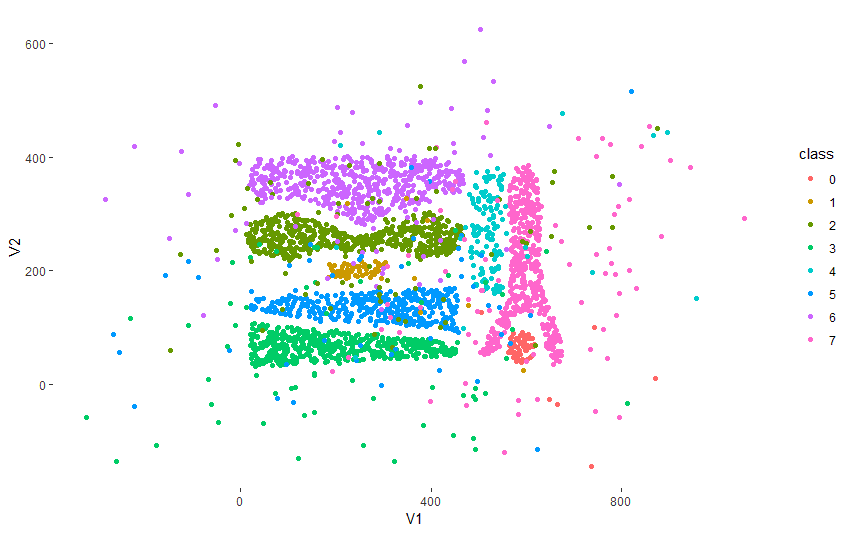
Students Name: Chansik Moon, Duong Nguyen

Instructor: Dr. Christoph Eick

***PROGRAMMING ASSIGNMENT 4***

*TASK 1:* Visualize the data set:



*TASK 2:*

**Chansik chose to use distance-based approach**

The technique that Chansik used was a distance-based approach to detect outliers (to be explained further in Task 4). Overall, the technique was implemented with the help of KNN algorithm.

**Duong chose to use density-based method:**

The task was done by using kde2d function from MASS library (to be explained further in Task 4).

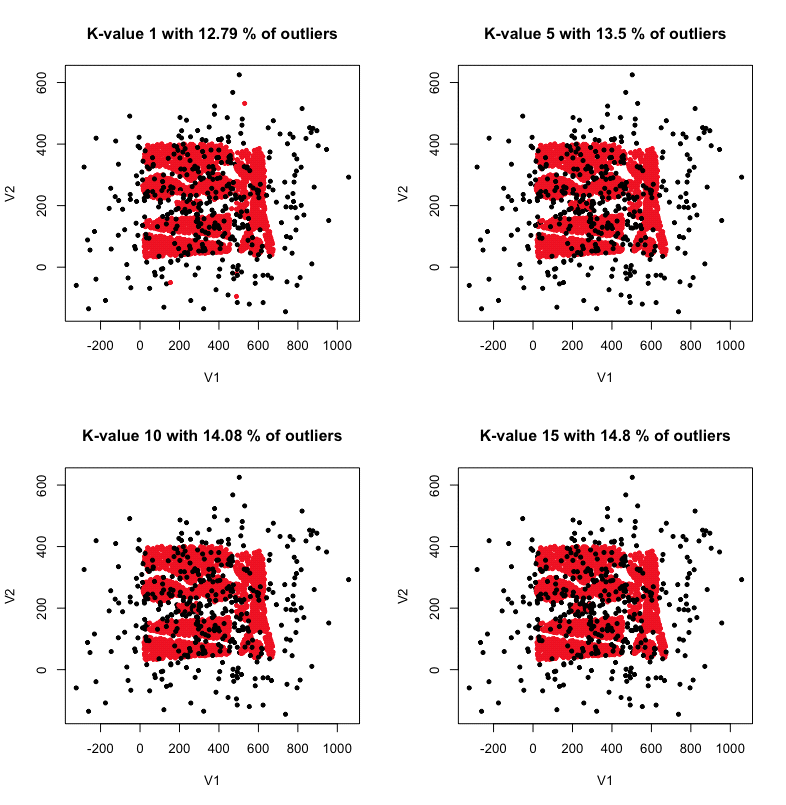
The value returned by kde2d function will be used to calculate ols.

