Regression Suite README

Lakshmi Krishnamurthy

v1.0, 7 June 2012

Introduction

Overview

RegressionSuite is a software test suite that incorporates measurement of the startup lag, measurement of accurate execution times, generating execution statistics, customized input distributions, and processable regression specific details as part of the regular unit tests.

Essentially, RegressionSuite provides the frame-work around which the individual unit regressors are invoked (and details and statistics collected). Unit regressors are grouped into named regressor sets (or modules), and regressors are created by implementing specific regressor interfaces. These two features make regressors particularly amenable to testing analytics (and other similarly state-light) modules.

RegressionSuite Design Objectives and Features

The key objective behind the design of the RegressionSuite is that it enable/ease incorporation of testing to beyond simple unitary unit testing (pass/fail), and allow for (in this edition) clock time based performance measurements. To that end, the following is a comprehensive suite of design goals and features that it handles:

- Execution Time Distribution: A first objective is the measurement of the execution time (clock time, as opposed to CPU time – to gauge the response in a given setting/environment) distribution. Measurement is done by isolating the actual regression run from the preparation and processing (more on this later). Further start-up is treated separately from the routine run, again with a view to measurement isolation. Central statistical measures such as mean and variance are generated – so are the extremal measures (minimum and maximum execution times).

- Enhanced Unit Testing: The next goal is to enhance the unit testing by extending the typical unit regression tests in a couple of ways. First, RegressionSuite is intended for generating a variety of inputs to span the full range (this can, of course, cause it to be prohibitively expensive for practical use in many situations – strategies for input variance reduction may be employed as suitable – as in identifying the parameter-correlated valid input ranges). Next RegressionSuite also needs to distinguish between situations where the unit tests fail, but the regression is deemed to have succeeded (e.g., over specific inputs). Further RegressionSuite will be extended to capture both the execution time distribution and the unit test success/failure over the given input ranges. As noted earlier, measurement of execution times makes most sense if done by excluding the preparation and output processing times, unit tests need to account for that as well (more on this on the frame-work discussion).
- Pluggable and automated frame-work: RegressionSuite is built using interfaces and frame-works, yet with simple process controls, invocation freedom (minimal constraints), and without side-effects. Both the regression set/modules as well as the unit regressors are built around interfaces, and the core invocation/execution functionality is delegated to the frame-work. The frame-work also generates the regression statistics and other details.
- Regression Details: Details are as elaborate or brief, and their emission is controlled typically at multi-levels – at the frame-work level, at the individual module level, and at the unit regressor level. Field level details are built in by the unit regressor, and there are no limitations on the what constitutes a detail. Being named values, the details are built to be compact/efficient, parseable, and procesesable (XML is not the format of choice). Of course distribution may be generated on any of the number parsable detail fields.

Finally, CreditAnalytics also calculates an elaborate sequence of measures relevant to each product. It is built with an enhanced sequence of standard scenario curves that can be used to generate very customized scenario measures.

Documentation

Apart from the information provided in this user guide, additional documentation of RegressionSuite functionality and release notes may be found in the RegressionSuite website. Consult the javadoc for elaborate API usage information.

Installation and Dependencies

The core module of RegressionSuite consists of just one jar – RegressionSuite_<<version>>.jar. This contains the complete suite of the entire RegressionSuite library. Download and install this in your class-path.

However, in the attached set of samples, regression is done of the CreditAnalytics library – please refer to the Credit Analytics Installation information (http://code.google.com/p/creditanalytics) for information.

Configuration

No special configuration is required to run the RegressionSuite library. Obviously, configuration may be required for the modules that undergo regression.

In the attached set of samples, regression is done of the CreditAnalytics library – please refer to the Credit Analytics Configuration information (http://code.google.com/p/creditanalytics) for information.

Getting Started

Once you have downloaded, built, and installed the RegressionSuite, you are now ready to execute the RegressionSuite. The best way to start is by running the attached CreditAnalyticsRegressionEngine sample (make sure you've installed CreditAnalytics in your path to run the latest samples).

The most important calls in the CreditAnalyticsReressionEngine are in its main method. The main initializes the CreditAnalytics library, adds the regressor set, and launches the regression suite. Refer to the javadoc and the comments in the CreditAnalyticsRegressionEngine.java class.

```
       public static void main (final String[] astrArgs) throws Exception {
             CreditAnalyticsRegressionEngine care = new
CreditAnalyticsRegressionEngine (10,
                    REGRESSION_DETAIL_MODULE_AGGREGATED);

             /*
              * Add the regressor sets: Refer to the implementation of the
corresponding regresors
              */

             care.addRegressorSet (new CreditCurveRegressor());

             care.addRegressorSet (new DiscountCurveRegressor());

             /*
              * Launch regression - and that's it!
              */

             care.launch();
       }
```