

# Statistical Analysis of Student Marks

*Daniel Heath*

**Student ID: 19454643**

***Advanced topics in Second Year, Semester 2, 2006***

Contact: daniel.r.heath@gmail.com

## **Abstract**

A longstanding problem in the education of computer science is that lecturers do not have time to invest in a statistical analysis of their students marks. An analysis of student marks data may enable them to improve the teaching methods in their course.

This project provides a graphical application to perform analysis on student marks data. Hopefully this will make such an analysis significantly less time-consuming.

## **Goals**

- Provide a usable, extensible tool for lecturers to analyse student data without requiring an indepth knowledge of statistical methods or a significant time investment.
- Make the installation of the tool as simple & quick as possible
- Learn a new language

## **Introduction**

The purpose of this report is to outline what methods were used in writing a program for statistical analysis of student marks and describe the functionality of the program.

## **Research**

Due to my limited background in statistics, research was done to establish the most relevant statistical methods for the task of analysing student marks. It was found that the most commonly used techniques in the computer science education field included correlation coefficients, scatterplots and frequency histograms, and all of these have been implemented.

## **Design**

I decided to use Python ([www.python.org](http://www.python.org)) with the Tkinter library both for cross-platform support and the availability of documentation, and to use the model-view-controller design pattern to separate tasks into related files. This allows new features to be added easily, by categorising the feature into a data model feature, an interface feature or a program logic feature.

## **Implementation**

The final build of the project includes support for manipulating statistical variables (which represent the marks for individual assesment tasks) as well as analysing them. This allows users to build a

new variables. For example, by replacing the attendance data for tutorials (made up of strings) with numeric data, then summing the numeric attendance variables. This could then be compared with exam performance to examine whether or not the tutorials have any positive effect.

## Production

Finally, the module has been packaged with its dependencies, greatly facilitating installation.

## Screenshot

Correlation describes how well performance on A predicts performance on B. Correlations greater than 0.5 or less than -0.5 are, respectively, strong direct or strong inverse correlations

Histograms are bar charts where each result achieved by a student is shown with the number of times it appears in the input file.

These drop-down lists are used to select which variables (assessments) to analyse.

Histogram Quantizing allows you to see a histogram of the marks earned by students grouped into sets. The step size sets the range of each set, so large step sizes yield histograms with few sets.

The replace frame enables you to replace all occurrences of one value with another in one or several variables. One use of this feature is to compare attendance records with final results; replace all occurrences of "absent" with zero and all occurrences of "present" with 1 in tutorial marks, then use the Sum frame below.

Scatterplots graphically show the distribution of two variables against each other. Pairwise Delete removes invalid data completely. Zero Substitution replaces invalid data with zeroes. Mean Substitution replaces invalid data with the mean value of the valid data.

The basic statistical properties of the data are displayed. These are mostly of interest when examining a single assessment.

The Sum frame can add several variables together. The new variable (with the name specified) has the numeric totals of each variable selected in the listbox below. Pairwise Delete removes all incompatible records. This makes some comparisons meaningless as it removes some data. Zero Substitution replaces non-numeric records with zero, thus keeping the new variable the same length. Mean Substitution replaces non-numeric records with the mean of the numeric records. This artificially reduces the deviation of the data.

	Prac 2	Prac 4
Number of Numeric Val's:	91 of 100	93 of 100
Min:	1.5	2.5
Max:	12.0	12.0
Range:	10.5	9.5
Mean:	7.86813186813	9.37096774194
Median:	9.0	10.0
Mode:	11.5	2.5
Std. Dev.:	2.8726018807	2.67093397724
Variance:	8.25184156503	7.13388831079

## **Discussion**

### **User Interface**

The primary difficulty in this project is the design of the user interface to allow for new components to be added easily. The user interface went through several revisions.

Initially, I attempted to provide graphing functionality as a 'built-in' before realising that this was not at all extensible. I replaced my crude graphical display with an interface to the freely available gnuplot ([www.gnuplot.info](http://www.gnuplot.info)). This change made the program more extensible as new types of graphs could be easily added.

I later moved each of the user interface 'widgets' into containing frames, grouping related elements together. This meant that related interface elements could be moved around as a coherent set. While this was originally done to help me implement the rapidly-changing design of the user interface, it also makes the program overall more extensible. It has the further advantage of providing a clear separation to the user between unrelated items.

### **Project Management**

For a project involving a single person, one would expect that such technologies as source control would be unnecessary. By the end of this project, I have gained a keen awareness that this is not at all the case. Revision control is less essential, but it is still an excellent resource – for instance, it helps eliminate the clutter of old code (long since commented out) since you can safely delete any part you are not currently using.

### **Learning**

Through this project I have learned how to use the python language to build GUI applications as well as underlying logic.

I have also become far more able to interpret statistics through the research undertaken as part of the project.

I have practiced modern design patterns, and learned about interfaces between programs.

### **Future Recommendations**

In future, additional types of analysis could be added to this package. In addition, the builtin documentation could be expanded to include a more complete description of functionality.

The installation procedure could be made significantly more streamlined.

Finally, more graphical displays of data could be added through gnuplot.

### **Summary**

This statistical package is not a complete solution; ideally, the software could be installed far more quickly. It is, however, a useful tool which may help lecturers improve their course, and it can be extended to include many more functions.

### **Acknowledgements**

Many thanks to my supervisor Robyn MacNamara of the Monash University Computer Science department for her assistance not only in this project but as a tutor throughout my university life.

### **Bibliography**

- # That book robyn lent me on tkinter.
- # Wikipedia for correlation algorithm
- # That journal of articles about CS education.

