CSC730: Report for Assignment 3

South Dakota School of Mines and Technology

Carson Price, Kris Jensen

February 4, 2024

1 Introduction

Our task with assignment three is to describe the OptiGrid algorithm, analyze the Optogrid code available on Github [1], in great detail. After analyzing the algorithm and the code, we will modify the demo example to accept a 2D dataset of 30000 points and create visualizations of the data points and cutting planes. If time permits, we can earn extra credit by extending the modifications and visualization to a 3D dataset of 30000 points.

2 Methodology

Let's begin by setting the stage for the OptiGrid code, by analyzing the pseudocode set out by Hinneburg and Keim [2]. Then the structure of the code will be outlined, followed by a detailed analysis of each function and its purpose.

OptiGrid(dataset D, q, min_cut_score)

```
1. Determine a set of contracting projections P = $\{P_0, . . ., P_k\}$
2. Calculate all projections of the dataset $D \rightarrow P_0(D), . . ., P_k(D)$
3. Initialize a list of cutting plances $BEST\_CUTS \leftarrow \emptyset, CUT \leftarrow \emptyset$
4. FOR i=0 TO k Do
    (a) CUT $\leftarrow$ Determine best_local_cuts($P_i(D)$)
    (b) CUT_SCORE $\leftarrow$ score_best_local_cuts($P_i(D)$)
    (c) Insert all cutting planes with a score $\geq$ min_cut_score into $BEST\_CUTS$
   END FOR
5. IF $BEST\_CUT = \emptyset$ THEN RETURN $D$ as a cluster
6. Determine the $q$ cutting planes with highest score from $BEST\_CUTS$ and delete the rest
7. Construct a Multidimensional Grid $G$ defined by the cutting planes in $BEST\_CUTS$ and insert all data points $x \in D$ into $G$
8. Determine clusters, i.e. determine the highly populated grid cells in G and add them to the set of cluster C
9. REFINE(C)
10. FOREACH Cluster $C_i \in C$ DO
    OptiGrid($C_i$, q, min_cut_score)
```

2.1 Functions of class OptiGrid

1. $\_\_init\_\_$
   Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

2. fit

3. $\_iteration$

4. $\_fill\_grid$

5. $\_create\_cuts\_kde$

6. $\_find\_best\_cuts$

7. $\_find\_peaks\_distribution$

8. $\_estimate\_distribution$

9. $score\_samples$

10. $\_score\_sample$

2.2 Functions of class GridLevel

1. *__init__*

2. *add_subgrid*

3. $get_sublevel$

3 Results

4 Discussion

5 Conclusion

6 References

[1] mihailescumihai/optigrid. (2019). GitHub. https://github.com/mihailescum/Optigrid

[2] Hinneburg, A., & Keim, D. A. (1999). Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In Proceedings of the 25th International Conference on Very Large Data Bases (pp. 506-517). Morgan Kaufmann Publishers Inc.