This file is about functions of KNN algorithm by Touqir

(I have given a high level description of this sampling in page 5-6 in my document)

Sampler algorithm:

*We select the first class and assign its lower bound as 0 and upper bound as the class's probability

*For cases other than first and last class we assign the lower bound as the upper bound of the previous class in the list and then assign the upper bound of this class as (previous class's upper bound+this class's probability)

*In the case of the last class in the list we assign its upper bound as 1 and its lower bound as the previous class's upper bound.

*We 3 consecutive random numbers and average them and then check in which class's probability bound, that averaged number is found and then select that class and append it to a our fake sample list

*Repeat the process until your fake sample list size becomes (k-z).

(I have visually described this algorithm in page 7 and 8 in my documents)

K_nearest_neighbour_search algorithm:

*It maintains a "to_visit" list and its copy, present_k(number of k points we have as the iteration goes on), k_points(a list to contain the k_points)

*We put the query box into the prev_visit list

* We start a while loop and let it run until we our present_k becomes greater than assigned k. Inside that loop we use a for loop to go through the boxes in the prev_visit list.

* We have two cases inside the loop: First case is when you are in the first iteration and you only have the query box in the prev_visit list. It then assesses whether up, left, right, down neighbour boxes are present in the grid(in case of an adjacent boundary, the respective box wont be present). It does it by comparing its coordinates with the size of the x axis, y axis of the grid and also by seeing if the coordinates are greater than 0 or not.(the code explains this better)

Second case is just for other boxes in the prev_visit list. See my code comments for this

*In this loop if neighbor boxes are found, we append them into to_visit list

*after this loop, we copy the contents of to_visit to prev_visit and empty the to_visit list

*we create a dist list that has a list for each neighbor boxes with the first parameter as the distance between it and query box and the second parameter is its coordinate

*we send this dist list to k_search function to return those boxes that have points(not empty boxes) and contents of the returned list into k_point list.

*update our present_k and repeat our main while loop

*After loop ends we store all the k_points found from the search into KNN dictionary that has the key as the coordinate of the query box.

I have tested weighted KNN and probabilistic KNN(binomial distribution based) few times. Weighted KNN actually works pretty well but accuracy of probabilistic KNN is often poor. I know it doesn't affect my marks but just wanted to tell my findings.