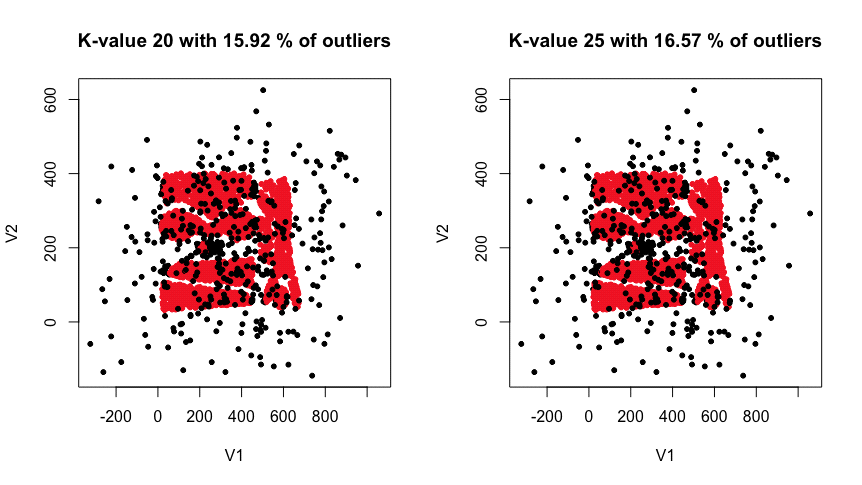
The code is included in the .R file.

***TASK3:***

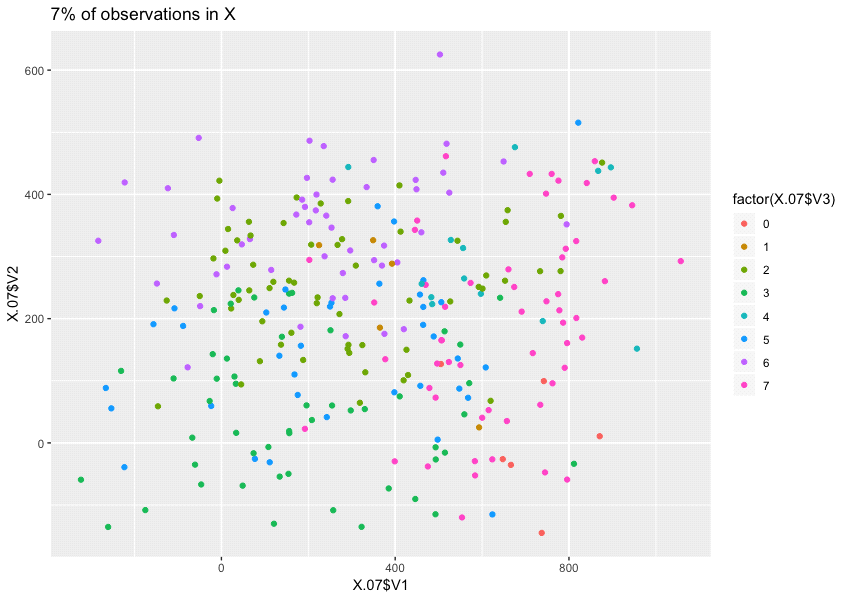
***Part a, b: the code is included in the .R file***

