CS771A

Assignment-2
Group A-top

1 Problem

Melbo has purchased a new toy to improve his vocabulary called the Melbot. The toy lets Melbo play a word-guessing game. There is a dictionary of N words that is known to both Melbo and Melbot. In each round of this game, the following steps are taken:

- Melbot chooses one of the words from the dictionary as secret say villager for this round.
- Melbot tells Melbo the number of characters in that word by sending Melbo the string "`_ _ _ _ _ _ _ _`" (without the quotes).
- The following steps are repeated until the round is terminated by either Melbo or Melbot.
  - Melbo guesses a word from the dictionary by taking an index i between 0 and N 1 and sending it to Melbot as a query. For example, Melbo chooses the index 4972 that corresponds to the word violation.
  - Melbot check's Melbo's query to see if it is a valid one. If the query index is invalid i.e. not in the set [0, N - 1], then Melbot assumes that Melbo no longer wants to continue and terminates the round.
  - If the query index is valid, Melbot checks if Melbo's guess is indeed the secret word and if so, the round is terminated and the win count is incremented by 1.
  - If the query word is not the secret word and Melbo has made too many queries in this round (the limit is Q = 15 queries per round), then Melbot terminates the round.
  - If the query word is not the secret word but Melbo has not reached the query limit, then Melbot reveals to Melbo all characters that are common to the query word and the secret word (only if those characters are in the correct location as well). For example, if the secret word is villager and the query word is violation, then Melbot would return the string "`v i _ l a _ _ _`"(without the quotes).

Melbo plays N rounds of this game, once with each word in the dictionary. Melbo's performance is judged based on the number of words guessed correctly within the query limit (the win count divided by N), the average number of queries asked per round and some other measures described below. Note that at any point, Melbo can ask any word as a query so long as it is in the dictionary. It is not necessary that if Melbo is at a certain node in the decision tree, then only one of the words that reached that node must be asked – words that did not reach that node may also be asked if they help discriminate between words that reached that node.

2 Question-1

Give detailed calculations explaining the various design decisions you took to develop your decision tree algorithm. This includes the criterion to choose the splitting criterion at each internal node (which essentially decides the query word that Melbo asks when that node is reached), criterion to decide when to stop expanding the decision tree and make the node a leaf, any pruning strategies and hyperparameters etc.

Answer:

The code that was given beforehand was taking randomly any word as a splitting criterion for any particular node, which leads to unbalance tree formation, large training time, poor accuracy, and the large depth of the tree. Which further leads to the large model size. So to avoid this, we choose the splitting criterion word at the node on the basis of maximum information gain or minimum child entropy.

Entropy is a measure of the impurity or disorder of a set of data. In decision trees, the entropy is calculated for each subset of data based on the distribution of classes within that subset. A low entropy indicates that the subset is mostly composed of examples of the same class, while a high entropy indicates that the subset contains a mix of different classes.

By minimizing the entropy during the splitting criterion, decision trees will be able to create smaller subsets that are more homogeneous in terms of their class distribution. This results in a tree structure that is better able to accurately classify new data that it has not seen before.

For instance, consider the node having the indices of only 3 letter words, which formed after splitting on the basis of the length of the word at the root. After this, we are left with a dictionary of words [cat, bat, cup, hut, sky, pet].

Considering the following word as a splitting criterion.

Case 1: 'cat'

word	mask1	mask2	mask3	mask4	mask5
cat	_ a t	c _ _	_ _ t	_ _ _	cat
Frequency	1	1	2	1	1

For this splitting, the entropy is 0.3333.

Case 2: 'sky'

word	mask1	mask2
sky	_ _ _	sky
Frequency	5	1

For this splitting, the entropy is 1.9349.

Case 3: 'bat'

word	mask1	mask2	mask3	mask4
bat	_ _ _	_ a t	_ _ t	bat
Frequency	2	1	2	1

For this splitting, the entropy is 0.6667.

Case 4: 'cup'

word	mask1	mask2	mask3
cup	_ _ _	c _ _	cup
Frequency	4	1	1

For this splitting, the entropy is 1.3333.

Case 5: 'pet'

word	mask1	mask2	mask3
pet	_ _ _	_ _ t	pet
Frequency	2	3	1

For this splitting, the entropy is 1.1258.

Case 6: 'hut'

word	mask1	mask2	mask3	mask4
hut	_ _ _	_ _ t	_ u _	hut
Frequency	1	3	1	1

For this splitting, the entropy is 0.7924.

From the above implications, we can say that the maximum information gain or minimum entropy is for the word 'CAT'. So, it will be chosen as the best splitting criterion for the node.

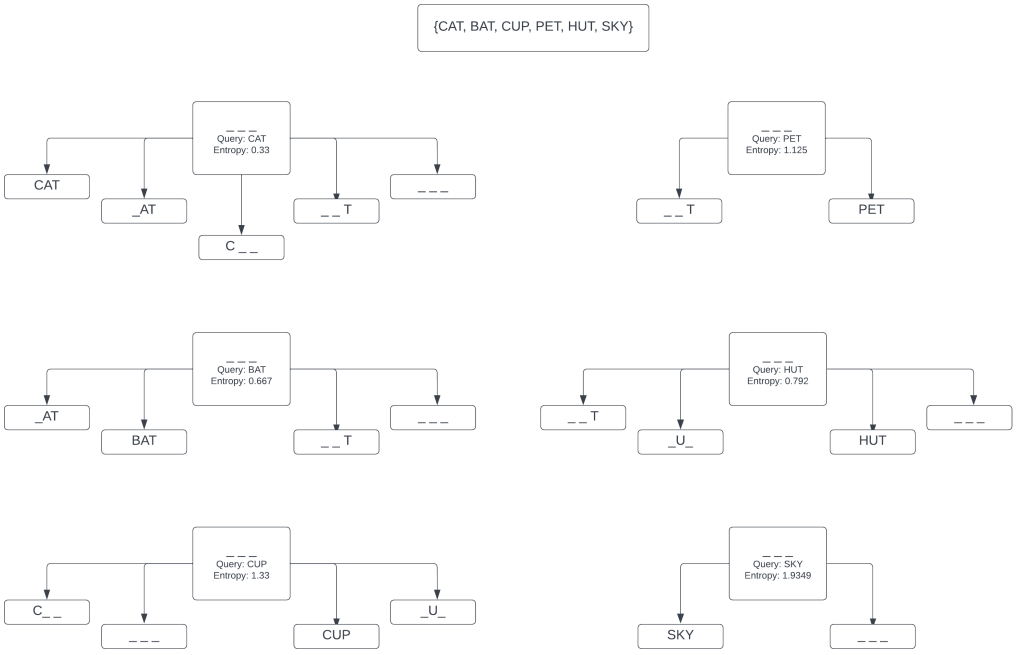

Figure 1: Splitting for example words dictionary

2.1 Other optimizations applied in the code:

- Here, max depth considered for constructing the decision tree is 15.
- Deleted the history variable (containing the list of previously asked queries in a branch) from the code to reduce the memory size. This also reduced the training time of the Decision Tree.
- Randomly selecting elements from the my-word-idx list instead of selecting continuous range of indices.

2.2 Results

Avg. Training Time	Avg. Memory Size	Avg. Accuracy	Avg. Query
1.590s	617860.6 bytes	100%	4.0776