DORUS

Distributed Open Resources for Understanding Statistics

Technical Report for the Preliminary Examination in Statistical Computing

Greg Ridgeway
Department of Statistics
University of Washington
September 25, 1996

DORUS is advised by Werner Stuetzle and David Madigan
Funded by the University of Washington, College of Arts and Sciences

1 Introduction

Statistical education suffers. Traditional methods of statistical education have failed to train students to use statistical methods to solve real world problems (Cobb 1993, Hogg 1991, 1992, Mosteller 1988, Snee 1993, Yilmaz 1996). Students rarely have an opportunity to *do* statistics but rather are buffered from the data by examples that have already been worked out for them.

The modern computer has revolutionized the field of statistics and many attempts have been made to bring this revolution into the introductory statistics classroom. Packages such as Minitab (Ryan, Ryan, Joiner, Pennsylvania State University, 1973) and various computer based tutorials have sprouted up to give introductory statistics students an opportunity to involve themselves in real data. These packages and systems have failed to enhance modern statistical education.

At the University of Washington the Department of Statistics set out to redesign STAT 390 - Probability and Statistics for Engineers and Scientists. The new curriculum departed from the traditional introductory statistics curriculum and demanded more computer intensive data analysis tools. The new curriculum included analyzing data using permutation tests, classification and regression trees (CART), and interactive graphics. These data analysis tools offer a more intuitive approach to data analysis than traditional methods but cannot be performed simply with paper and pencil. Permutation tests and CART are products of the computing era and are computer intensive. Although software packages that implement these tools exist, their inadequacies led to a decision to design a new statistical package.

Why create yet another package for students? The current packages available were judged inadequate for instruction because they either lacked accessibility, an instructional component, or scalability. DORUS (Distributed Open Resources for Understanding Statistics) was created to provide the statistical tools for the new STAT 390 curriculum. This package, though still under development, shows promise for a whole new way of incorporating data analysis and computing into the classroom.

2 Goals

In order to avoid creating another statistical package plagued with the same inadequacies as those already in existence the following principles were established to which DORUS would be conformed.

- accessible
  The barriers to using this software should be minimal. Barriers to avoid include:
  - finances - the students should not be required to purchase expensive computing equipment (hardware nor software).
  - lack of computing expertise - the students should not be expected to be able to install complicated software or integrate "add-on" packages to existing software. This also implies that the user-interface should not be a barrier to the data analysis tools.
  - time - the program should require only a minimal amount of learning to operate and manuals should not be necessary.
- instructional
  - Provide an environment for students to interactively experiment with data and data analysis tools.
  - Provide statistical instruction while arriving at a solution.
  - Augment classroom or textbook lessons
- scalable
  DORUS should be available to the large user community in such away so that potentially a large number of people could operate the program simultaneously.

3 Design Rationale

3.1 Early model design proposals

Motivated by the goals of being accessible, instructional, and scalable, candidate models were proposed for implementation of this software package.

A potential candidate for such a model included coding new packages for S+, Mathematica, MathCAD, or Lisp-Stat that augmented them to handle the desired functionality needed by the student. Under this model a student would, for example, download or receive a disk with these new packages and execute them under, say, Lisp-Stat. This approach is

certainly scalable in that any number of students could simultaneously use this system. All of these models also provide a sufficient environment for creating an instructional tool as well. However, this model is far from accessible. This model assumes that a student can either access the programs on campus or alternately purchase one of these programs to install on their home computer, and execute the topic specific package. This requires that the students have the finances and ability to set up such a system on their own, distracting them from the real purpose of learning statistics. The instructional and scalable facets of this kind of model that would somehow distribute the computational resources to the students has redeeming qualities only if the accessibility issue could be overcome.

The World Wide Web is fast becoming an accessible method for distributing information. More and more people are gaining access to the Web from school, work, and home. Using the Web is easy and requires little computing expertise. In fact, most home computers now come with "Internet Ready" kits installed eliminating the need for the user to spend time figuring out how to access the Web. The Common Gateway Interface (CGI) that is widely implemented to allow Web users to submit forms or queries to a Web server potentially provides a means for submitting data to a C program that could analyze and return the results of a student's data analysis. This model, however, is also flawed. Despite its accessibility this model is not scalable and lacks interactive instructional capabilities. CGI scripts are executed on the Web server. If one hundred students simultaneously attempted to run their data analysis then all computation would be executed on the server and the server's computing power would be drained. Furthermore, this model lacks dynamic user interaction and, therefore, provides only a mildly instructional environment. It is impossible under the static CGI model to allow the student to interact with the data during analysis.

3.2 The Java model

Despite the limitations of CGI, usage of the World Web Wide still provides an attractive media for developing an accessible, instructional, and scalable statistical package. With Sun Microsystems' recent (April 1995) release of Java, developing software to be disseminated via the Web has become a practical option.

Java is an interpreted, object-oriented, platform-independent, multi-threaded, secure pro-

gramming language. The feature that sets it apart from other languages is that programs written in Java can be embedded into Web documents. When a user accesses a Web document that contains a Java program the program code is sent to the user's machine for client-side execution. Java's extensive abstract windowing toolkit and built-in abstract data types are ideal for the construction of a student oriented statistical package. The most widely used web browsers, Netscape 2.0+ and Microsoft's Internet Explorer 3.0, provide a Java interpreter for all competitive computing platforms (PC Windows, Macintosh, Unix, and others). The computing industry (including Microsoft, Oracle, Borland, Netscape, and IBM) has embraced Java as the programming language of choice for developing Internet applications. For these reasons a package developed in Java would achieve the original goals.

