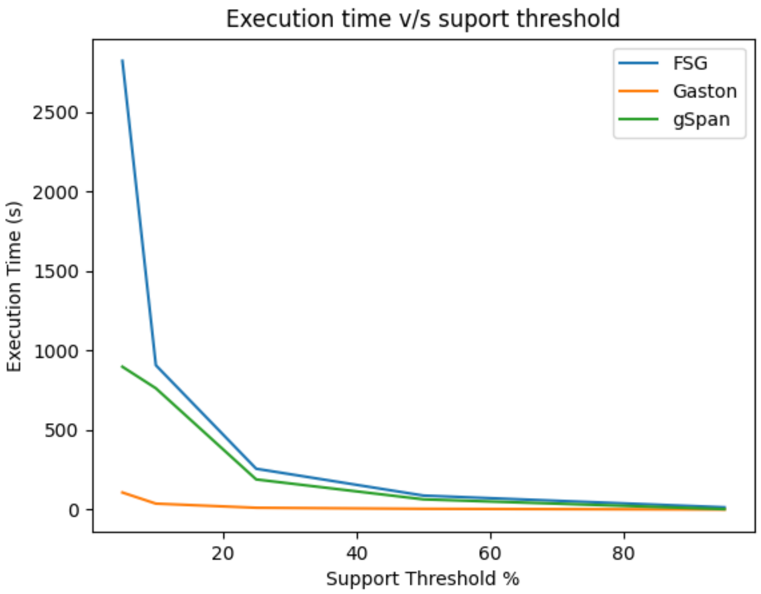
Assignment 3 Q1 Report

The following plot shows the running times of FSG, gSpan and Gaston algorithms for different support thresholds



The graph clearly shows that running time of FSG is the highest whereas running time of Gaston is lowest

Execution Times → FSG > gSpan > Gaston

The reason for best performance with Gaston is that Gaston makes use of the "quickstart principle", which is based on the fact that these classes of structures (paths, trees and graphs) are contained in each other, thus allowing for the development of structure mining algorithms that split the search into steps of increasing complexity. GrAph/Sequence/Tree extractiON (Gaston) algorithm implements this idea by searching first for frequent paths, then frequent free trees and finally cyclic graphs [1].
Gaston stores all embeddings, to generate only refinements that actually appear and to achieve fast isomorphism testing. The main insight is that there are efficient ways to enumerate paths and (non-cyclic) trees. By considering fragments that are paths or trees first, and by only proceeding to general graphs with cycles at the end, a large fraction of the work can be done efficiently. Only in that last phase, Gaston faces the NP-completeness of the subgraph isomorphism problem. Gaston defines a global order on cycle-closing edges and only generates those cycles that are "larger" than the last one. Duplicate detection is done in two phases: hashing to pre-sort and a graph isomorphism test for final duplicate detection [2].
gSpan is faster than FSG because gSpan uses DFS codes as enumerations which is much faster than the canonical labelling of adjacency matrix used in FSG, In DFS codes we can judge which DFS code is smaller even without enumerating all edges. Unlike FSG, gSpan does not require redundant candidate generation and False test. gSpan eliminates the

creation of redundant candidate generation and pruning by using Minimal DFS codes. gSpan is a DFS approach whereas FSG is a BFS approach. Also, in gSpan, at each iteration, the mining procedure is performed in a way that the whole graph dataset is shrunk to the one containing a smaller set of graphs, with each having less edges and vertices.

We can also conclude that the running time for each algorithm increases exponentially as we decrease the support threshold. This is because as we decrease the support threshold, the number of frequent item sets increases exponentially. Hence, the growth rate increases exponentially with decrease in support threshold.

References

[1] *Siegfried Nijssen and Joost N. Kok. 2004. A quickstart in frequent structure mining can make a difference. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '04). Association for Computing Machinery, New York, NY, USA, 647–652.* https://doi.org/10.1145/1014052.1014134

[2] *WÖRLEIN, Marc, Thorsten MEINL, Ingrid FISCHER, Michael PHILIPPSEN, 2005. A Quantitative Comparison of the Subgraph Miners MoFa, gSpan, FFSM, and Gaston. In: JORGE, Alípio Mário, ed., Luís TORGO, ed., Pavel BRAZDIL, ed., Rui CAMACHO, ed., João GAMA, ed.. Knowledge Discovery in Databases: PKDD 2005. Berlin, Heidelberg:Springer Berlin Heidelberg, pp. 392-403. ISBN 978-3-540-29244-9. Available under: doi: 10.1007/11564126_39*

