We present a novel difficulty rating metric and three puzzle generation algorithms for the popular Sudoku puzzle. Our metric focuses on the human difficulty of a puzzle, rather than an arbitrary computational metric. By using a large set of average human solving time data, we are able to predict human solving time of a puzzle with $R^2 = 0.71$. Furthermore, by giving predictions on a continuous scale, the metric is both extensible and versatile. Thresholding can be used to define any number of arbitrarily sized difficult levels. Our technique also more provides a more relevant measure of difficultly – rather than requesting a 'Diabolical' puzzle, one can request a puzzle that will take 30 to 40 minutes to solve for an average solver.

We use bootstrapping to demonstrate the effectiveness of our metric on previously unseen data and bagging, or bootstrap aggregating, to improve the stability and accuracy of regression models, reducing variance and helping to avoid overfitting.
We also present 3 puzzle generation algorithms: a library based generator, a generator that 'builds up' a puzzles from a solution grid, and a template based generator.

The first precomputes the difficulty of a large set of puzzles. It can then provide puzzles of a requested difficulty extremely rapidly in O(1) time. The second uses Latin Squares to construct a valid solution grid, and then 'builds up' the puzzle leading to that solution from a blank grid. It ensures the puzzle has a unique solution and is of the appropriate difficulty. The third can be used to create puzzles with a desired initial configuration, which mimics popular human-constructed Sudoku puzzles. It can also be used to explore the space of puzzles with initial configurations believed to yield particularly difficult puzzles.

Each of these algorithms, particularly the first two, rapidly provides puzzles with a broad range of difficulties.

Finally, we propose a efficient method of transforming a single valid puzzle into many. The set of valid Sudoku puzzles is closed under several transformations, including row permutation within a band, for example. Using these transformations, we can convert a single puzzle into over 1 trillion, further expanding the efficacy of our generators.