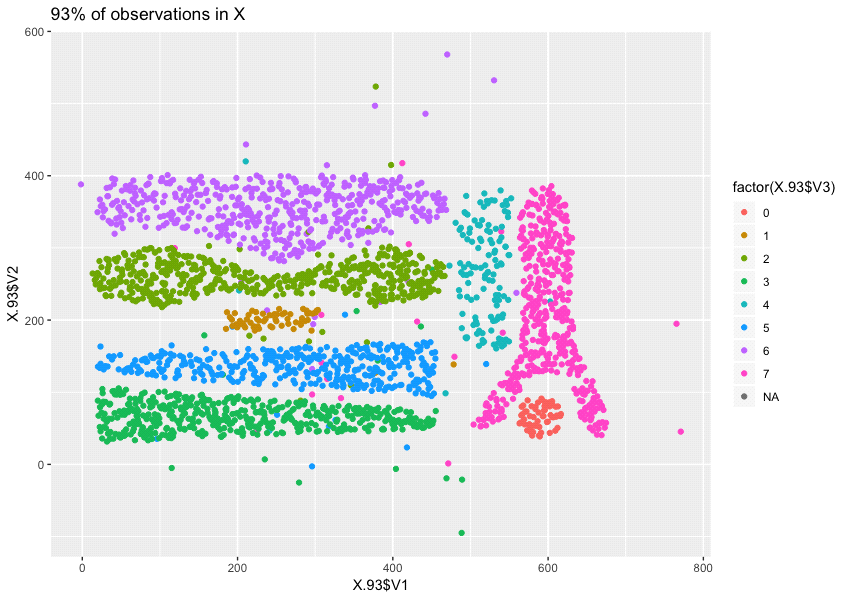
**Results from Chan’s technique:**



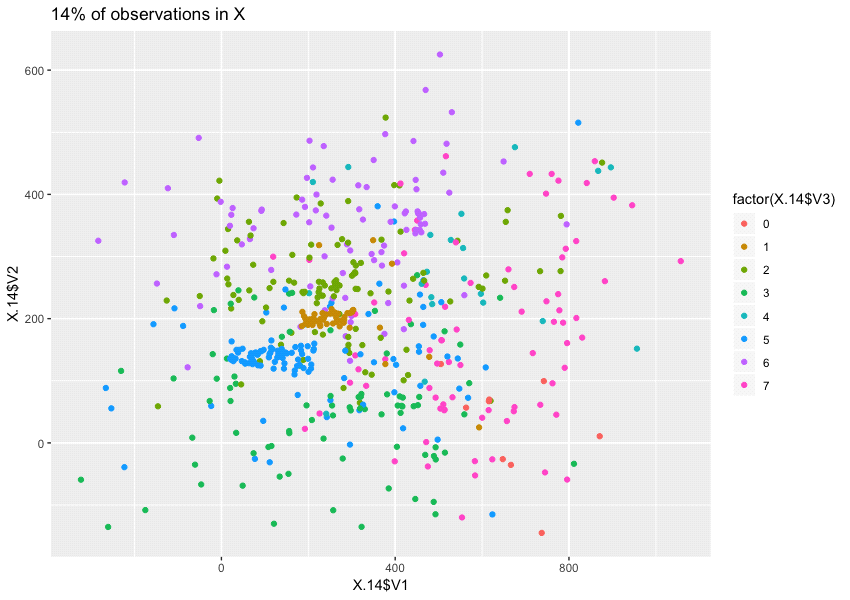


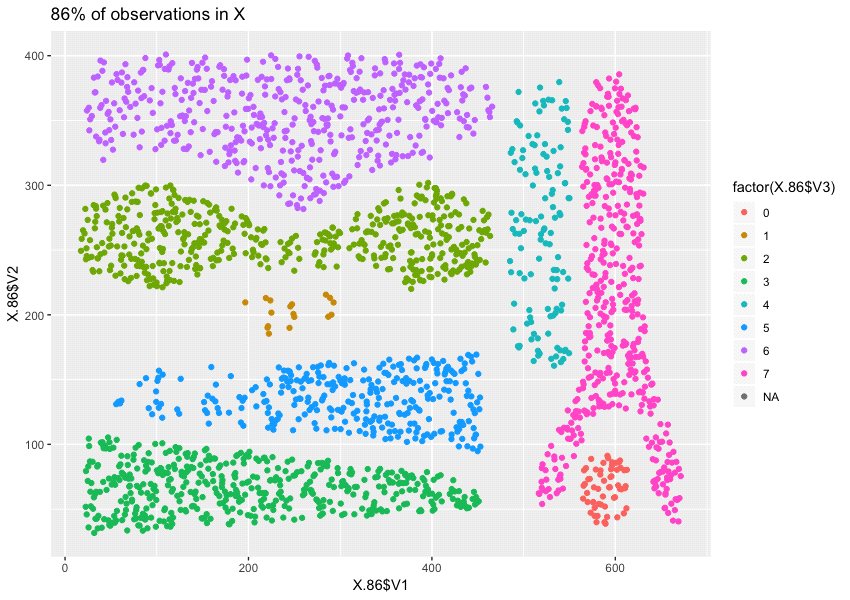
Part c:



