Mini Project 1: predicting logic errors of digital circuits

Background: We are all computer engineers and we design circuits. However, as human, we make mistakes, which can lead to erroneous circuits. For example, an erroneous multiplier may compute 3 * 3 = 8.9 rather than 9. So, can we answer these question: will there be errors for 2 * 3? 1000 * 999? Any arbitrary number X * Y? In other words, can we predict the errors of multipliers for a given input pair?

Method: Fortunately, when speaking about prediction, we have a powerful tool called machine learning. Using machine learning, we can train a predictive model that can predict the logic errors for multipliers. You can use the Scikit learn library for doing this. Note that Sk-learn library contains many useful internal functions such as accuracy_score(), precision_score(), recall_score(), etc.

Training data and labels: For every machine learning problem, we need to identify three things: what is input variable, what is output label, and what should be the ML method?
Input variable: Binary vector. Example: 1000 0001 → {1,0,0,0,0,0,0,1}
Output label: Each bit position. Example: TFTFTTFF → {0, 1, 0, 1, 0, 0, 1, 1}
ML method: Your choice! While we only studies three ML methods now (logistic regression, decision tree, and random forest), I would not limit your choices here. You are free to use any methods.

A glance at the training data:

Training data: 1001000001010010 0100111100011000
Here, each 1 or 0 represent a bit, i.e., a binary value

Training label: 00000000011110001000001000100000
Here, each 1 or 0 represent correct/erroneous (T/F)

We have 150,000 training data and 50,000 testing data.
Training data: https://www.dropbox.com/s/n2m6dzkvmyu33ii/training_data?dl=0
Training label: https://www.dropbox.com/s/8njlz6urqjbc1y6/training_label?dl=0
Testing data: https://www.dropbox.com/s/h700siqb5tn9q7w/testing_data?dl=0
Testing label: https://www.dropbox.com/s/ueympowbsrqq5ot/testing_label?dl=0

For each multiplier, we have 32 output bits, so we need to establish 32 machine learning models. That is, f(data) → bit_0, f(data) → bit_1, …, f(data) → bit_31. Once you select a model, you will use this model for all (32) bit positions.

Your delivery:

1. Your source code.
2. A report, including at least how you design your classifier, what is your final mean accuracy/precision/recall for all bit positions, running time, your running output screenshot.
3. Both code and report sent to xjiao@villanova.edu and dma2@villanova.edu.

Grading:

1. Your program quality (executable/readable): 50 points
2. Your report quality (clear/organization/necessary components): 30 points
3. Your ranking based on mean accuracy for all bits: 20 points. First get 20, second get 19, third one get 18, …, last one get 12.

Deadline:

1. By Oct.25, 11:59pm.
2. Send your report and python code to xjiao@villanova.edu and dma2@villanova.edu.