Assignment 3 Q3

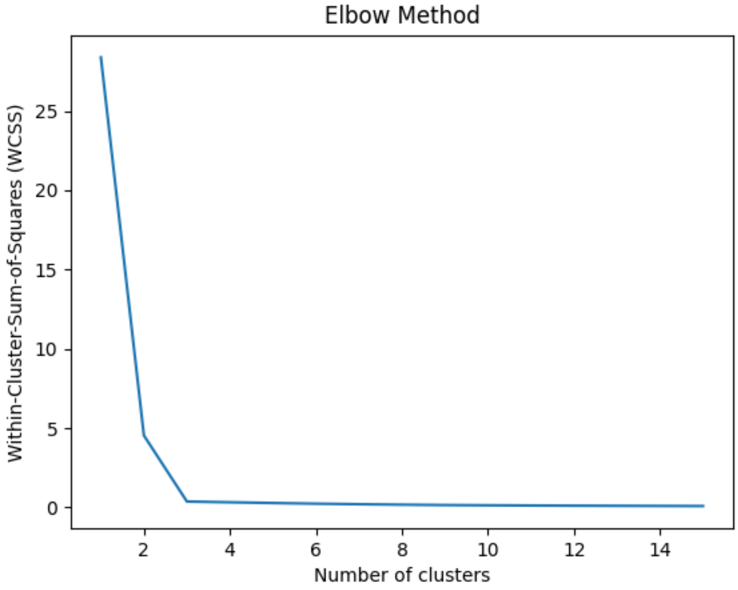


Image 1: Elbow plot for 2-d dataset

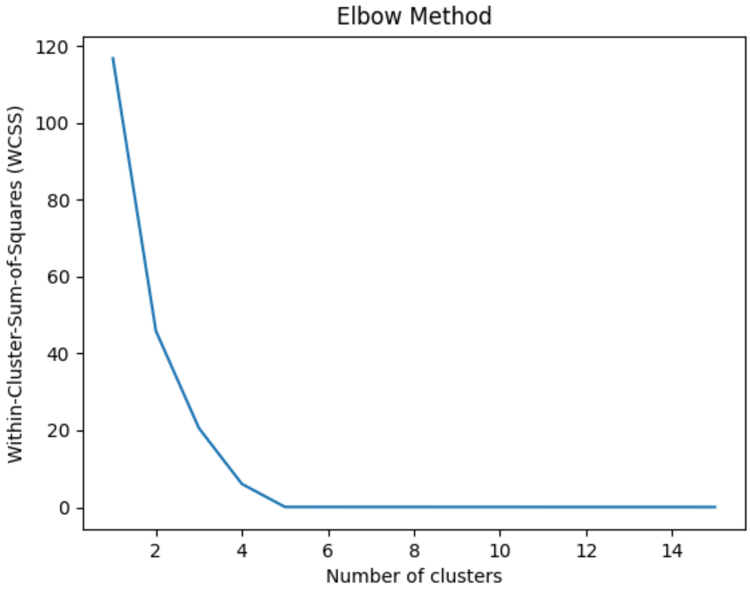


Image 2: Elbow plot for 3-d dataset

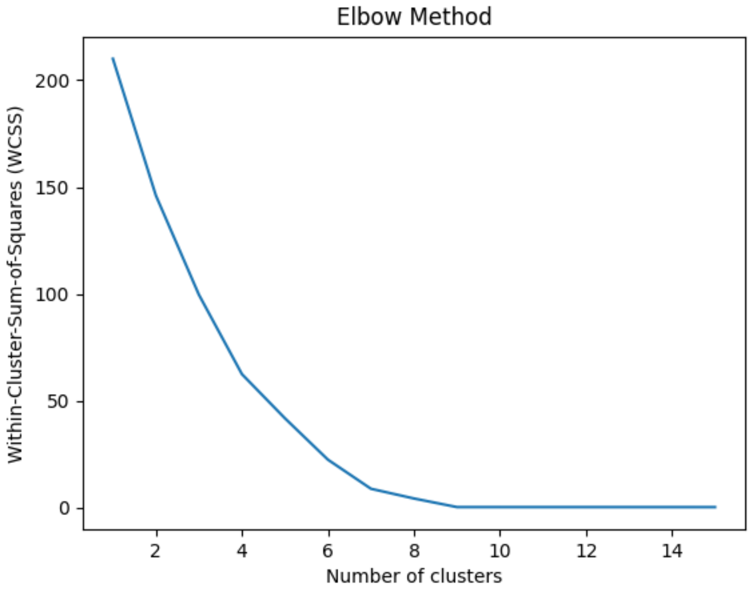


Image 3: Elbow plot for 4-d dataset

Explanation: As we can see from the images above that initially there is a steep decline in within cluster sum of square distances with the increase in number of clusters. We observe a elbow shape with a knee point at a specific value of k. Beyond the knee point as K increases we do not observe significant decrease in within cluster sum of square distances. This is because beyond the knee point the groups of datapoints are already compact (having low intra cluster distance) thus creating a cluster. This knee point specifies the number of clusters in our dataset.

For 2-D dataset, we observe knee point at K=3, indicating that there are 3 clusters.

For 3-D dataset, we observe knee point at K=5, indicating that there are 5 clusters.

For 4-D dataset, we observe knee point at K=9, indicating that there are 9 clusters.

There are multiple metrices to evaluate the quality of clusters formed such as –

1. Silhouette index
2. Dunn’s index