2. Examples are represented by a matrix. The matrix has as many rows as the number of attributes. Each column is an example. We chose this design because the matrix in Python is stored in row-major order. If we want to create a decision tree, we want to access the information in each attribute. Therefore it makes more sense to store data of each type of attribute in the same row.

3. For each node, we calculate the information gain (minimum entropy) for a split in each attribute. We pick the attribute with largest information gain. For each node in the decision tree, we store several information: the attribute we chose to split the node, the threshold for that attribute, the left node, right node, and parent node.

4. For the missing attributes, we replace them with the average of all other attributes of the same type. So for example we replace “?” in winning rate with 0.5 if the average winning rate is 0.5

5. We search the lowest point of entropy by calculating the entropy on both side of a point and moving to the side with lower entropy. The termination for each node split is when we reach a local minimum in entropy. The whole training process terminates when every leaf is pure, that is every leaf contains only one type of final result, either 0 or 1.