I implemented a decision tree regression algorithm on a univariate dataset without using any libraries.

The algorithm was implemented using the pre-pruning rule, such that if a

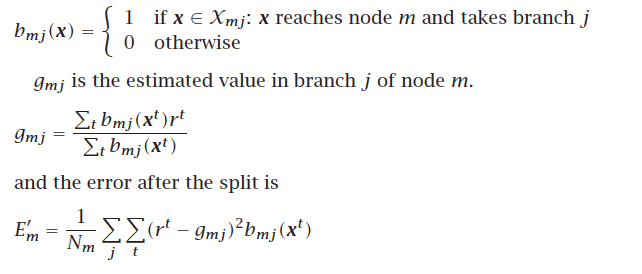
node has 𝑃 or fewer data points, it is converted into a terminal node. Using pre-pruning rule is one of the methods to prevent over-fitting.

In Python, due to having indexes starting from 0, the following indexing rule was used and all data structures were made consistent with this:

For left node: 2\*parent + 1 (instead of 2\*parent)

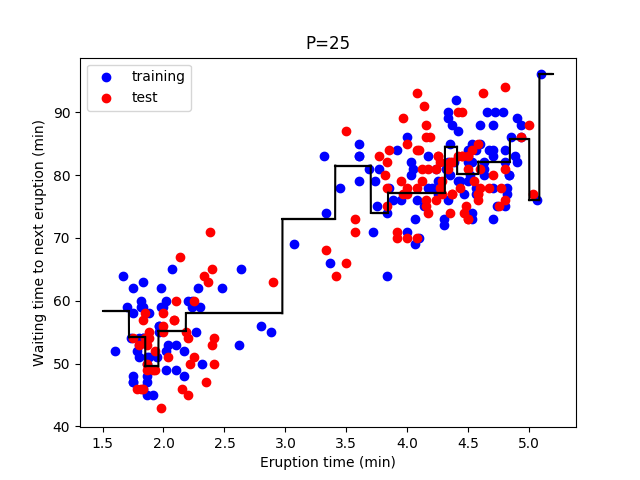
For right node: 2\*parent + 2 (instead of 2\* parent + 1)

During the learning process, at each step we generate all possible split positions and then pick the best one based on the score function given below:



Based on the selected split, then we generate the left node and the right node. The algorithm goes on until there are no nodes to split or we reach the P value for remaining nodes.

The decision tree is visualized for P=25 as shown below:



Y\_predicted is also calculated for X\_test and the RMSE is calculated to compare y\_predicted and y\_test. The error was around ~6.45 when P=25.

The algorithm was then run on varying P values and the corresponding RMSE were compared. We can see from the following graph that RMSE is minimal when P is between 30 and 40.