A statistical package developed in Java would be...

- **accessible** - Most students have access to the Web. Many machines on campus at the University of Washington as well as other institutions offer connections to the Web and many students have Web access at home as well. Using the Web requires little computing knowledge and little, if any, added financial burden to the student.

- **instructional** - The multi-threaded Java environment allows for the creation of appealing animated graphics for visual statistical instruction. This model also surpasses the static CGI model by allowing students to interact with data through a Web distributed program.

- **scalable** - Scalability of the Java model is one of its chief virtues. Once the student accesses the program all execution is passed to the user's machine. Such an instructional system could potentially reach a world-wide audience with no additional strain on the server.

3.3 Potential of the Java model

The potential for this model is enormous. Two visions that can be seen for this model are the concept of a decentralized statistical analysis package and a distance learning program accessible worldwide.

As the cost per megabyte of hard drive space becomes cheaper the software market sees the size of software packages become huge. A complete version of SAS for a desktop PC requires over 100 megabytes! On the other hand, one can envision a Web based statistical package where no single copy of the package exists, but rather this package is dispersed throughout cyberspace. Under this model a specialist could develop a data analysis tool, notify the statistics community of its location, and data analysts simply would need to link their dataset to this new tool for analysis.

The advantage of a package designed in this manner is that users need only to download the features for which they have an immediate need. Java programs are segmented into classes which are downloaded by the client's Java interpreter on an as needed basis. An enormous data analysis system could be developed, made available through Web connections, and shared by any number of users. Several large computing companies (Sun, Lucent) have already begun developing such a system for business applications and Wyse has actually implemented it.

As well as a dispersed statistical analysis system this model provides an excellent method for dispersing education around the globe. The non-traditional student, like single mothers and night time students, could take statistics courses completely via the Web, analyzing data and interacting with on-line worked out examples.

Although the intention of creating DORUS is specifically to supplement the new STAT 390 curriculum, this technology has potential for initiating a new paradigm for statistical analysis and distance learning.

3.4 Limitations of this Model

Despite meeting the initial goals of accessible, instructional, and scalable, this model does have some drawbacks, many of which soon may be overcome.

- **No local file access** - Most Web browsers do not permit local file access in order to prevent the creation of a Java virus that could be spread via the Web. While protecting the user, blocking access to the user's file system also prevents DORUS from saving datasets, storing graphs, and printing. Research is active in this area and a solution

may be expected by 1997.

- **Mathematical computation is slow** - Java is an interpreted language. The user's Web browser receives the programs code and then interprets it for local execution. Unvalidated reports indicate that an algorithm executed with interpreted Java causes a 14 to 28 times reduction in speed when compared with compiled C code (Digital Espresso, November 27, 1995). Under these conditions the likelihood that a Java statistical package could be competitive is small. Symantec and Borland have been developing JIT (Just-In-Time) technology where the Java code to be interpreted is compiled on the user's computer into machine code on the fly. Preliminary reports indicate that the performance of Java under a JIT compiler is equivalent to natively compiled C code.

- **Java is not completely debugged** - Because Java is still quite new, bugs in the compiler and platform specific interpreters still exist. Programs often appear quite different on various platforms. Some popular platforms still do not implement all of Java's features. With time these problems should be solved.

4 User Model Interface Design

The user interface should not be a barrier between the student and the statistical tools. The Java Abstract Window Toolkit (AWT) provides many GUI objects that allow for the construction of a transparent interface to the statistical tools. The DORUS system is operates around two types of objects: variables and analysis tools.

4.1 Variable objects

Variable creation and appearance (Figure 1 is modeled after DataDesk. Since most aspects of variable creation is not as critical to learning statistics less effort has been made to enhance them. The user interface to the variables is rather simple.

Variable objects contain data and a data type. Variables are uniquely identified by their variable name. For this reason the user is not permitted to create variables with the same

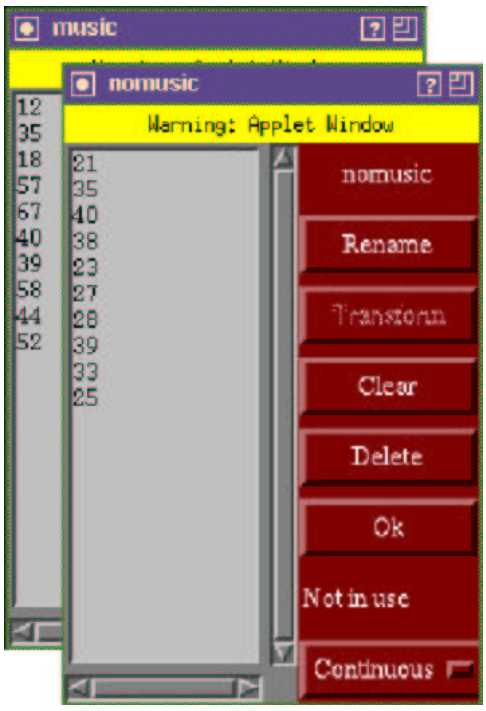

Figure 1: *Variable object*

name. The user enters and edits the data in a free form text region (`java.awt.TextArea`). The data remains in this text format until the user submits the data by pressing *Ok* at which point the text region is parsed (data elements are separated by commas or any kind of white space) into an array. By default, the data type is assumed to be continuous but may be changed to nominal or ordinal by the user. Continuous variables are parsed into array of doubles and nominal and ordinal variables are parsed into arrays of Strings. Continuous variables are also scanned for non-numeric data errors. Ordinal variables have a user-specified order to them (e.g. small, medium, large). Nominal data has no order associated with it (e.g. male, female).

