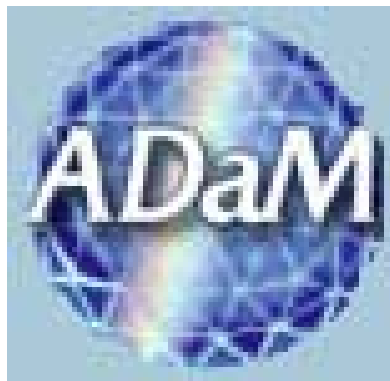


## **CMPUT 695: Project Proposal**

### ***ADaM: Algorithm Development and Mining system***



**For: Osmar Zaïane**

**Date: November 5<sup>th</sup>, 2004**

**By:**

**Dean Cheng  
John Sheldon**

The data mining tool reviewed below is the Algorithm Development and Mining system, further referred to as ADaM. ADaM was developed by the Information Technology and Systems Center at the University of Alabama. It was originally used to analyze images from satellite data. The system contains both a set of image processing toolkits along with a set of data mining toolkits. This review will evaluate how ADaM is utilized and will present the negative and positive aspects of the tool. This review will also compare ADaM to two other data mining tools, XL Miner and WEKA.

When downloading the application from the website, binaries can be downloaded for either Windows or Linux platforms. Two versions, 4.0.0 and 4.0.1 are provided on the website and version 4.0.1 was chosen to be reviewed given that it is the latest version. Along with the binaries, documentation can be downloaded giving a general overview of the tool so to better understand its functionality.

ADaM is expanded to give a folder of executable files and Python modules. The goal of the executables or Python modules is to provide a modular architecture that allows users to create their own applications, through the use of Perl or Python (see appendix A for an example), to solve specific data mining problems.

Another feature that ADaM supports is the ability to be accessed through a number of external interfaces. This can be useful for situations when large groups are working on a project from different physical locations.

ADaM contains (although not an exhaustive list) the following major tools for data mining:

- Classification Techniques
  - Naïve Bayes Classifier
  - Backpropagation
  - K-Nearest Neighbour
- Clustering Techniques
  - K-Means
  - Feature Selection/Reduction Techniques
- Pattern Recognition Utilities
  - K-fold Cross Validation
- Association Rule Mining.
- Optimization Techniques
  - Genetic algorithms

There are also many useful image processing features:

- Basic Image Operations
  - Image Arithmetic
  - Image Sampling
- Edge and Shape Detection
- Filtering
- Texture Features

These image processing modules make analyzing images fast and effective. Image analysis would be useful in situations where spatial data analysis would be important such as in the situation mentioned above with satellite image data.

An API document is given that describes each module, the options that each module has and the intended purpose. The API document is not extensive and is further discussed later in the document.

ADaM requires that a data set be converted to ARFF (Attribute-Relation File Format) data format. See appendix B for an example of an ARFF file.

We will now compare ADaM to two other known data mining tools, WEKA and XL Miner.

XL Miner is an add-on application to Microsoft Excel. It contains a number of similar packages to ADaM in a nice drop down list in the Excel user interface. With this package, a novice user can test out the features with relative ease and little background knowledge of the function of the tools. The trial version of the application add-on that is available is not powerful and is limited by a set of constraints. Due to this, in depth testing was not done to see what capabilities it had. In short, the data mining tool is limited by the Excel interface in terms of input ranges and output ranges. In terms of integrating XL Miner into larger projects, unlike ADaM, it is a stand alone tool that would not be easily integrated and is inflexible to user needs. XL Miner is intended more for the business person who wishes to get more from his data yet does not wish to invest time and money into developing or learning a more powerful tool.

Comparing ADaM to Weka, another data mining tool, the lack of user interfaces seems to be a disadvantage for ADaM. Weka provides a graphical user interface where users can load and run the different tools. Weka also provides visualization and reports for some of its tools. In addition, Weka includes a tool to visualize and display a workflow of complex multiple steps analysis including several components of the tools. All of the above are all desirable features that ADaM could perhaps provide in a separate package. One advantage ADaM has over Weka is the

ease of integration. The modular nature of ADaM provides it with more potential to build a more effective and efficient data mining analysis program.