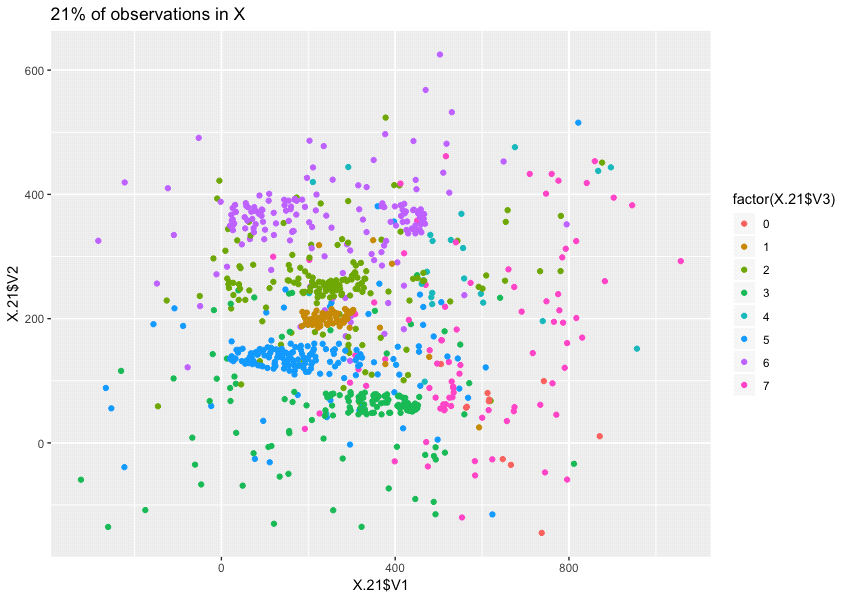


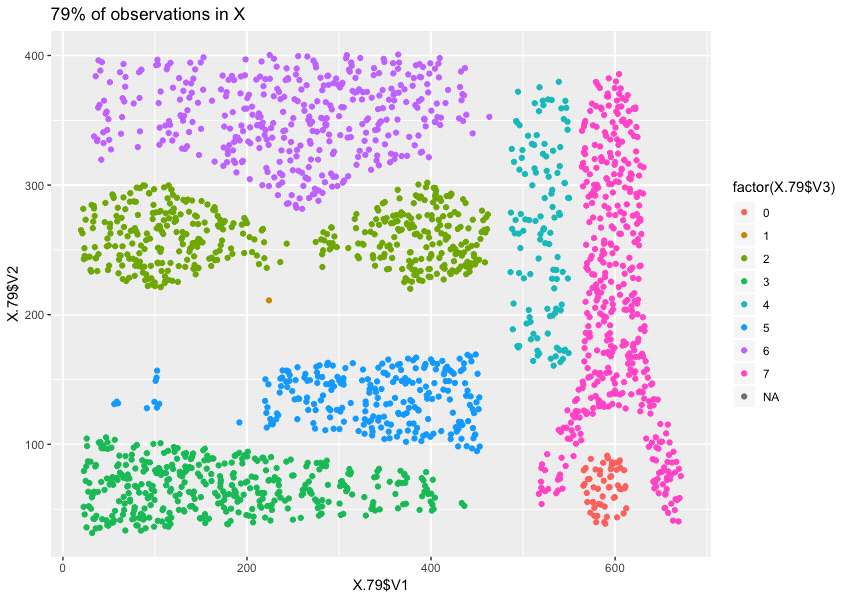
Part d:





Part e:





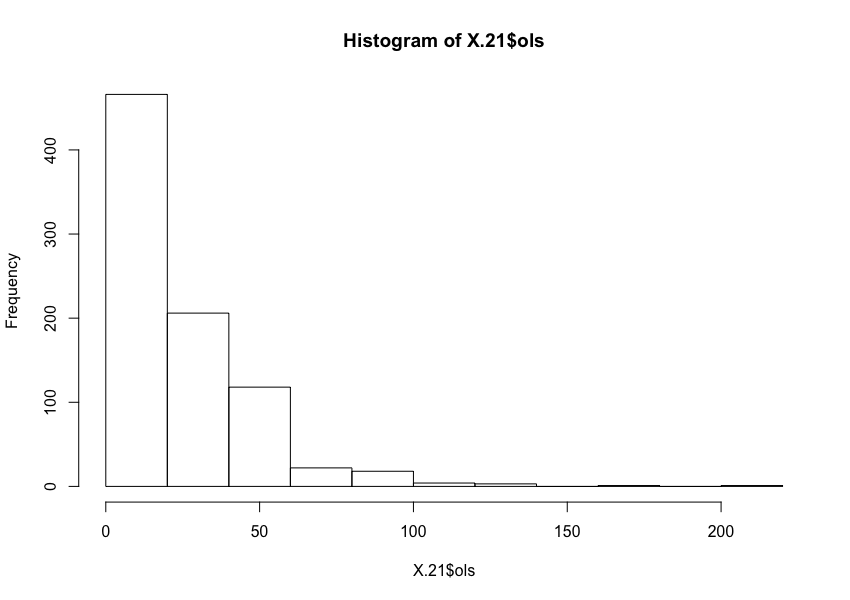
Part f:

The 7% of outliers detected by the *kNNOutlierDetection* function are primarily data points surrounding the 8 original clusters. Majority of points were located far away from the clusters. However, in the normal observations, a few outliers are still located far away from the clusters but did not detected as outliers. Some of them are not even closer to the perimeters of the clusters but still visible outliers. Despite some outliers are still visible, the *kNNOutlierDetection* function did a good work of characterizing the first 7% of outliers from the data points.

The 14% of outliers detected by the *kNNOutlierDetection* function are data points that either a part of the clusters or outliers. The data points that were classified as outliers are all gone. The normal observations contain the only its cluster. However, a few clusters got rid of the majority points. Overall, the *kNNOutlierDetection* function did a decent work of removing the 14% of noise and outliers from the data points.

The 21% of outliers detected by the *kNNOutlierDetection* function removed an essential part of a few clusters, especially cluster 1 and 5. Cluster 1 is almost removed and the 1/3 of cluster 5 is removed. Some data points were taken from the original clusters too. The normal observations show a cleaner representation of the clusters but the clusters are sparse and are missing lots of significant data points. Overall, the *kNNOutlierDetection* function still works fine to get rid of noise and outliers but it seems Complex8\_N15 does not contain 21% of outliers.

