AC53013 GROW Assignment Report

Introduction:

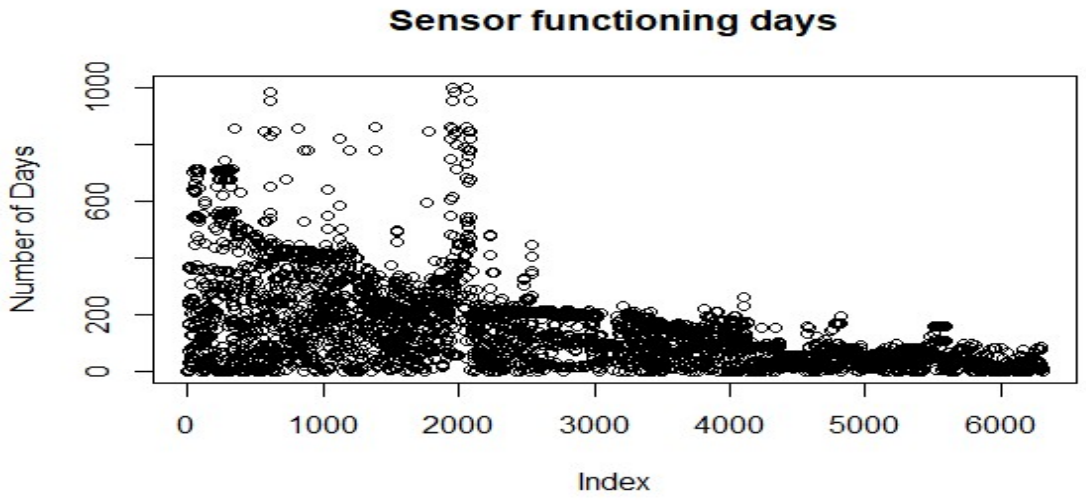
GROW Observatory is one of the first continental scale Citizens' observatory which works on improving soil, land use, climate change and overall sustainability.

GROW has, for the first time used crowdsourced ground data from low-cost sensors to validate the soil moisture information from satellites, including Sentinel 1, a new generation of high-resolution satellites. There are GROW communities, which are in 13 European countries, using the ground-based soil sensors and generating millions of soil data.

Grow Data Outline:

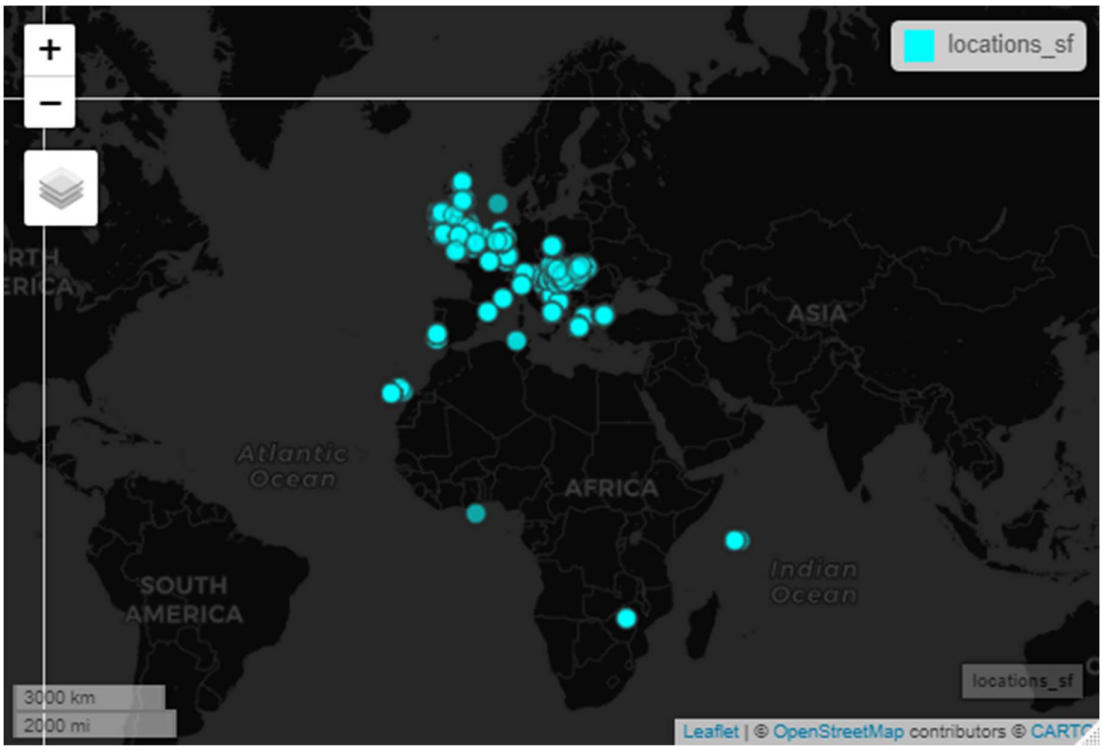
The file "GrowLocations.csv", had in total 39252 rows of data, with 7 columns namely Serial, Latitude, Longitude, Type, SensorType, Code, BeginTime and EndTime. All sensors were of same SensorType, "Flower Power". Each sensor also recorded 6 types of data, which are AirTemperature, BatteryLevel, Fertilizer level, Light, SoilMoisture and WaterTankLevel.