Ideally students will use predefined datasets that can be loaded from the server (Appendix C). Avoiding having students manually enter data, especially since at this point it cannot be saved, allows the students to immediately get involved in doing data analysis.

Analysis tools, which will be further explained in Section 4.2, only have access to the parsed data. Therefore, no analysis can be performed on a particular variable until the user

selects *Ok*. Furthermore, if the variable's data has been edited none of the changes will be registered until the user again selects *Ok*. Analysis tools are capable of preventing variable editing if correct analysis insists upon strict data integrity. If a tool prevents editing then the user may still view the data but cannot alter the text region and is notified that the variable is in use.

No concept of variable relations have been developed nor have they been deemed necessary. For this reason, when the user selects a procedure that requires a relation, such as a paired test, DORUS assumes that all first observations in the variables to be analyzed relate to one data record. Future work may include developing variable relations or improving methods for displaying the data values to aid the user in visualizing the inferred relational structure. Variables are loosely organized by a dataset object which controls the interaction of analysis tools with the variables. The dataset object handles all creation, deletion, and retrieval of variables. The variable objects themselves implement methods for parsing, editing, typing, and naming.

Variable oriented methods are:

- †Variable naming
- †Associating data with the variable named
- Transforming the data elements (e.g. log transforms)
- †Variable deleting and clearing
- †Parsing

Features marked by † are currently implemented. Data transforms have not yet been implemented. Future work will implement creating new variables as univariate or multivariate functions of the existing variables using David Liu's (SGI) Expression evaluation package.

4.2 Analysis tools

Subsequent to the creation of variable objects, statistical analysis tools are selected. In all analysis tools DORUS attempts to minimize the need to use the keyboard and attempts to maintain an interactive visual environment for the student. The execution of all statistical

processes assumes that merely obtaining an answer is not sufficient for achieving the instructional goal. DORUS is designed so that statistical procedures will be instructional while the end result is being calculated.

DORUS' analysis tools follow this flow:

1. DORUS displays a window for the user to select variables and set the analysis tool's parameters. The selection window examines the variables currently stored in DORUS' dataset object and presents to the user only those variables which are sensible for the particular analysis by taking into account the variable's data type. This way a categorical variable will never be offered as a possible response variable for a regression analysis.

2. After the user selects *Submit*, a tool-specific data object is created that grabs the data from each variable submitted to the analysis tool and stores it in a format for efficient analysis. During the data objects creation DORUS checks the format of the data to verify that it is suitable for the selected analysis tool (e.g. variables selected for paired analysis must have an equal number of observations). To protect the integrity of the original variables, only the data objects may be altered by the analysis tools.

3. The analysis tool is executed and provides an environment for the user to interact with the data.

4. The tool is terminated and the system resources are garbage collected.

The following section expands more on the development of these analysis tools.

5 Functionality

Set upon the Java model, the new course curriculum for the University of Washington's Statistics 390 - Probability and Statistics for Engineers and Scientists motivated DORUS' initial set of features. The three critical areas of statistics deemed critical by the new curriculum are inference, data modeling, and data visualization,. To meet these needs, each area was individually approached keeping in mind that students are better at learning

intuitive procedures. Therefore, the new curriculum and DORUS take a permutation testing approach to inference, teach issues of modeling through classification and regression trees, and supply descriptive statistics and graphical methods for data visualization.

DORUS' analysis tool set

1. **Inference - Permutation tests**
   - †Unpaired two-sample
   - †Paired two-sample
   - †One way ANOVA
   - N-way ANOVA
   - †Slope parameter from simple linear regression
   - Parameters from multiple linear regression
   - Univariate confidence intervals
2. **Modeling - †CART**
3. **Data visualization**
   - Univariate statistics
     mean, variance, boxplot, histogram, mean confidence interval
   - Graphics
     - †Two-dimensional scatterplots
     - Three-dimensional scatterplots
     - Multiple-dimensional scatterplot matrix
     - Brushing and spinning

Features marked by a † have been implemented.

A data analysis tool that merely yields an answer does not sufficiently meet the instructional goal. Pedagogy is of utmost value even at the expense of speed. The instruction goal, therefore, is fundamental in the implementation permutation tests, CART, and graphics.

5.1 Permutation tests

Hypothesis testing is a central concept to teach to introductory statistics students. The traditional approach to presenting hypothesis testing, however, is to teach students the classic testing methods such as t-tests and F-tests. Students at this level usually learn about the paired and unpaired t-tests, one way analysis of variance, and significance tests for the slope parameter for simple linear regression. Unfortunately, most students just memorize the testing formulas and few ever really grasp the concept of a hypothesis test at all.

Permutation testing, on the other hand, presents a more intuitive approach to hypothesis testing. Performing permutation tests, however, requires the use of a computer. In the past, lack of computer access prevented permutation testing from being a viable topic for instructors to present in their introductory statistics courses. Since DORUS overcomes the problem of accessibility, presenting hypothesis testing under the permutation framework is now feasible.

