|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Study** | **Dependent Variable(s)** | **Confounding Variable(s)** | **Data Set** | **Analysis** | **Results** |
| **Li and Henry, 1993** | Maintainability (measured by LOC changed) | None factored in | Two commercial systems written in Classic-Ada | Linear Regression | Concludes that NOC, LCOM, RFC, WMC, DIT all predict maintenance efforts beyond what can be predicted for size alone. |
| **Basili et al., 1996** | Fault-proneness | None factored in | Eight student projects written in C++ | Univariate linear regression. Multivariate logistic regression | Finds that LCOM is not a significant predictor of fault-proneness while the remainder of the CK metrics are. |
| **Harrison et al., 1998** | Size (LOC), testability (time to create automated tests), changeability (time to implement modifications), understandability (Boehm measures) | None factored in | Five small projects written in C++ | Correlation analysis | No results on DIT and NOC as no inheritance in data set. Negative correlation between WMC and the time to create automated tests for software. WMC was found to be negatively correlated with understandability. Both WMC and LCOM were negatively correlated with changeability. |
| **Tang et al., 1999** | Fault-proneness | None factored in | Three small/medium commercial systems written in C++ | Logistic regression | RFC and WMC strong predictors for fault-proneness |
| **Emam et al., 1999** | Fault-proneness | Class size (LOC) | One medium-sized commercial project | Logistic regression | After controlling for size, only CBO was an indicator of fault-proneness |
| **Cartwright and Shepperd, 2000** | Fault-proneness | None factored in | One large commercial project | Linear regression | Found the inheriting classes were more defect prone (identified as classes having a DIT or NOC > 0) |
| **Subramanyam and Krishnan, 2003** | Fault-proneness | None factored in | One large commercial project written in C++ and Java | Linear regression | CBO, DIT, WMC predictive of fault-proneness |
| **Elish and Rine, 2003** | Stability | None factored in | Three medium-sized FLOSS projects written in Java | Correlation analysis | CBO, DIT, LCOM, RFC, and WMC (particularly CBO and RFC) were all found to be negatively correlated with stability. |
| **Bruntink et al., 2006** | Testability | None factored in | Five medium/large-sized projects written in Java | Correlation analysis | Using the lines of test code and the number of test cases in the unit tests as a proxy for testability, they find that only DIT and NOC are predictors of testability. |
| **Xu et al., 2008** | Fault-proneness | Class size (LOC) | One medium-sized government project written in C++ | Neural networks | CBO, RFC and WMC are reliable metrics for defect estimation finding that overall |
| **Badri et al., 2011** | Testability | None factored in | Two medium-sized FLOSS projects written in Java | Correlation analysis and logistic regression | Found a correlation between LCOM and unit test coverage, validating the use of OO metrics as a predictor of the testability of classes |
| **Malhotra and Jain, 2012** | Fault-proneness | None factored in | One medium/large-sized FLOSS project written in Java | Logistic regression and machine learning techniques | Machine learning models comparable in performance to linear models. Found that CBO, LCOM, RFC and WMC not to be significant predictors of fault-proneness. The rest of the CK metrics were indicators. |
| **Saberwal et al., 2013** | Bad code smells (binary determination) | None factored in | One medium-sized FLOSS project written in Java | Logistic regression | RFC, LCOM, NOC and WMC found to be useful predictors of bad code smells. |