Figure below shows for how long the sensors sent the data, by calculating the days between Start and End date. The sensor averaged about 142 days of data recording, which can be seen below:

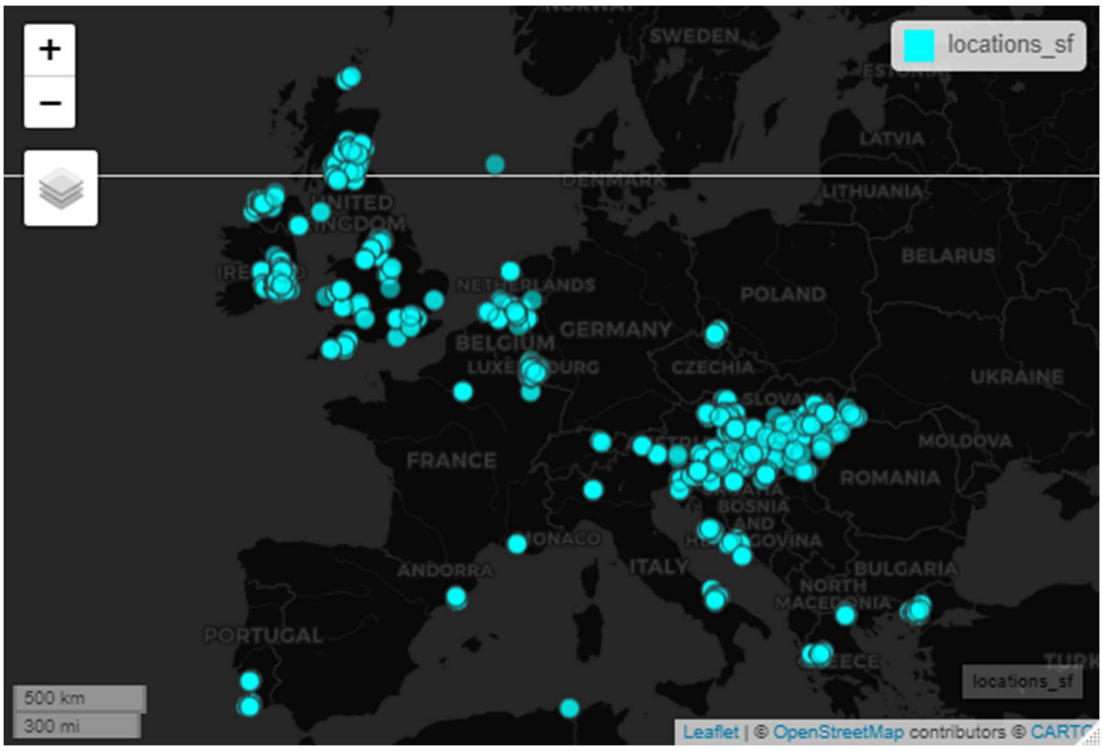


GROW Location plot:

