**Algorithm Pseudo Code**

The main part is in the regression tree python file and there are 2 separate parts in the regression tree part.

Part 1: Build a tree

Part 2: Predict the data based on the current tree model and validate it

2.1 Regression Tree Model

Function get\_slittting\_points

Iterate through the attribute value of the parameter.

Sort the attribute value array in ascending order.

For index from 0 until array.length-1

if(array[index) not equals array[index+1]

Add median value to the array

Return the array.

Function find\_best\_split\_parallel

Best\_least\_square = infinity

For i in all possible splits

Children = split\_children(data, i,)

Least\_square = least\_square(left) + least\_square(right)

If  Least\_square < Best\_least\_square

Best\_least\_square = Least\_square

Assign best\_children

Assign best\_split\_point

Function split\_children

Left\_index = all indexes which value is <split

Right\_index = all indexes which value is >=split

Left\_label = all labels which index is contained in Left\_index

Right\_Label = all labels which index is contained in right index

Return Left\_index, Right\_index, Left\_label, Right\_label

Function create tree(data, all\_pos\_split, max\_depth, curr\_depth, ideal\_ls)

// checking two stopping conditions

All\_features = all\_pos\_split

Condition1

sum(all features) is 0 ,

Ending because there is no more features

Condition2

End when curr\_depth > max\_depth

For j in all\_pos\_split

Get min\_split which creates min\_error

Children = split\_children(min\_split)

// make the recursive functions call on create\_tree

create\_tree(children.left\_data…)

create\_tree(children.right\_data…)

Function make prediction(curr\_node, data)

// base case here

If curr\_node is leaf

Return its prediction

Get the attribute value from the data

If attribute\_value < split\_vaule

Choose left path

Else

Choose right path

2.2 Using tree model to build Lambdamart  model.

Lambdamart python file

Function dcg(scores)

Pass an array as parameter,

Iterate each entry of the array,  get the value of 2 to the power of entry value, divide by the log likelihood of number of index.

Add all values together return it.

Function dcg\_pred(scores)

Temp\_a = np.power(2, scores[i]-1/ np.log(i+2) for the top 10 values and add them up

Function dcg\_pred(scores, i, g, idcg)

// calculate the percentage we could improve by swapping 2 values

//idcg is the ideal dcg value here

Swap the values of index i and j in the array.

Old\_Dcg =dcg(scores) without swapping

New\_Dcg = dcg(scores) after applying swapping

Return (new\_dcg- old\_dcg)/idcg

Function single\_dcg(scores, i, j)

For 2 different index i and j of scores array.

Return (np.power(2, scores[i]) - 1) / np.log2(j + 2)

Function lambda\_parallel(args) which args include

Calculate the delta\_ndcg according to the passed array and i, j index

Then calculate the lambda value based on the delta\_ndcg

Function compute\_lamda(args):

Core function

Based on the passed parameter true scores, predicted scores,

First using numpy argsort to retrieve the corresponding index of the sorted array of both true value and our predicted value.

For all the i,j pairs  in good\_ij\_pairs

Calculate  single\_dcg for each pair and assign key with (i,j) and single\_dcg(i,j) as the value, store the pair in the dictionary.

For all the i,j pairs in good\_ij\_pairs

Calculate lambda value based on the difference between i,j index and z\_ndcg value of (i,i) and (i,j)

Function constructor of class LambdaMart

LambdaMart has its own training data, its own defined number of trees, leaves per tree and its learning rate.

Function  predict

Iterate through each query indexes

Use the regression  tree predict function to predict the training data of each entry

Add the predicted value together and assign it to the predicted scores which we return from the function

Function validate

Predict the testing data based on the tree build from training data

Function save and load

Save the Lambdamart model in the file by dumping it and load it every time

Main function

Read our training data from the file

Build the Lambdamart based on training data and learning rate by calling the constructor

Calculate the average dcg score by validating  the testing data.