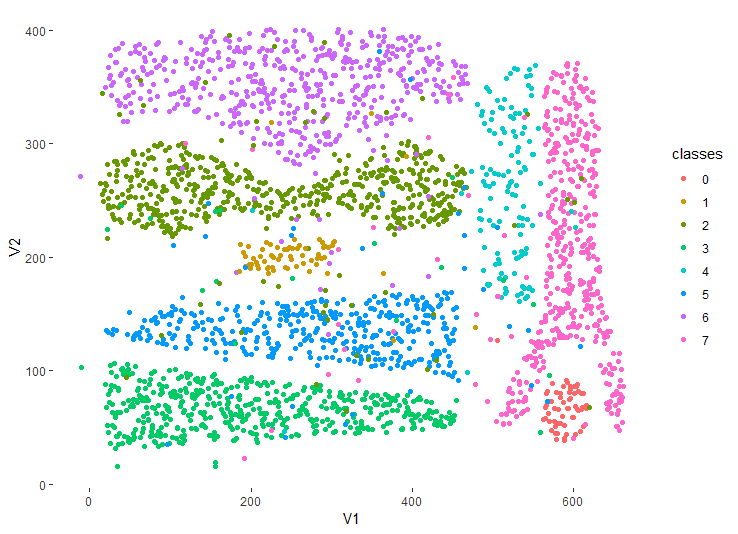
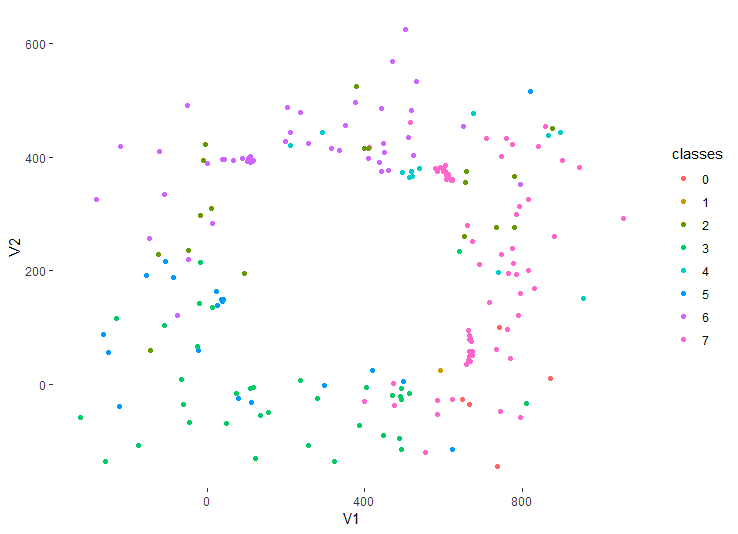
Part g:

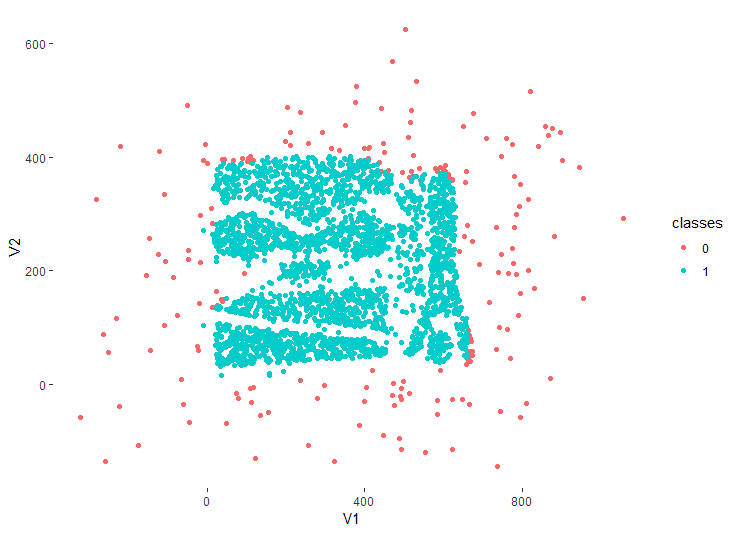


This histogram of the outlier scores from dataset where the first 21% of data points as outliers, the distribution of this histogram is predominantly unimodal with a large peak of more than 400 outliers score between 0 to 20. The remainder data points that have outlier scores greater than 20 have smaller frequencies.