After getting a proper set of data, the spatial data was plotted using RStudio:

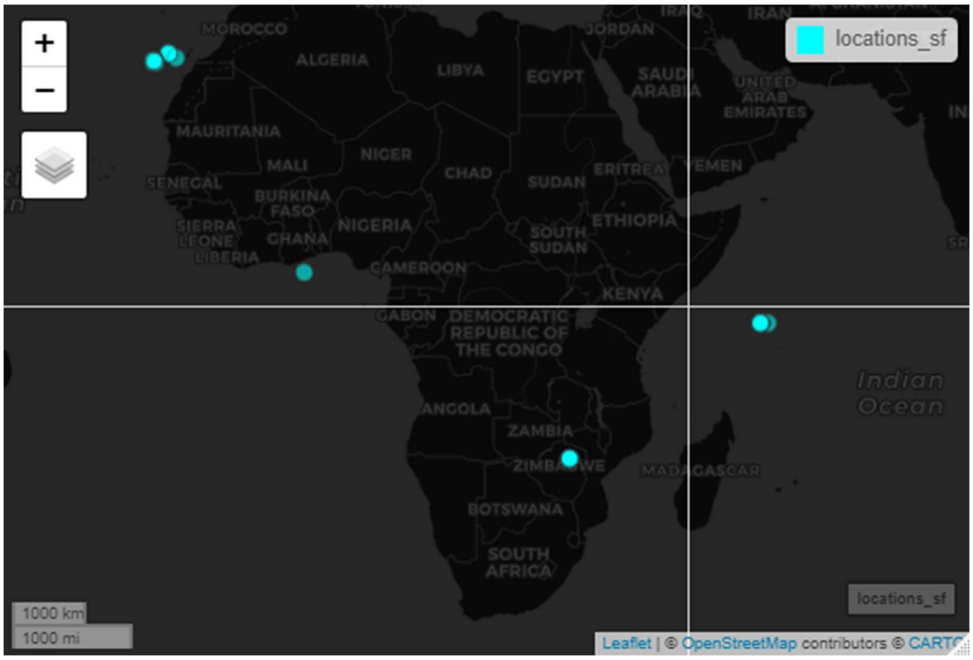


To get a better picture of the data, we can zoom in and look at the locations where is been placed in the European region, as seen below:



Countries like UK, Ireland, Hungary and Luxembourg can be seen as having the greatest number of sensors placed between them. Other sensors can be seen in parts of countries like Netherlands, Belgium, Austria, France, Italy, Portugal Croatia, and Greece.

There were some sensors which were outside the EU regions, shown below:



Few of the sensors were tracked in a small island near Morocco, Zimbabwe, and few more in Atlantic and Indian Ocean.

The data had multiple duplicate values before cluster analysis, so some data cleanup was performed. While going through the data, some oddities were found, including:

- Locations having both Latitude and Longitude values as 0.
- Latitude and Longitude values in the range of thousands.
- A sensor with an incorrect date recorded as well, with year showing as 0001.
- Some rows had empty Serial number values, but other data were present in it.

These values were removed and a new file with the required dataset was saved in another file. The plotting and cluster analysis was done after getting the new dataset.