Goals for DORUS' implementation of permutation testing

- Let students visualize the construction of the reference distribution
- Give students a feel for how the data is permuted
- Define p-values graphically for the students
- Show students a variety of ways in which permutation testing can be applied

As with all analysis tools DORUS implements the primary goal is not to analyze data but rather to educate. DORUS' implementation of permutation testing intends to aid the student in grasping the concept of hypothesis testing. DORUS achieves this by allowing the student to manually iterate through permutations of the data. At each step they can see graphically the construction of the reference distribution. When the number of permutations gets large the reference distributions become more familiar as they approach a Gaussian or F distribution. The methodic construction of the reference distribution aids the student in visualizing the meaning and interpretation of p-values.

Music group efficiency	No-music group efficiency
12	21
35	35
18	40
57	38
67	23
40	27
39	28
58	39
44	33
52	25

Table 1: *Music and efficiency example without group labels*

DORUS currently implements permutation tests for significant differences between means of two unpaired or paired samples, one way analysis, and significance of the slope parameter for simple linear regression. Future work will add N-way analysis of variance, multiple regression, and goodness of fit tests. DORUS will also eventually offer a more enhanced linear regression environment to demonstrate the actual permutation of data and the resulting line fits. Providing students with multiple scenarios in which permutation testing can be utilized to test hypotheses aids the student in understanding their diverse applications.

5.1.1 Permutation testing example

To demonstrate the usage of DORUS, a typical problem that a student taking an introductory statistics course might need to solve is solved using DORUS.

A researcher attempted to determine if music positively affected the efficiency of assembly-line employees. Twenty employees, assembling the same unit, were randomly assigned for one month to a working environment with music or to an environment without music. The employees' efficiency, calculated as average number of assembled units per hour, is reported in Table 1.

This particular problem calls for a two sample unpaired permutation test testing the hypothesis that the mean work efficiency for the group with music is significantly greater from the work efficiency for the group without music. The variables created for this analysis

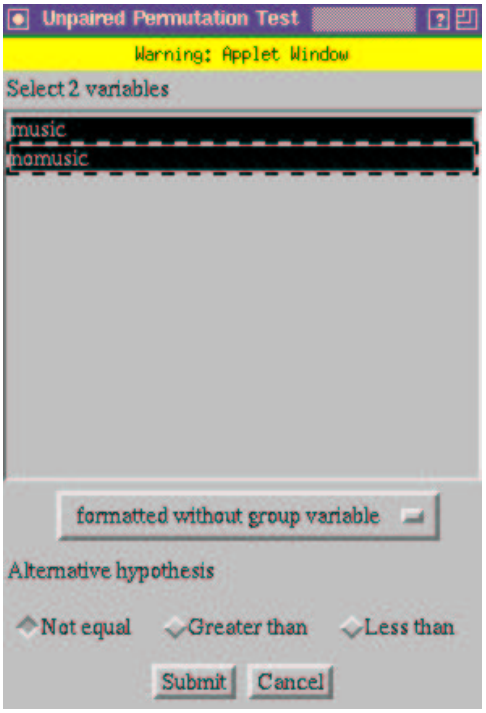

Figure 2: *Variable selection for unpaired test*

are shown in Figure 1. After requesting the unpaired permutation test the user is presented with a window for variable selection (Figure 2). Upon making an appropriate selection and pressing *Submit* a new window appears to display the construction of the reference distribution. The window (Figure 3) initially contains the histogram with only the sample test statistic plotted. The user can then step manually through the construction of the reference distribution. At any time the user can request to see the permuted data. In this example the sample test statistic was 11.3. After 3009 iterations, 105 of the permuted data sets had test statistics greater than 11.3 as shown in Figure 4 so the p-value is 0.0349. To double check these results, the same analysis was also performed using S+ which yielded a p-value of 0.0384.

5.2 CART

Permutation testing provides students with an intuitive way of learning the concepts of hypothesis testing. In the same way, the classification and regression tree (CART) [1] provides

