## **Correlation between Code Coverage and Post-Release Defect Density**

***We have started our analysis with a hypothesis that, the projects with low code coverage contains more bugs***. Code coverage gives us an idea of the thoroughness of testing by providing information about the amount of code that is tested. So that, increase in code coverage is likely to lead to a decrease in post-release bugs. For computing the Correlation between the statement coverage and branch coverage with post release defect density, We have calculated the spearmen correlation coefficient for the statement coverage and the number of bugs as well as for the branch coverage and the number of bugs as Shown in the Table-1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. no** | **Project Name** | **Statement Coverage** | **Branch Coverage** | **Number of Bugs** |
| **P1** | Apache Commons Collections | 65% | 49% | 3 |
| **P2** | Apache HTTP Components Clients | 69% | 23% | 2 |
| **P3** | JFreeChart | 72% | 35% | 5 |
| **P4** | Apache Commons IO | 90% | 45% | 24 |
| **P5** | Apache Commons Configurations | 86% | 56% | 6 |

Table-1: Showing Statement Coverage, Branch Coverage, and Number of Bugs.

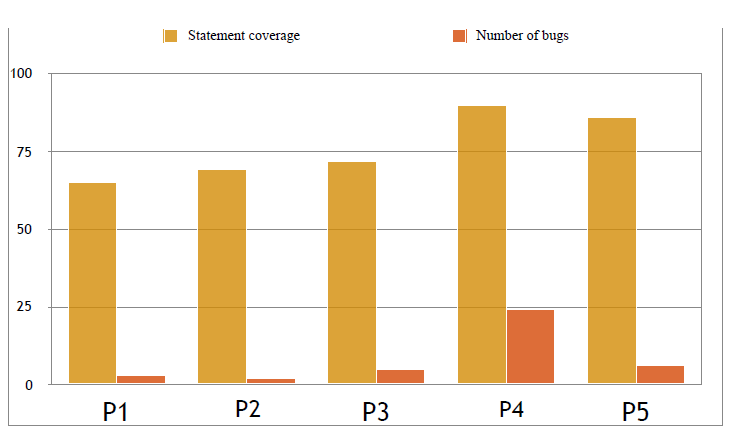


Chart1 : Correlation between Statement Coverage and Number of Bugs.

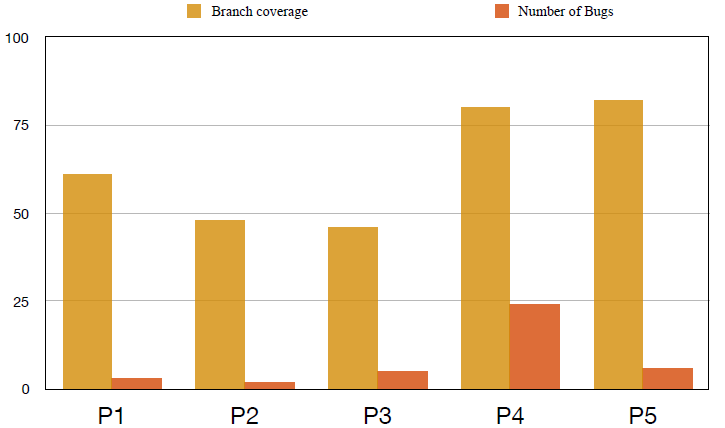


Chart2 : Correlation between Statement Coverage and Number of Bugs.

***Calculating Spearman Correlation***

Spearman’s Rank correlation coefficient is one of the most-prominent technique which can be used to find out the strength and correlation between two variables.

***Method used to calculate the Spearman correlation***

* Create a table from your data and get the ordered pairs of two variables.
* Rank the two data sets. Ranking is achieved by giving the ranking '1' to the biggest number in a column, '2' to the second biggest value and so on. The smallest value in the column will get the lowest ranking. This should be done for both sets of measurements or the variables used to find the correlation for.
* Tied scores are given the mean (average) rank.
* Find the difference in the ranks (d).
* Square the differences (d²) To remove negative values and then sum them
* Calculate the coefficient (***Rs***) using the formula mentioned below.

When written in mathematical notation the Spearman Rank formula looks like this:



Here,

ρ= Spearman rank correlation

di= the difference between the ranks of corresponding variables

n= number of observations

We have used the formula of spearmen correlation coefficient by considering the statement coverage values as X- values and the number of bugs as Y- values.

***Calculated Spearman Correlation Coefficient values***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Project*** | ***Statement coverage (1)*** | ***Branch Coverage (2)*** | ***Average Cyclomatic complexity (3)*** | ***Spearman Correlation (1&3)*** | ***Spearman Correlation (2&3)*** |
| ***Apache Commons Collections*** | ***86%*** | ***49%*** | ***14*** | ***-0.278*** | ***0.469*** |
| ***Apache HttpComponents Client*** | ***53%*** | ***23%*** | ***10*** | ***0.0458*** | ***0.497*** |
| ***JFreeChart*** | ***72%*** | ***35%*** | ***24*** | ***-0.506*** | ***0.488*** |
| ***Apache Commons IO*** | ***90%*** | ***45%*** | ***16*** | ***-0.400*** | ***-0.0369*** |
| ***Apache Commons Configurations*** | ***91%*** | ***56%*** | ***17*** | ***-0.369*** | ***0.533*** |

Table-2: Result of Spearman Correlation Coefficient

By analysis of the result seen in the Table-2 , we have studied that there is strong correlation between the statement coverage and the number of bugs since if the more statement coverage is achieved , there are less chances of bugs to creep in to the system.

We also studied that there is medium correlation between the branch coverage and the number of bugs since it only considers the branches not the every statement. Hence we have the low value of spearmen correlation coefficient compared to the strong value in the statement coverage and the number of bugs.

Hence our hypothesis of the projects with low code coverage contains more bugs holds true.