**Results from Duong’s technique:**

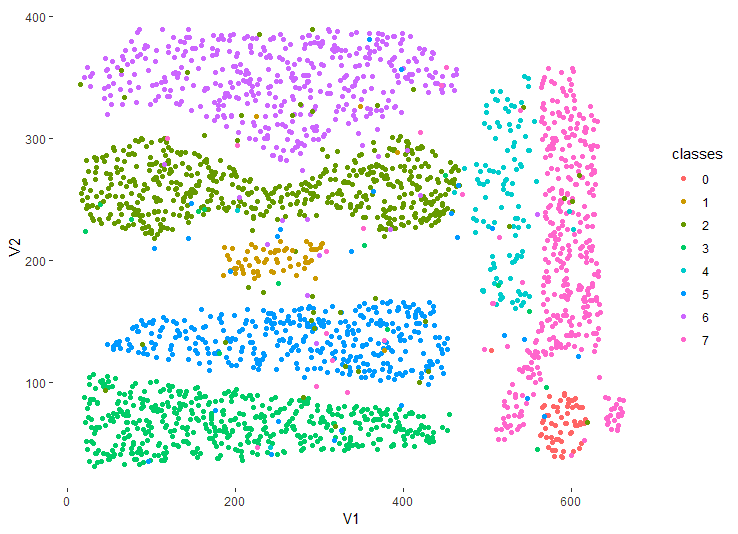
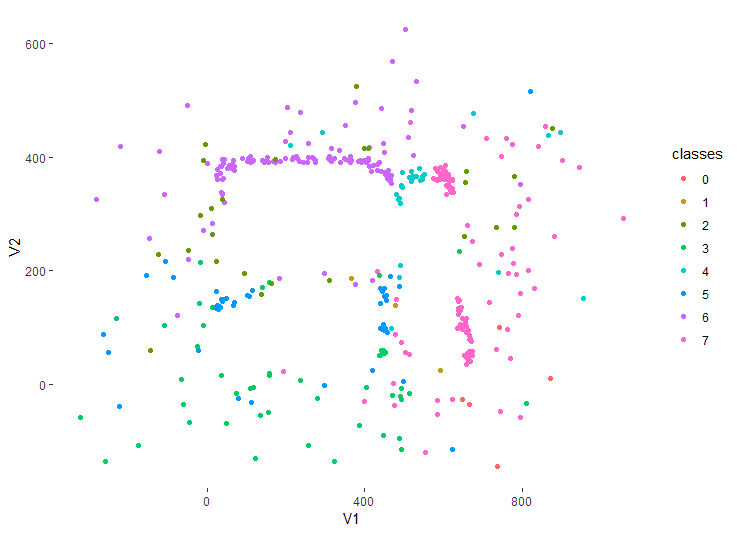
Part c:

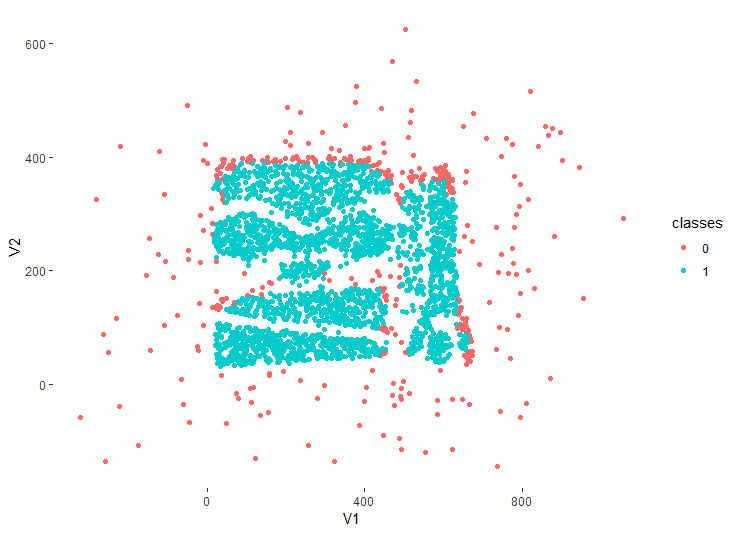




First left display visualizing 7% outliers, right displaying 93% ‘non-outlier’ and the last one display them all together with class 0 represents outlier, class 1 represents non-outliers. This was done for a somewhat better visualization of where the outliers are on the whole plot.