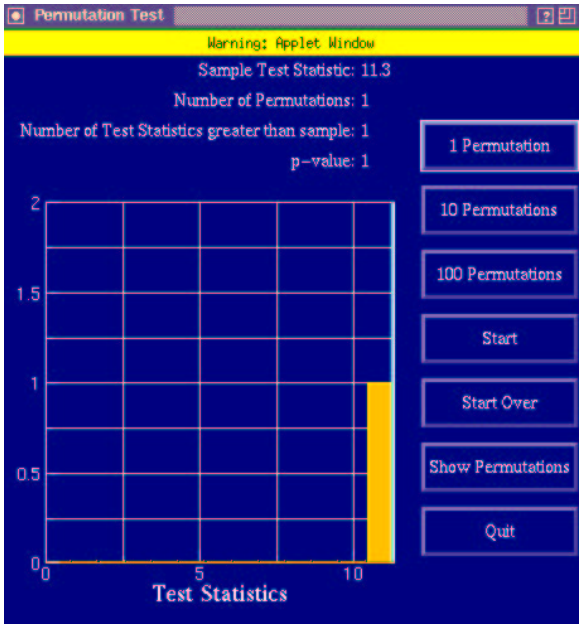

Figure 3: *Initial reference distribution*

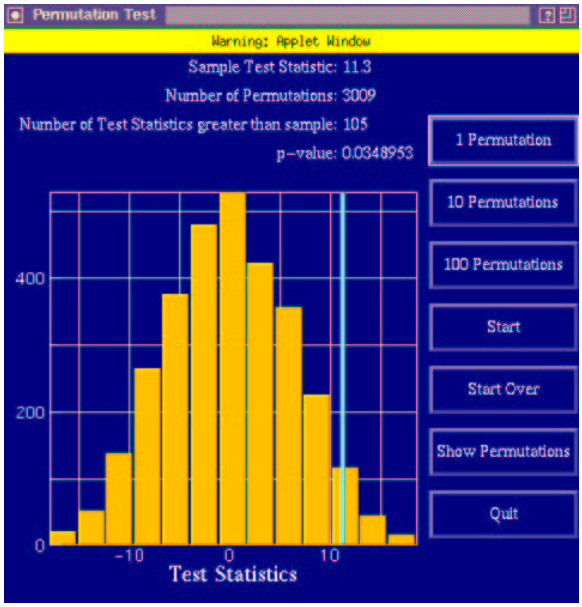

Figure 4: *Distribution after 3009 iterations*

an intuitive framework for learning the concepts of model construction, model diagnostics, and prediction.

Goals of DORUS' implementation of CART:

1. Students will learn what CART does. They will learn the concepts of building a binary tree that recursively partitions space into more homogeneous regions.

2. Students will learn how to use training and validation samples to improve the performance of CART. Critically the student should learn that larger trees will always result in a better fit for the training sample but may have a poorer fit for the validation sample.

3. Students will learn to use tree diagnostics and node splitting and pruning to build an optimal CART.

CART is implemented so that students can easily interact with the model's construction. Tree nodes have three operations: split, prune, and inspect. Split and prune have the respective tree functions and inspect allows the user to see the contents of the node (number of observations, predicted value, optimal split, and diversity). With these three operations the student can learn how to construct a CART model.

At all times the student has control over which nodes to split or prune. DORUS calculates the optimal split for each node and highlights suggested nodes for the student to split or prune. In the special case where the student constructs a CART model with a categorical response and two continuous predictor variables DORUS produces a two-dimensional scatterplot of the data. As the student works on fitting a CART model the scatterplot displays the partition of the data. The combination of the student viewing the splits of the tree model and the partitioning of the two-dimensional plane is ideal for teaching the concepts of CART construction.

As with most statistical models, over-parameterized CART models always fit the training data well. Validation samples often indicate that these models fit poorly. DORUS attempts to demonstrate for the students, as the CART model is being constructed, the dangers of overfitting CART models. The variable selection window requests from the user a percentage

of the sample to be preserved for validating the model. Throughout the construction of the tree the number of terminal nodes versus the diversity measure is plotted for both the training and the validation sample. Some trees with the same number of terminal nodes may have different diversity measurements which will be displayed in the diversity graph.

Tree construction and model validation should lead the student to try and find an optimal tree. The graphics displaying the partition of the sample space and the differences in the training and validation samples allows the student to determine the desirable properties of an optimal tree. With this information the student should be able to learn how to choose the best model, the one that minimizes the validation samples diversity.

Features of CART in DORUS

- Main window displays the tree - initialized to the root node
- The user clicks on nodes. If the node is not split it is split. If it is split its subtree is pruned.
- If the CART model has a binary response with two continuous predictors then a scatterplot graphic is displayed. As the tree is split and pruned this graphic displays the partitioning of the 2D plane.
- At all times the optimal split node and the optimal prune node are highlighted as a suggestion for the user's next tree operation.
- The user may select *Fit full tree* which will grow the tree until all nodes are pure, can no longer be split, or contain too few observations.
- Any node may be "inspected" by the user by merely pointing to a node. The inspector window shows the values of N, the nodes diversity, its subtree's diversity, the node's optimal split, and a response estimate (predicted classification or mean).
- At all times a plot of the number of nodes versus the diversity measure is displayed for the training sample and the validation sample. Two trees may have the same number of leaves but have a different diversity measurement.

CART is currently implemented for continuous or binary response variables and continuous or categorical predictor variables using residual sum of squares as the diversity measure.

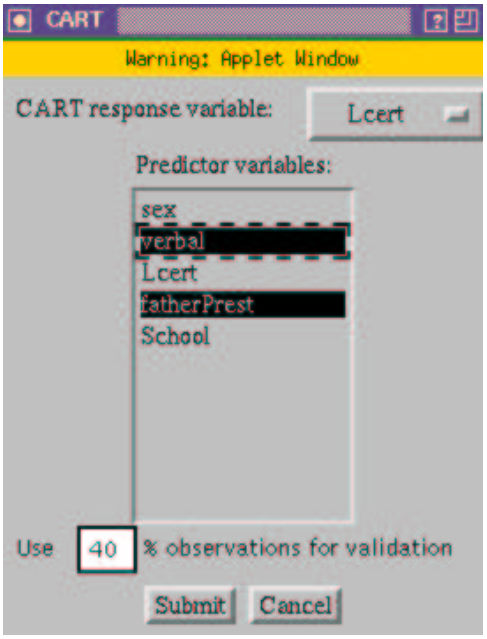

Figure 5: *Variable selection for CART*

Future work will add multiclass classification, various diversity measures, and implementation of the minimal cost-complexity tree selection algorithm.

5.2.1 CART example - Irish education data

The following example uses DORUS' implementation of CART to fit a CART model to data from a study on the equality of opportunity in Irish education [3].

