Homework #5

Submitted by: Aditi Shashank Joshi (1222838916)

Software reliability is the probability of failure-free operation of a computer program for a specified period in a specified environment. Reliability is a customer-oriented view of software quality. It is dynamic rather than static since it refers to program functioning rather than program design. [1] Software Reliability is an essential connection of software quality, composed with functionality, usability, performance, serviceability, capability, installability, maintainability, and documentation. Software Reliability is hard to achieve because the complexity of software turns out to be high. [2]

Some of the **approaches** that are used to predict software reliability are:

1. Statistical Modeling and Estimation of Reliability Functions for Systems (SMERFS) is a Software Reliability Estimation and Prediction Tool (SREPT). It is a program used for predicting the reliability of some software during its testing phase. It takes into consideration the total number of previous failures when making such predictions.
2. Machine Learning (ML) techniques such as Linear regression (LR), Support Vector Machines (SVMs) and Artificial Neural Networks (ANNs) are used for software reliability prediction [3]. SVM models can be built to find the relationship between reliability and error rate and therefore help in prediction of software reliability. Capability of ANNs to learn and generalize have made them useful in predicting failures in software systems during the testing phase [5].

These tools are **contrast** in the following ways:

1. SMERFS usually employs hypothesis testing and confidence intervals to make predictions, whereas ML models mainly work by learning the input data and making predictions for data that is not yet learned.
2. SMERFS is highly dependent on statistical methods which can be inherently wrong. ML models have very minimal statistical concepts involved.

3. SMERFS have low uncertainty tolerance which is opposite in ML models as they are more flexible.
4. SMERFS can be worked with a small dataset, ML models require large amounts of data.
5. SMERFS models include probability spaces, sampling, and diagnostics. ML model includes Neural networks, Support vector machines, decision trees, Bayesian networks, and genetic algorithms

The **inputs** to the tools are:

1. SMERFS - Time between error occurrences or number of detected errors per testing period
2. ML - The input is usually a dataset for the system.

The **output** generated by the tools are:

1. SMERFS - Output can be estimated of the number of faults, predicted reliability, precision, failure intensity, number of faults remaining etc.
2. ML - The main aim of the ML approach is to improve precision and reliability. The different Reliability Metrics are: Mean Time to Failure (MTTF), Mean Time to Repair (MTTR), Mean Time between Failures (MTBR) etc.

The **assumptions** which the tools use are: [7]

1. The software is always available, there's no downtime for the software.
2. Times between failures are independent
3. A detected fault is immediately corrected
4. No new faults are introduced during the fault removal process
5. Failure rate decreases with test time
6. Failure rate is proportional to the number of remaining faults
7. Reliability is a function of the number of remaining faults
8. Time is used as a basis for failure rate
9. Failure rate increases between failures
10. Testing is representative of the operational usage

References

[1]Software Reliability Anthony Iannino John D.Musa https://doi.org/10.1016/S0065-2458(08)60299-5

[2]https://www.javatpoint.com/software-engineering-software-reliability

[3]https://www.mdpi.com/2079-9292/11/17/2707/pdf-vor

[4]https://aip.scitation.org/doi/pdf/10.1063/5.0080442?casa_token=pRk22jnstIIAAAAA:QkD6CpPWgJbnPpAfPLKQzFFSCUOXjNSBDWbi7Dg5acI-LMZlhMkUob9Q_y3goeG6aHUyc8_xrnc

[5]https://www.igi-global.com/chapter/investigation-of-software-reliability-prediction-using-statistical-and-machine-learning-methods/180949

[6]https://www.sciencedirect.com/science/article/pii/S2352711019302134

[7] Goel, A. L. (1985). Software Reliability Models: Assumptions, Limitations, and Applicability. IEEE Transactions on Software Engineering, SE-11(12), 1411–1423. doi:10.1109/tse.1985.232177