Assembling the positive aspects, ADaM incorporates the ability to perform stand alone operations or to be used in distributed and parallel processing systems. Allowing for parallel running of this data mining tool provides a powerful tool that can be used in CPU intensive operations. ADaM is also versatile and it is easy to integrate part of ADaM into existing tools. Python or Perl can be used to integrate functions in different parts of ADaM into one script and then existing products can run the script to get the desired results. One example provided by the ADaM website is to use a Python script to represent the workflow of a Naive Bayes Classifier. See appendix A for a code snippet. Thus, instead of running each Bayes module step by step, one Python script can be run thus making it programmer and research friendly.

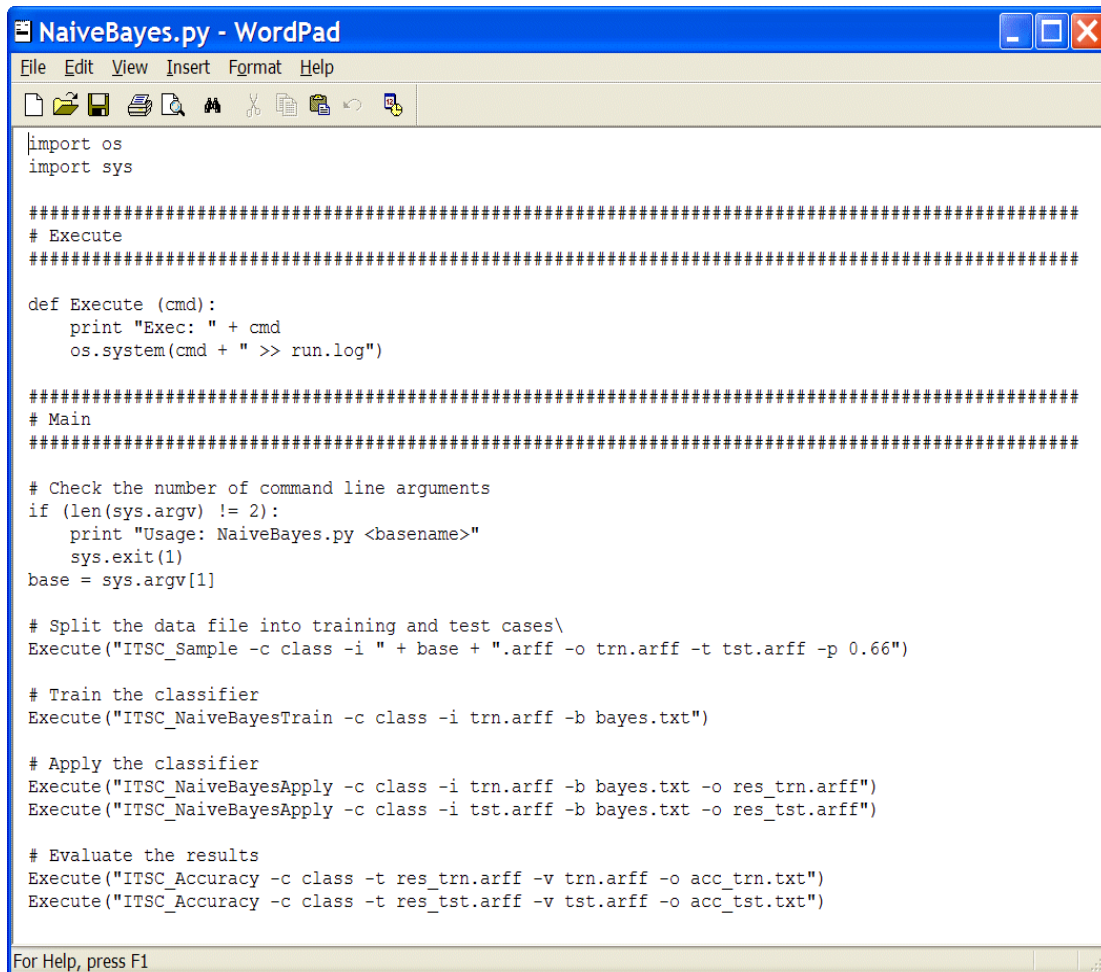
Negative aspects of ADaM pertain to user interface capabilities. An interface is not provided to the user and as such, there are no clickable buttons, no visualization tools and no graphical user interface. ADaM users are left to develop an interface that incorporates these modular aspects. Due to the fact that users are left to develop their own wrapper for the modules, ADaM is not user friendly for novices or small business owners' wishes to tap into the powerful world of data mining. The help provided in the downloaded version or in the `-h` option do not give thorough indications on specifically what inputs are required and what kind of output can be expected when the module is run.