The response variable, Lcert, is a binary response variable (yes or no) whether a student took a leaving certificate. Two continuous variables, prestige score of father's occupation and Drumcondra verbal reasoning score, predict Lcert in this model. Of the 500 observations, 40% are reserved for a validation sample. This example demonstrates the tree's display (Figure 6), the scatterplot of the partition of the plane (Figure 7), and the plot of diversity for validation and training samples (Figure 8).

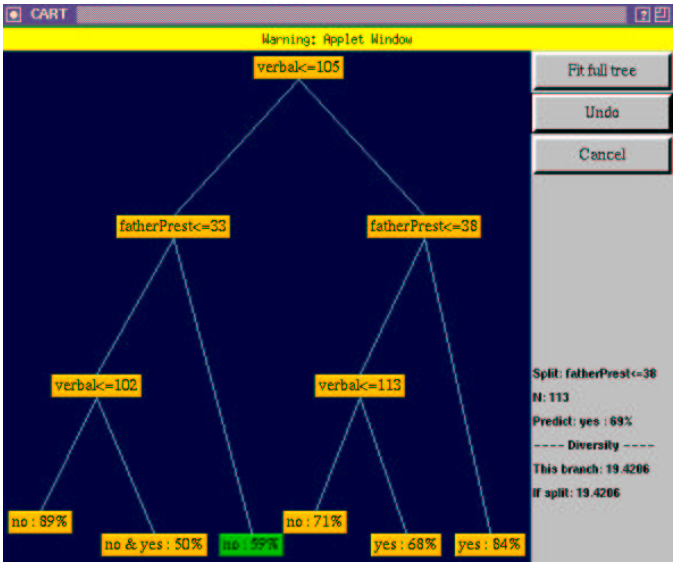

Figure 6: *Classification tree display*

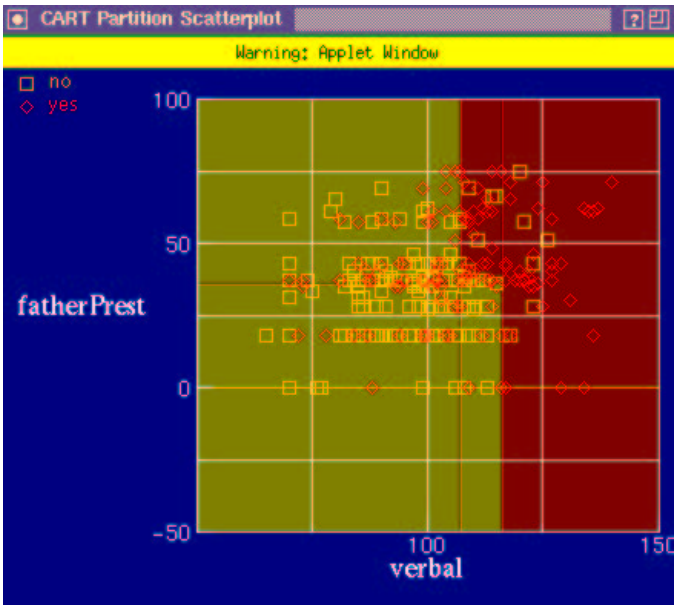

Figure 7: *Optimal partition of the plane*

5.3 Univariate statistics

Univariate statistics have not yet been implemented. The implementation will most likely be bundled into one analysis tool modeled after SAS' `proc univariate`.

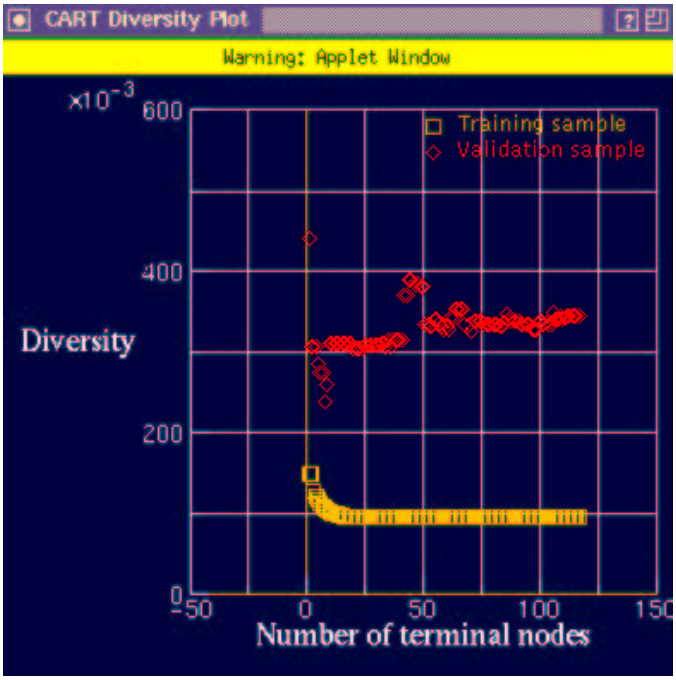

Figure 8: *Training and validation sample RSS*

5.4 Graphics

At this point only two dimensional scatterplots have been implemented. DORUS eventually will have an interactive graphics environment for better data visualization and exploratory analysis. DORUS will almost certainly have methods for brushing and spinning for multi-dimensional data visualization.

