R. Pereira

Institut de Physique Nucleaire de Lyon
N. Blanc
C. Buton
Y. Copin
E. Gangler
G. Smadja

Centre de Recherche Astronomique de Lyon
E. Pecontal
G. Rigaudier

What About Neural Nets?

In this study we haven't considered artificial neural nets (ANNs) yet. Why not? Experience from other applications indicates that ANNs likely aren't well suited for this problem for two reasons:

- In general ANNs don't perform well when there are significant outliers in the variable distributions. e.g. an ANN might treat a S/N=10000 candidate as being significantly better than a S/N=100 candidate, even if the shape is obviously bad.
- In comparison to boosted trees and random forests in other applications, artificial neural nets don't perform as well with a large number of input parameters (*N* > ~20).

Further Reading

This poster gives a flavor of what is possible with modern classification techniques, but doesn't cover the implementation details. For more information, see:

Hastie, Tibshirani, Friedman,
The Elements of Statistical Learning

A classic text on various machine learning and classification algorithms.

Byron P. Roe *et al.*, "Boosted Decision Trees as an Alternative to Artificial Neural Networks for Particle Identification", Nucl.Instrum.Meth. A543 (2005) 577-584 (physics/0408124)

Boosted decision trees applied to particle identification at the MiniBOONE experiment.

http://sourceforge.net/projects/statpatrec

The open source C++ package we used for boosted trees and random forests. This package implements several other classifier methods as well.

Abstract

We present the results of applying new object classification techniques to difference images in the context of the SNfactory supernova search. Most current supernova searches subtract reference images from new images, identify leftover objects, and apply simple threshold cuts on parameters such as statistical significance, shape, and motion to reject backgrounds such as cosmic rays, asteroids, and subtraction artifacts. This leaves a large number of non-supernova candidates which must be verified by human inspection before triggering additional followup.

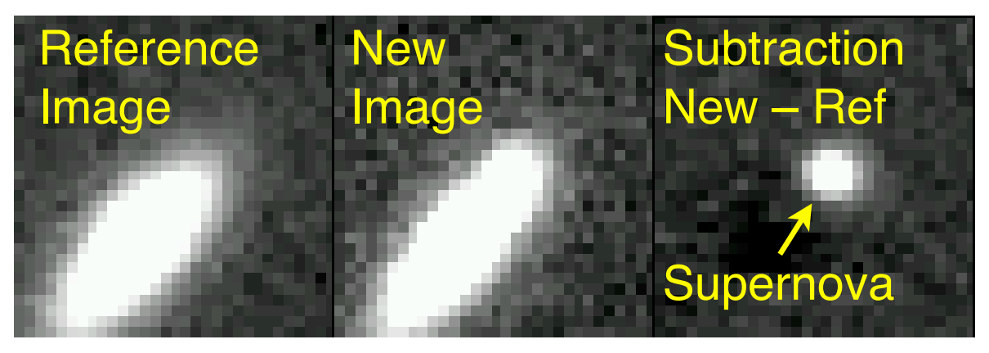
In comparison to simple threshold cuts, more sophisticated machine learning methods such as boosted decision trees, random forests, and support vector machines provide dramatically better true/false candidate discrimination. At the SNfactory, we reduced the number of background candidates by a factor of 10 while increasing our supernova identification efficiency. Methods such as these will be crucial for handling the large data volumes produced by upcoming projects such as PanSTARRS and LSST.

Can you find the supernovae?

How would you train a computer to recognize them?

1 Nearby Supernova Factory Search

The Nearby Supernova Factory searches for supernovae using data from the Near Earth Asteroid Tracking (NEAT) collaboration and the Palomar-QUEST survey. We process ~40,000 images covering ~600 square degrees per night. Reference image coadds are subtracted from new images and leftover objects are identified as potential supernova candidates.



