CS771 (Introduction to Machine Learning) : Assignment 2

Riya Saini
200810
Department of Chemical
Indian Institute of Technology Kanpur
riyasaini20@iitk.ac.in

Anubha Tripathi
200163
Department of Electrical
Indian Institute of Technology Kanpur
tanubha20@iitk.ac.in

Harsh Kumar Pandey
200414
Department of Electrical
Indian Institute of Technology Kanpur
harshkp20@iitk.ac.in

Ayush Garg
200241
Department of Electrical
Indian Institute of Technology Kanpur
ayushgarg20@iitk.ac.in

Swayam Gupta
201035
Department of Material Science
Indian Institute of Technology Kanpur
swayamg20@iitk.ac.in

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.

In this word guessing game, melbot chooses one of the word as a secret word using for loop so that each word is chosen once and a query is made. First of all, melbot gives us the length of the word in the form of number of underscores. We take all the words in a dictionary at the root node. We then ask a query using the word sent by get_query() function.

get_query() function takes in a dictionary and calculates entropy and information gain for each child node created out of parent node. These child nodes contain dictionary words split accordingly to work as a parent node used to create a child node in further steps. To make the decision tree balanced, we can select the feature with the highest information gain at each step. Pruning is a technique used to prevent overfitting by removing parts of the tree that do not improve its accuracy. Here at each step, we create two nodes, one that has a dictionary of words which is likely to contain the secret word and other which do not.

In the code, we have tried creating a function find_words(dict_words) which makes us select word after making an informed guess rather selecting them randomly. For making the informed guesses, we will calculate the frequency of each letter as it occurs in the dictionary. This way, we will calculate the information gain and entropy for each word present in the dictionary. We will choose that word as a query that has the least entropy and most information gain.

At each step, let us say we guessed a word through this function find_words(dict_words). Then after guessing, the melbot reveals those characters to us that are same and are also at the same position. after getting that string, we will split the root node into child nodes. For this, we will check each letter of the word and check if it is revealed or not. If it is revealed, then we will keep only those words of the dictionary which has that letter at that position in other words also and make an updated dictionary. If letter is not revealed, then we will discard those words from the updated dictionary which has that letter at that position.

This way we will keep shortening our dictionary. If a node has only one word in it, then it is the leaf node and we will stop expanding the decision tree and make it a leaf node. This way at each stage, we will use our own splitting criteria and make two child nodes out of which one will act as the parent node which has the most probability for it to have that word.

If we are able to reach the leaf node within fifteen guesses, then that is counted as win and if not then we lose. In this game, instead of Melbo making random guesses, we have trained it in such a way so that it learns from the previous data after each guesses and is able to make an informed guess at each step so that the probability of it being correct increases. If probability of it being correct increases at each step, then entropy decreases and information gain increases.

We have created split dictionary for each type of occurrences of patterns and then calculated the occurrences of each pattern. If found, frequency is updated accordingly and using that frequency we have calculated entropy and information gain.

Entropy is calculated using this formula:

$$Entropy = \sum_{n=1}^{c} -p_n \log_2 p_n$$

where p_n is the probability of randomly selecting an example in class n.