6 Conclusions and future work

The new curriculum proposed to present the statistical concepts of inference, modeling, and data visualization in an intuitive way for introductory statistics students. Permutation tests, CART, and exploratory graphics meet this need. This switch from traditional methods motivated the creation of DORUS. In order to be effective DORUS had to be accessible, instructional, and scalable, a goal which was met by coding the package in Java and distributing it via the Web.

DORUS still is under construction and several important features have not yet been implemented.

- Handling of missing values
- Redesign the graphics package to provide more flexibility
- More extensive exploratory graphics (brushing, spinning, high-dimensional graphics)
- Enhanced linear regression environment
- Univariate descriptive statistics
- Confidence intervals via permutation
- Storage of user created datasets
- A method for printing output
- Creation of new variables from univariate and multivariate functions of existing variables
- An online help system
- Assist users in visualizing the inferred variable relations
- Multiclass classification in CART
- Various diversity measures for CART
- User trials

Despite these necessary improvements, DORUS has the potential to be a model for means of data analysis and new forms of distance learning programs. Further modifications and improvements upon the current version of DORUS and new break-throughs in technology will improve DORUS' likelihood of being such a model.

Appendix

A Technical details

DORUS is property of the University of Washington. All Java code was compiled using Simon Leinen's port (February 1996) of Sun's Java Development Kit 1.0 on a Silicon Graphics workstation operating under IRIX 5.3. Scatterplots and histograms implemented in DORUS are subclassed from Leigh Brookshaw's Graph class library 2.1 developed at Lawrence Livermore National Laboratory (February 1996). The histograms for all permutation testing were developed by Kurt Partridge from the University of Washington's Department of Computer Science (March 1996). DORUS has been tested using Netscape Navigator 2.0 on SGI IRIX 5.3, DEC Alpha OSF/1 V2.0, SunOS 4.1.3, PC 486 Microsoft Windows 95.

DORUS is case-sensitive.

B Important Web sites

- DORUS:
  `http://www.stat.washington.edu/greg/DORUS/`
- DORUS source code:
  `http://www.stat.washington.edu/greg/DORUS/CART.java`
  `http://www.stat.washington.edu/greg/DORUS/DORUS.java`
  `http://www.stat.washington.edu/greg/DORUS/PermutationTests.java`
  `http://www.stat.washington.edu/greg/DORUS/ScatterPlotFrame.java`
  `http://www.stat.washington.edu/greg/DORUS/TestStatisticPanel.java`
- Sun's Java Web site:
  `http://java.sun.com/`
- Simon Leinen's port of JDK 1.0:
  `http://liawww.epfl.ch/~simon/java/irix-jdk.html`
- David Liu's (SGI) Expression evaluation package:
  `http://reality.sgi.com/employees/davidliu_mti/java/calc/calc.html`.
- Leigh Brookshaw's Graph class library 2.1:
  `http://www-igpp.llnl.gov/people/brookshaw/java/`

C Stored datasets

Although local file access cannot be implemented over the Web the client can obtain data from the server from which the Java program was downloaded. Therefore, example datasets can be stored for retrieval by the users.

```
Music & efficiency (#1),q1.dat
IQ study (#2),q2.dat
Milk intake (#3),q3.dat
Cocaine (#4),q4.dat
Cocaine-group formatted (#4),q4g.dat
Cynicism & alcoholism (#5),q5.dat
CART test data, CARTtest.dat
Irish education,irish.dat
** end **
```

Table 2: *Example of* `datasets.list`

```
music
Continuous
12
35
18
57
67
40
39
58
44
52
*
nomusic
Continuous
21
35
40
38
23
27
28
39
33
25
*
```

Table 3: *Example of* `datasets.list`

When DORUS is first instantiated on the user's machine it connects back to the server to the directory from which it is stored and reads `datasets.list`. `datasets.list` is formatted as shown in Table 2 with each line containing the datasets title and the file name which contains the data separated by a comma. `datasets.list` must be terminated by `** end **`. The titles are added to DORUS' the Dataset submenu.

Table 3 shows an example data file. Each line contains one item and data for each variable are separated by `*`'s. The first line for each variable contains the variable name, the second contains the variable type, and the remaining lines are assumed to contain the variable's data with one observation per line.

D Data format

Data for unpaired and one-way analysis can be entered in two ways, without group labels (Table 4) and with group labels (Table 5) (example shown is for an unpaired permutation test). `switchableSelectFormat` implements this for the user.

Music group efficiency	No-music group efficiency
12	21
35	35
18	40
57	38
67	23
40	27
39	28
58	39
44	33
52	25

Table 4: *Music and efficiency example without group labels*

Efficiency	Group
12	Music group
35	Music group
18	Music group
57	Music group
67	Music group
40	Music group
39	Music group
58	Music group
44	Music group
52	Music group
21	No-music group
35	No-music group
40	No-music group
38	No-music group
23	No-music group
27	No-music group
28	No-music group
39	No-music group
33	No-music group
25	No-music group

Table 5: *Music and efficiency example with group labels*

E Augmenting DORUS' set of analysis tools

This section will provide instruction for augmenting the source code to include new analysis tools.

1. Attach a `MenuItem` corresponding to the new procedure in `StatFrame.Initialize-StatFrame()`.

2. Add an else-if statement to `StatFrame.handleEvent()` to trap a user selection of the procedure. This event handling should instantiate a new `variableSelection` and add the new procedure to the analyses `Vector`.

3. Subclass `variableSelection` for the new procedure that allows users to select the variables for the analysis. Uniformity with the other layout formats is strongly encouraged.