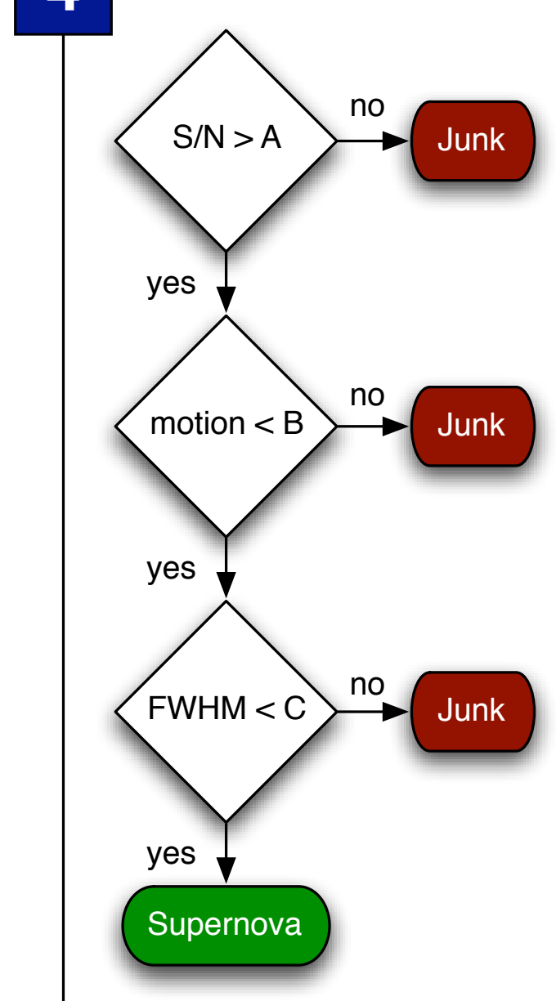
2 The Problem

Even after requiring objects to be detected at signal/noise > 3.5 on 2 or more images, we still had far more non-supernova objects than supernovae. Each of these (~1000 objects!) had to be individually scanned by a human to find the best candidates to schedule for spectroscopic followup. Threshold cuts on quantities such as roundness, full-width-half-max, and motion helped reduce the scanning load, but we were unable to achieve a reasonable scanning load while maintaining high supernova finding efficiency by using threshold cuts. This problem will be even worse for PanSTARRS and LSST which will cover much more area and intend to generate automated alerts.

3 Training and Validation

Classification methods require a training dataset to tune the parameters and a separate validation dataset to check the performance of the final results. We created these datasets by generating fake supernovae on real galaxies in our images. A nearby star of the desired magnitude on the same image was used to model the supernova, and this was added to the galaxy. By using real stars on the same image, we realistically model the point-spread-function, noise, image artifacts, etc. that a supernova on that image would have. These images with fake supernovae were processed through the same software pipeline as our real data to test the discovery efficiency.

