A02G14Q14 : KNN & KFF ALGORITHM IN 2D MATRIX

IV Semester B.Tech, Department of Studies in Information Technology,

Indian Institute of Technology Allahabad, Prayagraj, India

Abstract: In this paper, we have designed an algorithm that implements KNN and KFF algorithm in a given 2D matrix forming a kernel.
We have discussed the time complexity of the algorithm by both Apriori and Apostriori analysis.

I. INTRODUCTION

The k-nearest neighbours (KNN) algorithm is a simple, easy-to-implement that can be used to solve both classification and regression problem.
The KNN algorithm assumes that similar things exist in close proximity.
In other words, similar things are near to each other. "Birds of feather flock together".
Correspondingly KFF is also same as KNN its just find farthest elements (points) in a 2d matrix.
In our algorithm we have to initialize matrix of size 100*100, with the random number's, then we have to make rectangular kernel and apply KNN and KFF to its four corners of rectangle to find nearest and farthest neighbours.

II. ALGORITHM DESCRIPTION

This algorithm displays implementation of KNN and KFF algorithm on rectangular kernel in matrix.

Stage 1: we initialize a matrix of 100*100 with random no. using Random function.

Stage2: we take out a rectangular kernel from the given matrix and it's for corner points.

Stage3: we apply KNN and KFF on four corner points and store nearest points in arrays.

- Initialise k to our chosen no. of neighbours.
- Calculate distance between other points and corner points. Using Euclidean Distance formula (dist((x, y), (a, b)) = $\sqrt{(x - a)^2 + (y - b)^2}$).
- Sort the order of distances and indices from smaller to largest by distance.
- Pick the first k entries from sorted collection.
- Get labels of selection entries.
- knn basically predicts from given data.

III. ALGORITHM AND ANALYSIS

Algorithm: power
Input k, a, b,n;
Output result: arr[k];

Method:

ALGORITHM 1: Extractkernel

Function: extract_kernel(int[][] matrix, int rowstart, int rowend, int colstart, int colend)
Start
Sizerow ← rowend - rowstart
Sizecol ← colend – colstart
for i ← 0 to n-1
begin
for j ← 0 to n-2
begin
x ← i - rowstart
y ← j - colstart
result[x][y] ← matrix[i][j]
end
end
stop

ALGORITHM: KNN
IMPLEMENTATION

Function : *implements **knn** in the kernel matrix* .
Mat1[4][2] ← {(x,y),(x,n),(n,y),(n,n)}
B ←4
for p←0 to B
begin
for i ←x to n
for j ←y to n

```
Dist[i][j] ← sqrt(pow(Mat1[p][0]-
i,2)+pow(Mat[p][1]-j,2))
end
Min_Sort(Dist[][])
Begin
dist=Dist[0][0];
for p←0 to B
begin
for i ←0 to n
for j←0 to n
begin
closest_dist = Dist[i][j]
If(dist < closest_dist && dist != 0)
Begin
Mat1[p][0]←i
Mat1[p][1]←j
a←i
b←j
end
end
dist[a][b]=1000000000
end
for i←0 to k
for j←0 to 1
print Mat1[i][j]
```

ALGORITHM: KFF IMPLEMENTATION

```
Mat1[4][2] ← {(x,y),(x,n),(n,y),(n,n)}
B ←4
for p←0 to B
begin
for i ←x to n
for j ←y to n
Dist[i][j] ← sqrt(pow(Mat1[p][0]-
i,2)+pow(Mat[p][1]-j,2))
end
Max_Sort(Dist[][])
Begin
dist=Dist[0][0];
for p←0 to B
begin
for i ←0 to n
for j←0 to n
begin
farthest_dist = Dist[i][j]
If(dist > farthest_dist && dist != 0)
Begin
Mat1[p][0]←i
Mat1[p][1]←j
a←i
b←j
end
end
dist[a][b]=0
end
for i←0 to k
for j←0 to 1
print Mat1[i][j]
```

ALGORITHM: Main

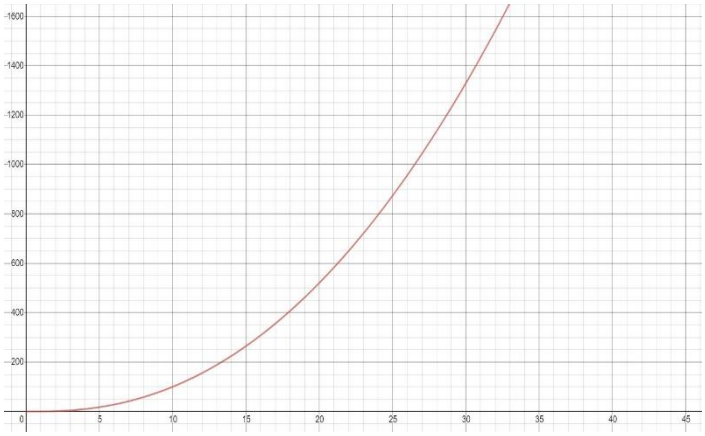
```
function GENERATE MATRIX(k)
Random dice = new Random();
for i ← 0 to n-1
for j ← 0 to n-1
a[i][j] ← dice.nextInt(2)

KNN(ExtractKernel)
KFF (ExtractKernel)

AlgEnds
```

III (a). Apriori Analysis

Table1: time complexity graph for Apriori analysis



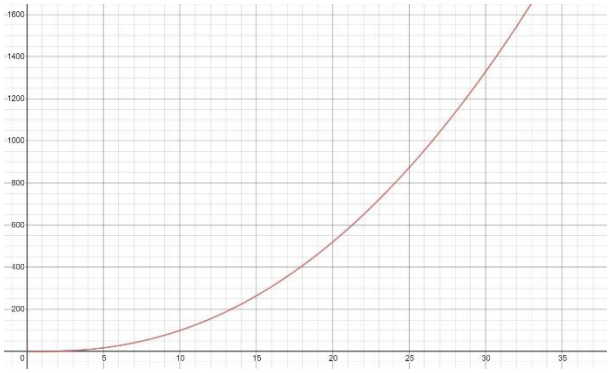
n vs time

$O(n^2Logn)$

The algorithm ***KNN and KFF take*** $O(n^2Logn)$as a worst case complexity and kff based model on different values of n.

IV. EXPERIMENTAL ANALYSIS AND PROFILING

Table2:Time complexity for Aposteriori analysis



n vs time

Figure2: Time complexity graph for aposteriori analysis.

This complexity is in between n and time.

CONCLUSION

From this paper we can conclude that the algorithm is an $O(n^2 \log n)$. knn or k nearest neighbours is a simple classification algorithm with wide application . we have implemented it in a matrix mainly can be used knn and kff algorithm in image processing and blurring image.

REFERENCES

[1] R.G Dromey, How to solve it by Computer, New Delhi: Prentice –Hall of India, 2004.

[2] https://www.projectguru.in/k-nearest-neighbor-knn-algorithm/