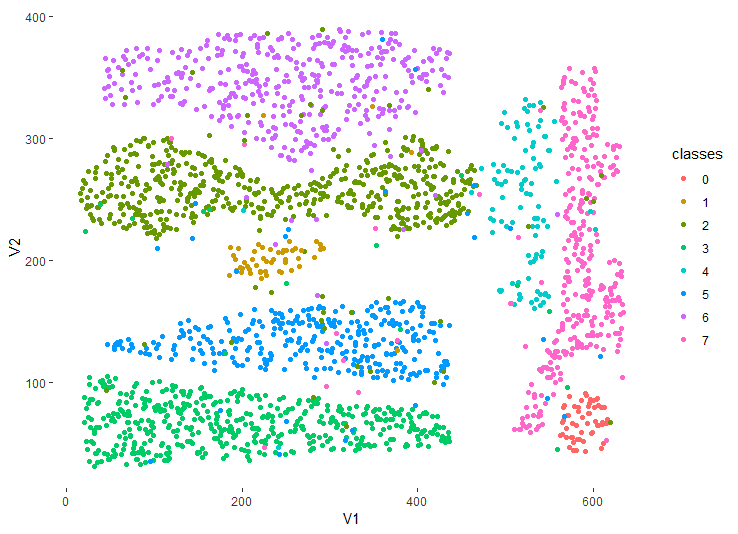
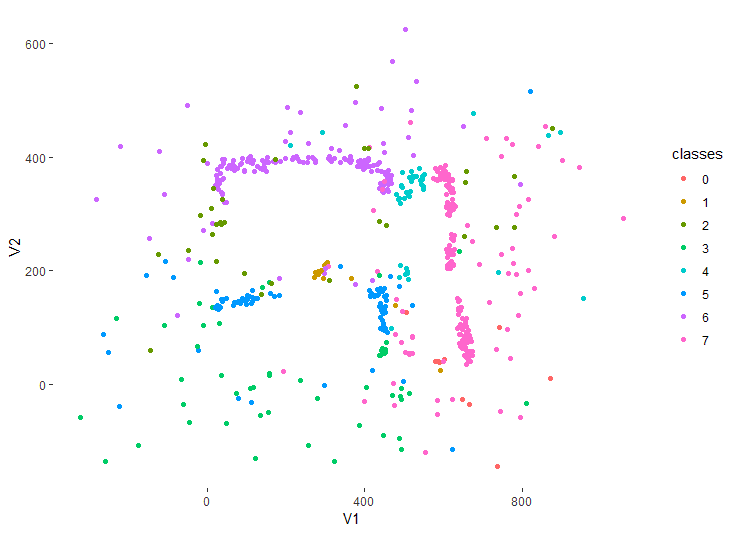
Part d:

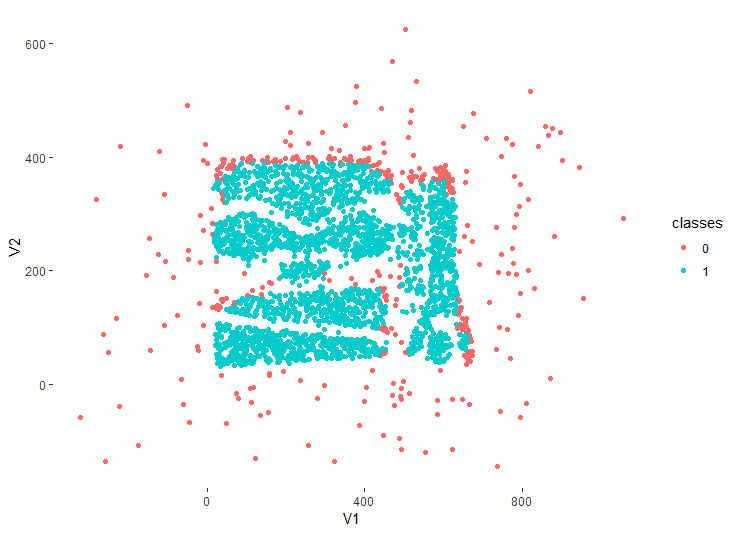




First left display visualizing 14% outliers, right displaying 86% ‘non-outlier’ and the last one display them all together with class 0 represents outlier, class 1 represents non-outliers. This was done for a somewhat better visualization of where the outliers are on the whole plot.

Part e:





First left display visualizing 21% outliers, right displaying 79% ‘non-outlier’ and the last one display them all together with class 0 represents outlier, class 1 represents non-outliers. This was done for a somewhat better visualization of where the outliers are on the whole plot.

Part f: Interpretations

Briefly look at the plots of outliers, we see that the density-based technique had done a pretty good job since we would expect to see outliers are the points that are far away from the 8 natural clusters. It looks like the ones with 7% outliers picked out most correct outliers. The more percentage of outliers we chose, the model start picking out non-liers. The model can only be effective if all outliers are sparse and far from clusters, it has no way to know if some outliers are inside the dense area, as we see from the plots of 14% and 21% outliers.

Part g:

***A screenshot of a cell phone

Description automatically generated***

Min. 1st Qu. Median Mean 3rd Qu. Max.

332993 364467 428372 2134411 735262 23844492

This is a statistics I also got from ols of 21% outliers. As we see from statistics as well as the histogram, the distribution of outlier scores is unimodal but it is very skewed, there are quite a few significant datapoints that have a very large ols (the maximum ols is almost 100 times larger than the minimum ols).