4 Threshold Cuts

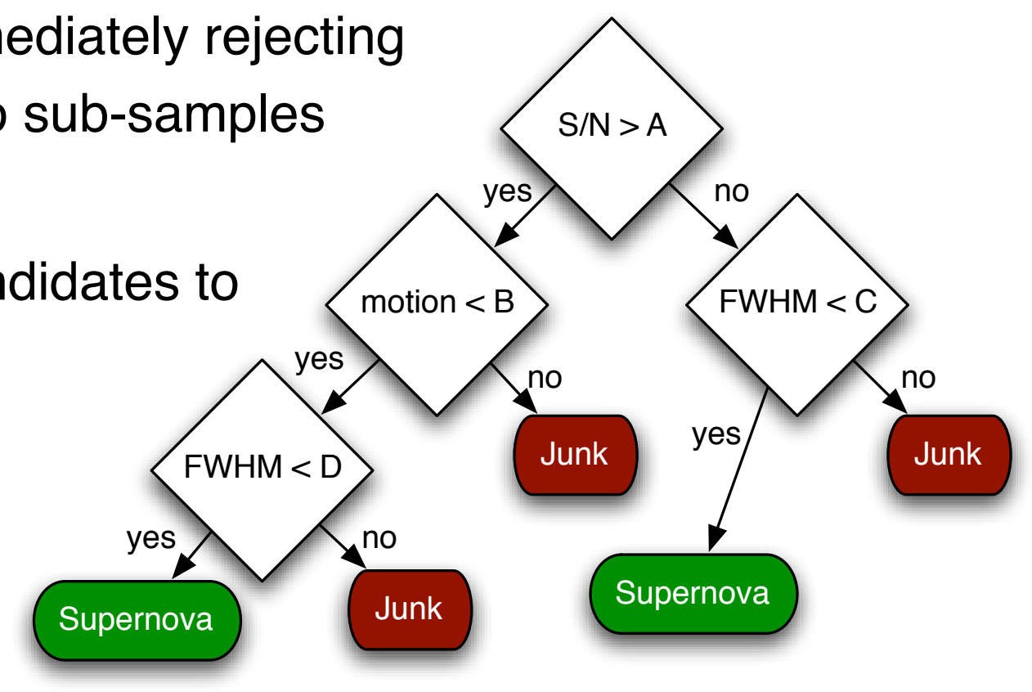


Threshold cuts remove objects which fail any of several cuts, e.g. a minimum required signal-to-noise threshold. Although they are easy to understand, they don't naturally account for correlations between variables and they treat objects which barely fail a single cut the same as ones which badly fail many cuts. Since objects must pass all of the cuts, every cut must have a high efficiency for the objects of interest. This makes it difficult to obtain good rejection power while maintaining a high efficiency for the combination of cuts on all parameters.

5 Decision Trees

Decision trees generalize upon the concept of threshold cuts by creating a tree of cut decisions. Rather than immediately rejecting events which fail a single cut, events are split into sub-samples and then further splits are applied.

This allows, for example, high signal-to-noise candidates to be treated separately from low signal-to-noise candidates. Correlations between variables can naturally be handled as well.



Boosted Trees

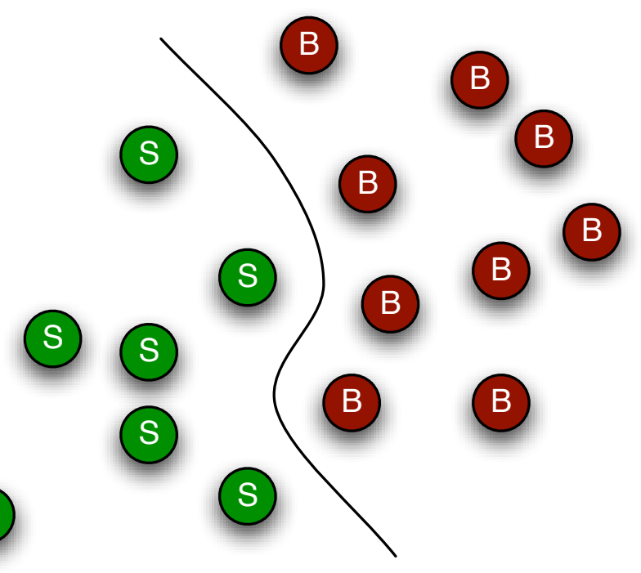
Boosted trees generate multiple decision trees from the same training dataset. At each iteration, events which were misclassified by the previous tree are given greater weight. This allows the training to focus on events which are hardest to classify to improve the quality of the decision trees. The final output is a weighted average of all decisions of all of the individual trees.

Random Forests

Random forests also generate multiple decision trees and take a weighted average for the final result. At each branch point, only a random subset of the possible split variables are considered. This provides a fast and robust training algorithm with results which are nearly as good as boosted decision trees.

6 Support Vector Machines

Support Vector Machines (SVM) attempt to separate two classes of events (e.g. supernovae and non-supernovae) in an *N* dimensional space by finding the hyper-surface which maximally separates the two classes. It uses the events at the border between the two classes to determine the hyper-surface; thus it focuses on the events which are hardest to classify, similar to a boosted decision tree.

