CS771 Assignment 2
Team: Spectre

Group Members:

Raghav Karan - 200749
Subhrajit Mishra - 201006
Suryanshu Kumar Jaiswal - 201025
Harsh Chaudhary - 200410
Aastik Guru - 200007

1 Overview of the problem

1.1 Basic description of the problem

Word-guessing game is played on dictionary of known N-words. Here is the step used in game:

- Melbot chooses one of the word of dictionary
- Melbot tells Melbo the length of the string by sending blank string
- If melbo guesses index outside range of 0 to N-1 then game is terminated
- If word is guessed correctly then win count is incremented

Description when Correct word is not guessed and query index lies in range of 0 to N-1

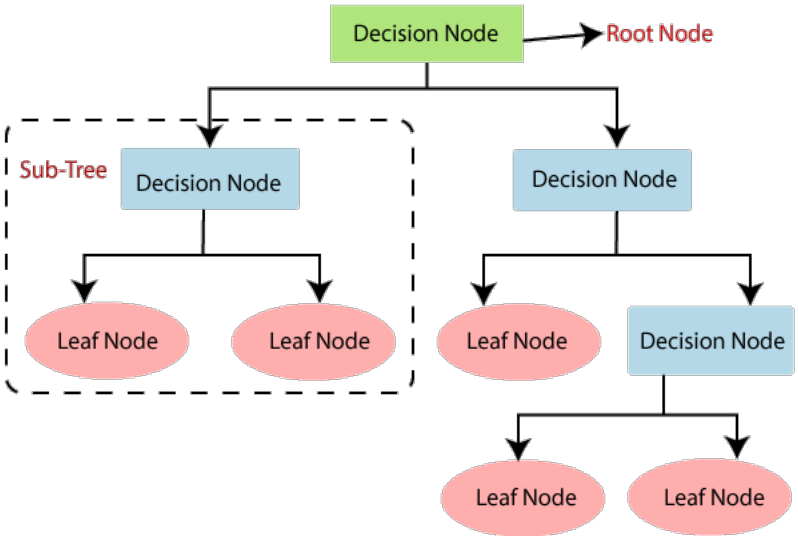
- Melbot reveal all the character matching to secret character
- The location of secret word is also important

2 Model being used: Decision Tree

2.1 What is decision tree?

A decision tree is a type of machine learning model that uses a tree-like structure to represent a series of decisions and their possible consequences or outcomes. It starts with a single node, called the root, which represents the initial decision or input. The tree then branches out into different paths, each representing a different decision or outcome based on certain conditions or criteria. As new data is fed into the model, it follows these paths to make predictions or classifications based on the learned rules and patterns represented by the decision tree.

Preprint. Under review.



2.2 Can decision tree solve our problem?

Answer is YES.
These are the reason which make DT best to solve our problem.

- Decision trees are a natural fit for problems with multiple possible outcomes that depend on a series of decisions.
- Decision trees are easy to interpret and visualize. The structure of a decision tree allows users to easily see the logic behind the decisions being made and understand how the outcomes are reached.
- Decision trees can be trained on a set of data to improve their accuracy over time.
- Decision trees can be used in combination with other algorithms and techniques, such as heuristics or machine learning, to further improve their accuracy and effectiveness.
- Decision trees are extremely useful in creating models with the use of non-numeric data (As in this problem statement).

2.3 Entropy and Calculation of Information gain

If a set have N entity then Entropy of that set can be defined as: $log_2 N$. More difference in entropy of parent node and children implies more information gain.
Let node containing N entity are divided in the k child each having N_1, N_2, ...,N_k elements.
Entropy of children:

$$= \Sigma_{i=1}^{k} \frac{N_i}{N} log_2 N_i$$

Information gain Can be calculated as:

$$\text{Information gain} = log_2 N - \Sigma_{i=1}^{k} \frac{N_i}{N} log_2 N_i$$

A good question would mean maximum information gain. We would target to gain maximum information at every node.
It can be seen evidently, that if a node contains a single word then its entropy is zero. Which means no more information can be gained from that node.

3 Creation of the Model

3.1 Implementation of the Decision Tree

1. The parent node of the decision tree contains all the words available in the dictionary. Let the number of words in the dictionary be 'N' (N=5167, given).

2. In total there will be N leaf nodes of the tree. Each node containing a single word from the dictionary. So the expansion of the tree stops when a node contains a single word or the depth of a node becomes equal to 15.
3. The query word to be guessed at each node is the one which gives maximum information gain i.e. Maximum difference between entropy of parent and entropy of children nodes.
4. The children of a particular node are created on the basis of possible outputs that the word can give. (Example : Suppose a node tells that max information is gained by guessing the word "cat" then there will be 8 children for the particular node. The children would be based on the categories ___, c__, _a_, __t, ca_, _at, c_t, cat)
5. The words will be distributed accordingly in respective child nodes. (Example : words like cup, clap, couple will enter the child node of the c__ category. Similarly bat, rat, hatch, etc will enter the _at categorized child and so on.
6. Finally the tree is prepared once the training is completed.

3.2 Improvements Used for quicker training :

1. Previously we were iterating on each word available at a node to check for the information gain and provide the best possible guess at the node.
2. The above method helped to reduce average query rate (3.99) but at the cost of a high training time (13.6 seconds)
3. We then improvised upon this by iterating over lesser number of words. The words to be iterated upon are chosen randomly at each node. This might not give the best guess, but it reduces training time drastically.
4. If a node contains more than 500 words then 10% of the words are used to iterate over and find the best possible guess. For nodes containing 50 to 500 words, 50 words are chosen at random for making the guess. For further smaller nodes, all the words are checked and the absolute best possible is used.
5. In the current method the average query rate increased to 4.07 but the training time reduces to 1.88 seconds. Since the iterated words are chosen at random, it is completely unbiased.

3.3 Game Play:

1. There is a secret word which has to be guessed by the decision tree within 15 queries.
2. The tree starts from the first node and gives the best possible guess at that node. Melbot gives an output to the guess by unmasking the correct letters at the correct position.
3. This shall be used to move ahead in the decision tree. There will be a child to the current node which will have all the words which lie under the category of the output provided by Melbot. This will be our next node.
4. The process continues till we either finally reach a leaf node or the output provided by Melbot has all the letters unmasked. The word contained at the leaf node is the required word (if we reach the leaf node).
5. Ambiguity: Consider a case where the guess is not exactly correct but all the letters are unmasked. Example: required word is "cat" and the guess at a node is "catch". In the question, there is no specification provided for such cases. We have considered it to be a correct guess in our Decision tree.
6. Since the population of such cases is a small number, so it will not affect the overall working of the model or tree.

References

1. Numpy Documentation
2. Decision Tree Blog