***Task 4:***

Chan’s KNN technique:

The technique that Chansik used was a distance-based approach to detect outliers. The function *kdist* computes the distance matrix of the data set and then it calculates the distance to the Kth nearest neighbor for each object. Using this function output, *kNNOutlierDetection* function calculates the outlier scores. If the K-distances are greater than or equal to the average of the K-distances of the data set then it is an outlier.

The *kNNOutlierDetection* function uses *dist()* function that is already built-in in R to compute the distance matrix of the data frame. There are two different functions, kdist and *kNNOutlierDetection*. The *Kdist* function takes two parameters: the data frame and a k-value, and it returns a vector of k-distances. The *kNNOutlierDetection* function also takes two parameters: the data frame and the k-value, and this function returns the vector of outlier scores, “ols” in addition attribute and use it to visualizing the dataset. The *kdist* function computed the distance matrix using the *dist()* function and it computes the kth Nearest Neighbor distance for all object in the data set.

The kNNOutlierDetection function calls the kdist function to initialize the outlier scores and the average of the k-distance is used as a threshold to distinguish outliers from non-outliers. If the object has an outlier score higher than or equal to the threshold, the object will consider as an outlier and is set to black color. Normal observations set to another color and the last part of the function plot the dataset showing outliers in a different color than the normal observations. This is how I visualize the true data points and the outliers to determine which k-value give the best outlier detection base on the visual integrity of the clusters.

Duong’s 2D Density-based model (KDE2D)

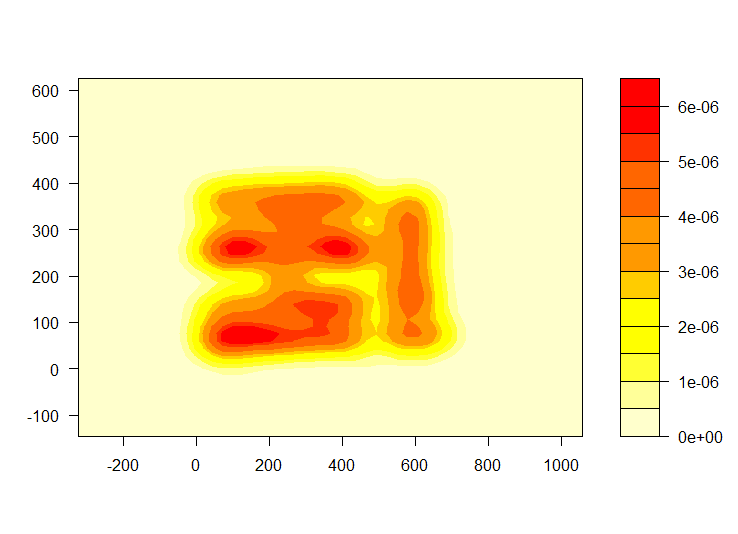
Function used: kde2d from MASS library, arguments passed the V1 attributes, V2 attributes and set n (number of grid points for each dimension) = 50. I had tried different n’s and the difference in n’s would not affect the outcome too much.

The function kde2d output 3 vectors:

* x, y: x and y coordinates of data (or the grid points)
* z: an n by n matrix of the estimated density: rows correspond to the value of x, columns to the value of y.

Based on the output of the function, I use x, y and z to estimate density of each point based on the where it would most likely land on the x-y plane, and it would be my outlier-score estimations. Because the estimations are density the larger the denser, the more likely it is to be a non-outlier) so I assign ols as 1/estimations so that the bigger the number, the more likely it is to be an outlier.

Prior to starting to visualize the outliers, I generated a 2D density heatmap, and as we will see below, the density function is promising since it shows the region where most of the points are is very dense, and further from it, it is not so dense any more (as we would expect).



***Task 5:*** By looking at the plots of outliers from Task 3, it looks like density-based model did a better job in this case. It was very successful at picking out 7% outliers since it picked out all points that are afar from clusters while KNN-method still left out the points that are far from clusters. Next for 14% outliers, density-based model started to pick out non-outliers, however, not as many as the KNN-method. KDE method failed to recognize outliers that are mixed into other clusters, this is reasonable because the model works based on density only. Nonetheless, while KDE falsely chose a part of cluster 7 (pink one) to be outliers, KNN-method did not do so, and it successfully cleared out all points that are supposedly outliers (points that have either very large or very small coordinates). Lastly, for 21% outliers (which obviously even the” best” model would falsely pick out non-outliers since the dataset was only added on 15% outliers), KNN-based method “wiped out” cluster 1(brown cluster) and a big part of cluster 5 (blue cluster). For KDE2D model, it took out a part of cluster 7(pink cluster), however, it preserves a pretty good shape of other clusters.

We did enjoy working and doing our project together. We have learnt that outlier detection is a good skill to have. Sometimes outliers are important indicators and people do unintentionally ignore them. Having some knowledge of how to detect outlier would benefit us later if we get into Data Science field or any field that requires some technique to work with data.