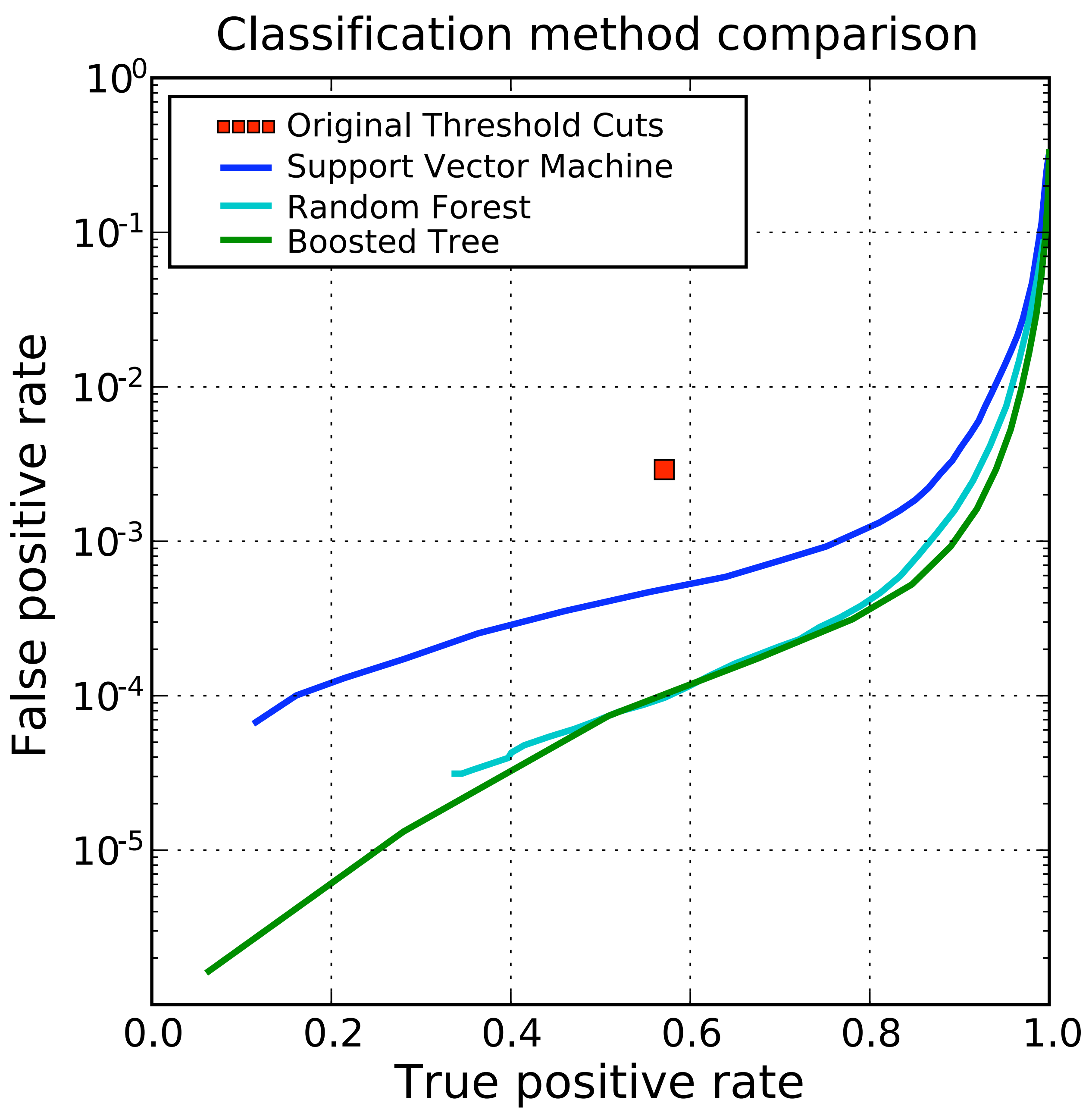


Although SVM produced considerably better results than threshold cuts for our data, it didn't perform as well as boosted trees or random forests. This is perhaps due to the fluctuations in our data which preclude a clean border between the two classes (e.g. a very young dim supernova on a bright host is difficult to distinguish from a statistical fluctuation).

10x fewer false positives
with *increased* efficiency
for real SN identification

Bottom Line:

Boosted Trees, Random Forests, and Support Vector Machines work dramatically better than threshold cuts for supernova identification in difference images. These methods will be crucial for maintaining a reasonable false positive rate in the automated rapid turnaround transient alerts from PanSTARRS and LSST.


Classification method comparison
🟥 Original Threshold Cuts
Support Vector Machine
Random Forest
Boosted Tree
False positive rate
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
True positive rate