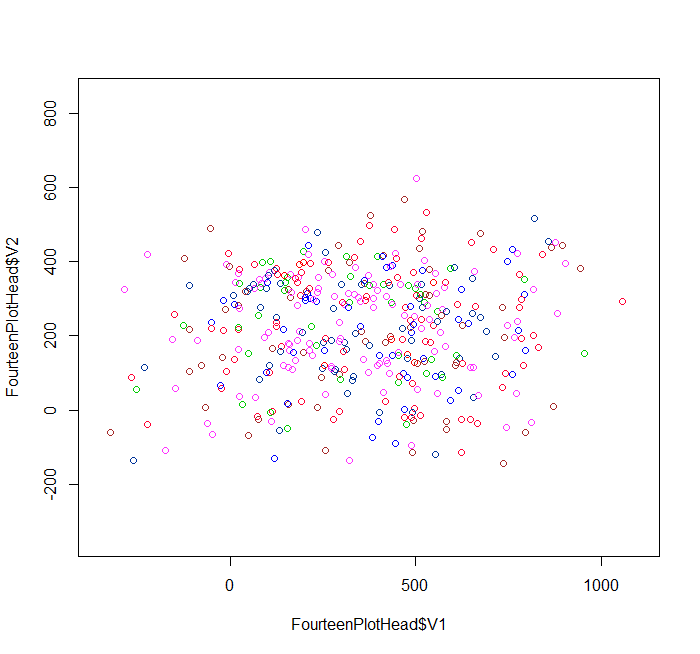
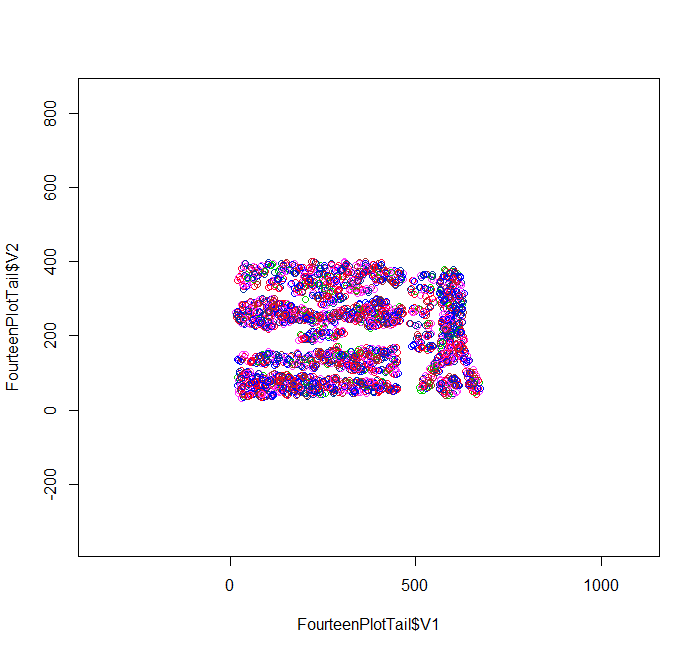
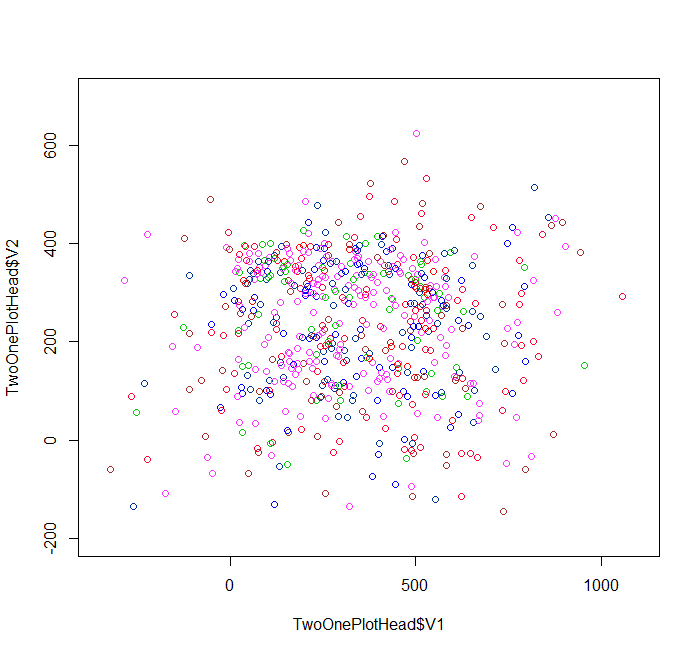
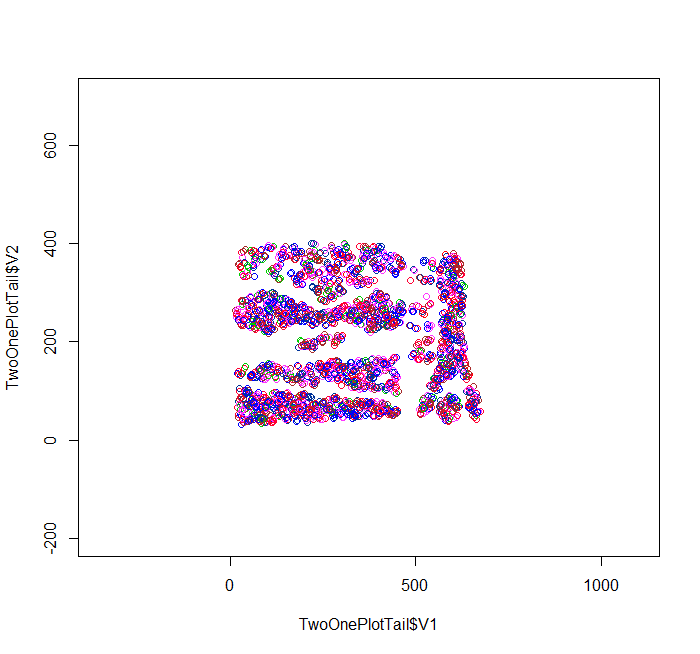
7%