4. Create a dataObject for efficient storage and manipulation of the data. The dataObjects are instantiated by the `variableSelection` object.

5. Create the classes that perform the analysis. If the new procedure implements a new permutation procedure than the new analysis tool class would subclass `PermutationEngine` and be used to instantiate a new `BuildReferenceDistribution`. Graphical procedures usually can subclass graph.Graph2D.

F DORUS' class and file structure

Class (.class)	Source file (.java)	Description
AboutDialog	DORUS	Displays information about DORUS system
BuildReferenceDistribution	PermutationTests	Window that contains all features for the actual production of the reference distribution histogram
CART	CART	CART's main window
CARTCanvas	CART	Graphic object that contains the tree graphic
CARTDataFormat	CART	CART's data object
CARTScatterPlot	CART	Scatterplot for CART models with discrete response and two continuous predictors
CARTScatterPlotFrame	CART	Window that contains CARTScatterPlots
CARTSelectVars	CART	Variable selection window for CART
CARTtree	CART	Tree structure for CART
CannedData	DORUS	Loads up previously stored datasets
DORUS	DORUS	The DORUS applet that is embedded in HTML documents
Dataset	DORUS	Controls the creation and deletion of variables and offers the analysis tools access to the variables
DisplayPermutation	PermutationTests	Simple output class for displaying the current permuted dataset
MsgBox	DORUS	A generic Dialog box that is used to post messages to the user
OnewayDataFormat	PermutationTests	Data object for one way permutation tests
OnewayEngine	PermutationTests	Permutation engine for one way tests
OnewaySelectVars	PermutationTests	Variable selection window for one way tests
PairedDataFormat	PermutationTests	Data object for two sample paired permutation tests
PairedEngine	PermutationTests	Permutation engine for two sample paired tests
PairedSelectVars	PermutationTests	Variable selection window for two sample paired tests
PermutationEngine	PermutationTests	Abstract permutation engine from which all other permutation engines are subclassed
PermutationTests	PermutationTests	Dummy class - contains no information
QueryBox	DORUS	A generic Dialog box to post messages to the user that requires them to respond with a Yes or No answer.
RenameVar	DORUS	Dialog box used to allow the user to rename variables
RunHistogram	PermutationTests	Maintains threaded control of animating the reference distribution
ScatterPlot2D	ScatterPlotFrame	The 2D Scatterplot graphic
ScatterPlotDataSet	ScatterPlotFrame	Contains the data to be used in the ScatterPlot 2D
ScatterPlotFrame	ScatterPlotFrame	Frame that contains the 2D scatterplot
ScatterPlotSelectVars	ScatterPlotFrame	Variable selection window for 2D scatterplots
SetOrdinalFrame	DORUS	Window to allow users to indicate order of ordinal variables
SimpleLinearRegressionDataFormat	PermutationTests	Data object for simple linear regression
SimpleLinearRegressionEngine	PermutationTests	Permutation engine for testing the significance of the slope parameter for simple linear regression
SimpleLinearRegressionSelectVars	PermutationTests	Variable selection window for simple linear regression
StatDataSet	TestStatisticPanel	Dataset that contains the test statistic values for all the permutated datasets for building the reference distribution
StatFrame	DORUS	The main screen that controls all of DORUS' tasks
StatGraph	TestStatisticPanel	The reference distribution graphic
StatVLine	TestStatisticPanel	Plots a vertical line for the value of the test statistic
StatVar	DORUS	DORUS' variable object.
TestStatisticPanel	TestStatisticPanel	Panel for containing and controlling the reference distribution graphic
UnpairedDataFormat	PermutationTests	Data object for a two sample unpaired test
UnpairedEngine	PermutationTests	Permutation engine for a two sample unpaired test
UnpairedSelectVars	PermutationTests	Variable selection window for two sample unpaired tests
alternative	PermutationTests	Allows the user to select the alternative hypothesis
choiceOfVars	DORUS	Creates a choice object containing variables in the Dataset.
listOfVars	DORUS	Creates a list object containing variables in the Dataset.
std	DORUS	Standard codes that are used throughout the DORUS system. This includes default fonts, special character strings, and numeric identifiers
switchableSelectFormat	PermutationTests	A special variable selection class when data for analysis tools may be have several possible formats
treeNode	CART	Nodes of the classification and regression tree
variableSelection	DORUS	Abstract class that all variable selection classes subclass

References

[1] Breiman, Friedman, Olshen, and Stone. *Classification and Regression Trees.* Wadsworth International Group, Belmont, Calif, 1984.

[2] Cobb, G. W. (1992), "Teaching Statistics," in Heeding the Call for Change, ed. Lynn Steen, MAA Notes No. 22, Washington: Mathematical Association of America, pp. 3-23.

[3] Greaney, V. and Kelleghan, T. (1984). *Equality of Opportunity in Irish Schools.* Dublin: Educational Company.

[4] Hogg, R. V. (1990), "Statisticians Gather to Discuss Statistical Education," Amstat News, No. 169, 19-20.

[5] Mosteller, F. (1988), "Broadening the Scope of Statistics and Statistical Education," The American Statistician, 42, 93-99.

[6] Snee, R. D. (1993), "What's Missing in Statistical Education?" The American Statistician, 47, 149-154.

[7] Yilmaz, M. R. (1996), "The Challenge of Teaching Statistics to Non-Specialists" Journal of Statistics Education v.4, n.1.