COMP114 - Assessment 2
Semester 2 – 2011
Software Upgrade Performance
Applying t-tests

Assessment Information

Assessment Number	2
Contribution to Overall Mark	20%
Submission deadline	**Friday 25th March 2011, 16.00**

Relevant Learning Outcomes for this Assessment

1	Awareness of different areas of CS for which experimental methods are relevant.
3	Understanding of the factors involved in constructing experiment settings.
5	Ability objectively to assess, analyse, and present experimental results.

Assessment Description – Overview

This assessment involves conducting the experiment outlined in the lecture notes concerning the performance of an "upgraded" antivirus system ($AV - New$) relative to its predecessor ($AV - Old$).

It will involve carrying out the following analysis with data supplied via the source described in the Detailed description:

E1. Determining if the actual *run-time difference* between the old version and upgrade is "*significant*" and/or "highly significant".

E2. Determining if the actual number of *detections* made by the upgrade is significant and/or highly significant.

Assessment Description – Details

A Java class

public class AntiVirusPerformanceData

can be downloaded from the module Web pages. [**Note:** The *source code* is **not** provided (and is not required for this assessment).] You should copy this to a file named

AntiVirusPerformanceData.class

in your local filestore.

This class will supply a collection of reports provided by different users when running both old and upgraded versions, i.e. similar to the structure $\{\langle AVO(1), AVN(1)\rangle, \ldots, \langle AVO(n), AVN(n)\rangle\}$ described in the lectures.

The class $AntiVirusPerformanceData$ contains a single constructor

AntiVirusPerformanceData(**int** n)

This provides four fields

Name	Description
private int[] OldTime	The n Run time values for old program
private int[] NewTime	The n Run time values for the upgrade
private int[] OldCover	The n figures giving the number of viruses found by the old AV program
private int[] NewCover	The n figures giving the number of viruses found by the upgrade

There are four instance methods

Name	Description
public int[] GetOldTime()	Returns the array OldTime[] for this instance.
public int[] GetNewTime()	Returns the array NewTime[] for this instance.
public int[] GetOldCover()	Returns the array OldCover[] for this instance.
public int[] GetNewCover()	Returns the array NewCover[] for this instance.

This assessment requires you to use class $AntiVirusPerformanceData$ in a Java program $UpgradeEvaluation$ which carries out the following:

1. Read a value n from the standard input. This value can be an arbitrary integer (> 0) and will correspond to the number of sets of user data that are available.

2. Create an instance Results = **new** AntiVirusPerformanceData(n) of the class $AntiVirusPerformanceData$.

3. Using the data given in $Results$ determine using relevant values of $t(n\ d.f.)$ for significance levels of 5% and 1% the conclusions that could be drawn for each of the experiments described in the overview.

Hints and Suggestions

Computing averages etc.

In order to obtain the experiments' result you will need to calculate a number of numerical values. In particular,

Value needed	Suggested variable name
The collection of *Time differences* NewTime[i]-OldTime[i]	**int**[] TimeDiff
The collection of *Coverage differences* NewCover[i]-OldCover[i]	**int**[] CoverDiff
Average value of the *time difference*, i.e. of TimeDiff[i]	**double** MeanTimeDiff
Average value of the *coverage difference*, i.e. of CoverDiff[i]	**double** MeanCoverDiff
The *standard deviation* of TimeDiff[i]	**double** SDTimeDiff
The *standard deviation* of CoverDiff[i]	**double** SDCoverDiff
The *Estimated Standard Error* for the Time Difference	**double** SETimeDiff
The *Estimated Standard Error* for the Coverage Difference	**double** SECoverDiff
The t-test value MeanTimeDiff/SETimeDiff	**double** tTestSpeed
The t-test value MeanCoverDiff/SECoverDiff	**double** tTestCover

Although you are free directly to implement the required calculation of these quantities if you wish (there is no extra credit assigned to doing so), you will find it simpler to copy the file

BasicStats.class

from the module Web pages to (an identically named) file in the directory where your code is being developed. This package provides three methods -

static double AverageValue(**int**[] S, **int** n)
static double StandardDeviation(**int**[] S, **int** n)
static double StandardError(**int**[] S, **int** n)

The first of which computes the average value of $\{S[0], S[1], \ldots, S[n-1]\}$; the second of which computes the standard deviation of the same collection of values.

The final method, computes the *estimated standard error* for this same set of values, i.e. the value StandardDeviation$(S, n)/n^{0.5}$ as described in the notes. Thus, the value of e.g. $SETimeDiff$ is obtained by

SETimeDiff = BasicStats.StandardError(TimeDiff,n)

Finding the value of t to use

A Java Applet from which the value $t(\ n\ d.f.)$ can be found given n and the significance level ($p = 0.05$ for 5%; $p = 0.01$ for 1% and $p = 0.001$ for 0.1% is given on the module web pages. Simply enter n and p into the relevant boxes and click the button marked "*Calc t*".

Note: You may find other methods for determing the value of $t(n\ d.f.)$ are available on-line (or via books). While it is perfectly acceptable to use these, you are *strongly advised* to confirm that the value of t reported by these is the same as that reported by the script provided. There are a number of technical reasons why these may appear to be different, and, if in any doubt, you should use the value given by the t-test script.

What should be Submitted

a. The source code of the single Java program carrying out the t-test experiments described.

b. The 4 tables of results reporting the outcome of the experiments (E1) and (E2). Each table should be structured as below:

n	$t(n\ d.f.)$ (significance)	t-test value (from program)	Reject N.H?
10	$t(10\ d.f.)$	???	Yes/No
20	$t(20\ d.f.)$	???	Yes/No
30	$t(30\ d.f.)$	???	Yes/No
40	$t(40\ d.f.)$	???	Yes/No
50	$t(50\ d.f.)$	???	Yes/No
100	$t(100\ d.f.)$	???	Yes/No
150	$t(150\ d.f.)$	???	Yes/No
200	$t(200\ d.f.)$	???	Yes/No

Recall that both (E1) and (E2), should be run with 5% *and* 1% significance levels (i.e. $p = 0.05$ and $p = 0.01$ in finding $t(n.\ d.f.)$

c. A summary of your conclusions from the experiment.

d. A completed and signed *Declaration On Plagiarism and Collusion Form.*

How the work should be submitted

Items (a–d) should handed in to the *Student Office.* A cover sheet should indicate all of the following information:

a. The Assessment *number*.

b. Your **name** and University **e-mail address**

c. Your lab group.

d. The name of the *demonstrator* responsible for this group.

e. Your degree programme, e.g. G400 Computer Science, G500 Computer INformation Systems.