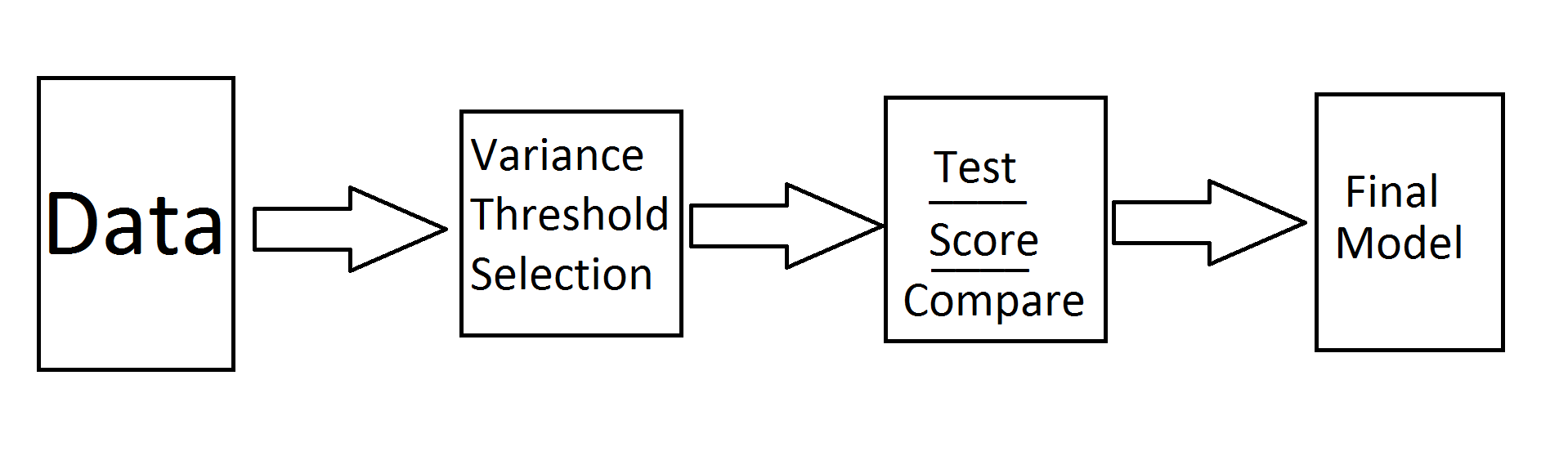
Lucas Hagel

Geoffrey Greenleaf

**Approach**

Our approach can be summarized as follows: we began by using library implementations of various machine learning algorithms (sklearn) such as logistic regression, support vector machines, neural networks, decision trees, etc. to get a rough estimate of each system's accuracy. The motivation was to test the waters and get a ballpark idea of which classifier to use. Afterwards we each picked the preferred classifier and tuned its parameters and the features of the input data to use the best one in the final evaluation. After this the next logical step was to collapse the data in a way that made the classifiers work better, one such technique was to reduce the number of features based on its variance. Lucas used support vector machines with an RBF kernel function after it outperformed a gamut of other classifiers in various combinations (Naive Bayes, Ada Boosted Decision Trees, SVM with different classifier functions, etc). Geoffrey used ExtraTreesClassifier and a brute force parameter search which looked for Variance and Tree Depth and scored the classifier with cross validation.

**Classifier Architecture:**



**Lessons Learned**

The biggest thing I learned was when I use the variance selection function also cuts off class labels. For a few weeks Lucas thought his classifier was working at 94%. After a debugging session to make the underlying features match the features being used with the preliminary data it was found that the class labels were cut off.

Using graphics cards seemed very promising because of the computational power would allow for more test samples to be done and after some research it appears that a good implementation results in at least a an order of magnitude speed increase. This would allow for more brute force feature or parameter searches. In particular I tried to use Theano; attempting to install it on windows was a colossal waste of time, then when I got it working in Linux it was tough to figure out how to use the libraries built on top of it. After some wrangling a classifier outputted 60% accuracy on 20% of the training data.

**Internal Evaluation**

Our internal evaluation consisted of looking at the rank, not seeing it at #1 which meant we had work to do.

**Improvements since the Preliminary Evaluation**

Initially after the preliminary evaluation the idea was to combine the strengths of 3-4 different classifiers because they had comparable cross validated accuracy scores. This did not seem to help so the approach we ended up using was adopted.