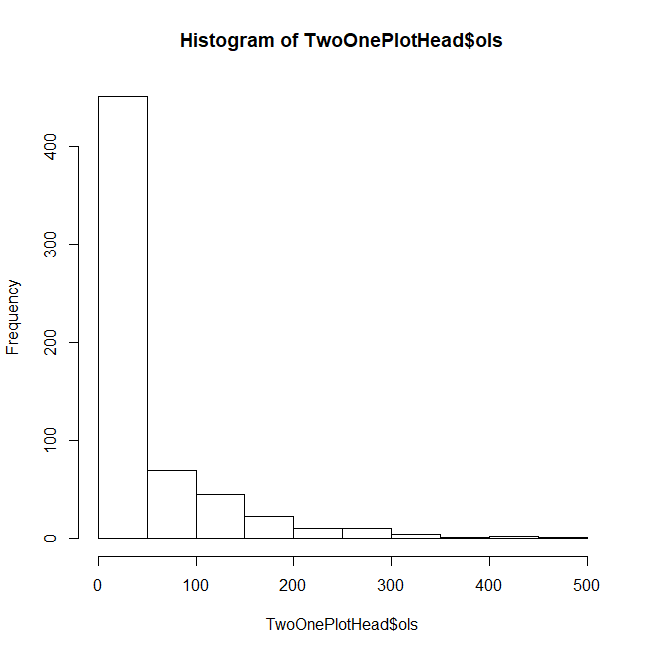
14%



21%



I think this outlier detection method works alright. At 7% the outliers that were detected seem to be distinct from the clusters that can be seen in the graphs on the right. At 14%, it is much harder to see where the clusters could fit in, but most of the points do seem to fall between the clusters. At 21%, the outliers start to eat away at the topmost cluster. At 7%, the outliers seem to be spread out, mostly along the outer edges of the graph, but there are some points in between cluster in the middle of the graph. Gaps can be seen that the clusters visualized on the right could fit into. At 14%, more outliers are detected towards the center of the graph, with much more being taken from the cluster nearest the top. Most points are in the upper 2/3rds of the graph, and spread out fairly evenly amongst the x-axis. I think all of the outliers that we can be sure of have been detected by this point. At 21%, the middle of the outliers graph is much more populated. It is harder to see where the clusters could fit in, although the space taken up by the bottom and right cluster are still largely empty. Points added in this graph seem to be between the 2 topmost clusters, and the area between the clusters at a y value of around 300.



Most values in the histogram had an outlier score of below 50. This suggests that those values are core points, while the others are the true outliers. These outlier points seem to take up about a quarter of the values in the histogram. This makes sense because a higher outlier score would mean a point is more likely to be an outlier. The majority of the outlier scores for outlier points are between 50 and 200. This shows that outliers may be more common to the outer edges of clusters than on the edges of the graph.

This selection method finds the 3 nearest points for each point in the data, and adds up the total distance to those 3 points to determine the likelihood of a point being an outlier. Points that are farther away from other points are more likely to be outliers, and thus will have a higher outlier score. This is a simple method for detecting outliers, and can be done with the base R tools. The runtime for this method is 0(n^2), which is not very efficient, and took my computer around 6 and a half minutes to compute all the distances and find the outlier score for the Complex8\_RN15 dataset. Perhaps the efficiency of this method could be increased if I were to create some predictive function to find outliers, since points near other points that are already determined to be outliers are more likely to be outliers themselves.

This method seems to give decent results in a simple, easy to understand way. It may not be very efficient compared to other algorithms, but it is competitive with some other clustering methods like DBSCAN, although it is simply determining an outlier score and not finding clusters. It is very similar to the k-nearest neighbor classification method, in that it checks the k-nearest neighbors, but it only uses the neighbors for their distance from the current point, and not to find the average of their classes.