In addition, ADaM does not contain descriptive reports to explain and to display the outputs so the user will need to interpret the output themselves or develop an interface to provide this visual representation. Another negative aspect of ADaM is that users will have to convert data into ARFF format and convert the output to users' own reports.

To conclude, ADaM has the capability to be included into a large and complex project involving many analyzes. One example is the use of ADaM to detect tropical cyclones or NASA's use of

ADaM on their Information Power Grid. In saying this, there is a steep learning curve with ADaM which must be addressed before results can be expected. Our recommendation would be to adopt another mining tool with a more interactive interface, such as WEKA, that would allow more immediate useful results. Upon understanding the limitations of the other product, if any, ADaM could be used to develop a more personalized and powerful tool to be used in our company for data mining and knowledge discovery.

## Appendix A



```
NaiveBayes.py - WordPad
File Edit View Insert Format Help

import os
import sys

#####
# Execute
#####

def Execute (cmd):
    print "Exec: " + cmd
    os.system(cmd + " >> run.log")

#####
# Main
#####

# Check the number of command line arguments
if (len(sys.argv) != 2):
    print "Usage: NaiveBayes.py <basename>"
    sys.exit(1)
base = sys.argv[1]

# Split the data file into training and test cases\
Execute("ITSC_Sample -c class -i " + base + ".arff -o trn.arff -t tst.arff -p 0.66")

# Train the classifier
Execute("ITSC_NaiveBayesTrain -c class -i trn.arff -b bayes.txt")

# Apply the classifier
Execute("ITSC_NaiveBayesApply -c class -i trn.arff -b bayes.txt -o res_trn.arff")
Execute("ITSC_NaiveBayesApply -c class -i tst.arff -b bayes.txt -o res_tst.arff")

# Evaluate the results
Execute("ITSC_Accuracy -c class -t res_trn.arff -v trn.arff -o acc_trn.txt")
Execute("ITSC_Accuracy -c class -t res_tst.arff -v tst.arff -o acc_tst.txt")

For Help, press F1
```

## Appendix B

An ARFF file is composed of two sections, a header section and a data section.

The header section contains the name of the relation and a list of the attributes, and their types. An example is given below.

An example taken from the website: [www.cs.waikato.ac.nz/~ml/weka/arff.html](http://www.cs.waikato.ac.nz/~ml/weka/arff.html) of a header.

```
% 1. Title: Iris Plants Database
%
% 2. Sources:
%   (a) Creator: R.A. Fisher
%   (b) Donor: Michael Marshall (MARSHALL@PLU@io.arc.nasa.gov)
%   (c) Date: July, 1988
%
@RELATION iris

@ATTRIBUTE sepallength NUMERIC
@ATTRIBUTE sepalwidth NUMERIC
@ATTRIBUTE petallength NUMERIC
@ATTRIBUTE petalwidth NUMERIC
@ATTRIBUTE class {Iris-setosa,Iris-versicolor,Iris-virginica}
```

The data section looks like the following: (example from same website)

```
@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
```