Data Clustering:

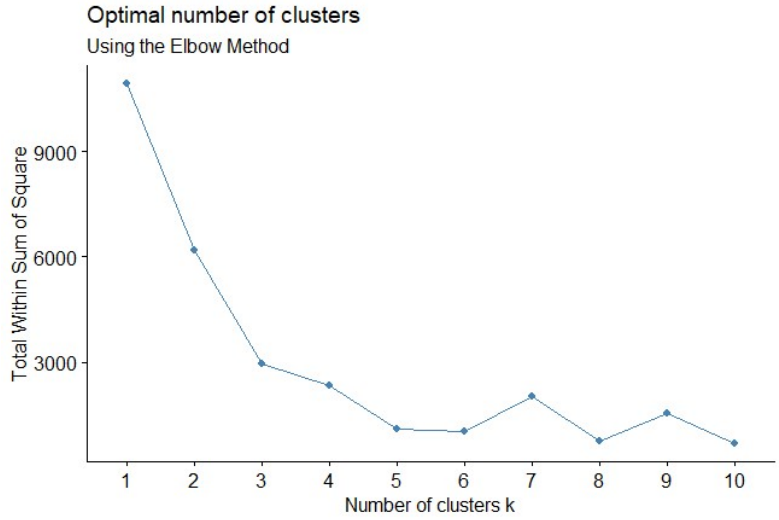
Data clustering was done using two methods (Each analysis was done using R Programming in RStudio):

- **K-Means Clustering -** K-means is a partitioning method (non-hierarchical method) that divides observations in data into k mutually exclusive clusters (Lletí et al., 2004). In this way, it treats each observation in data as an object having a location in space and finds a partition in which objects within each cluster are as close to each other as possible, and as far from objects in other clusters as possible.

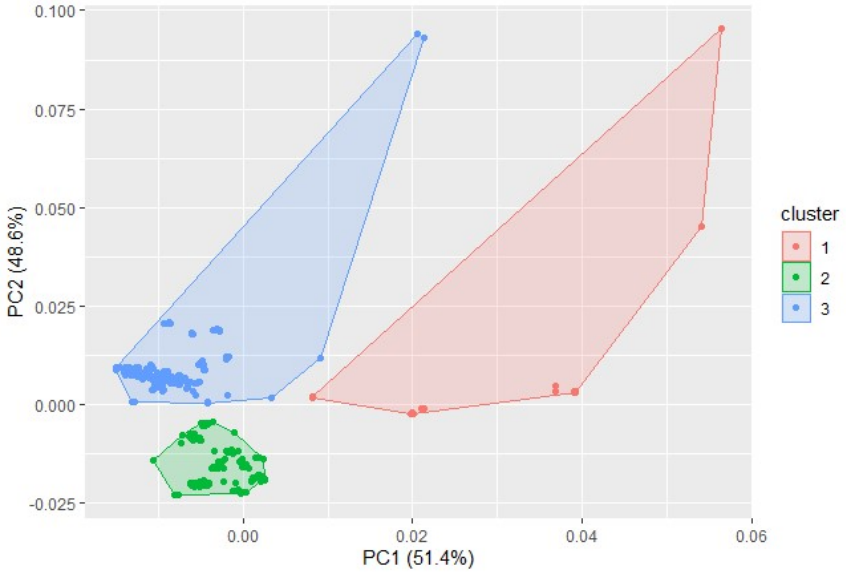
  Each cluster in the partition is defined by its member objects and by its centroid, or Centre. The centroid for each cluster is the point to which the sum of distances from all objects in that cluster is minimized. It is important to highlight that k-means is an iterative algorithm that minimizes the sum of distances from each object to its cluster centroid, over all clusters by moving objects between clusters until the sum cannot be decreased further. The result is thus a set of clusters that are as compact and well separated as possible (Lletí et al., 2004).

  The oldest method for determining the true number of clusters in a data set is inelegantly called the elbow method (Kodinariya & Makwana, 2013). It is a visual method. The idea is that Start with K=2, and keep increasing it in each step by 1, calculating your clusters and the cost that comes with the training. At some value for K the cost drops dramatically, and after that it reaches a plateau when you increase it further. This is the K value you want.

  Using the Elbow method, we tried to determine the K value which was optimum for our data, and it can be seen below:



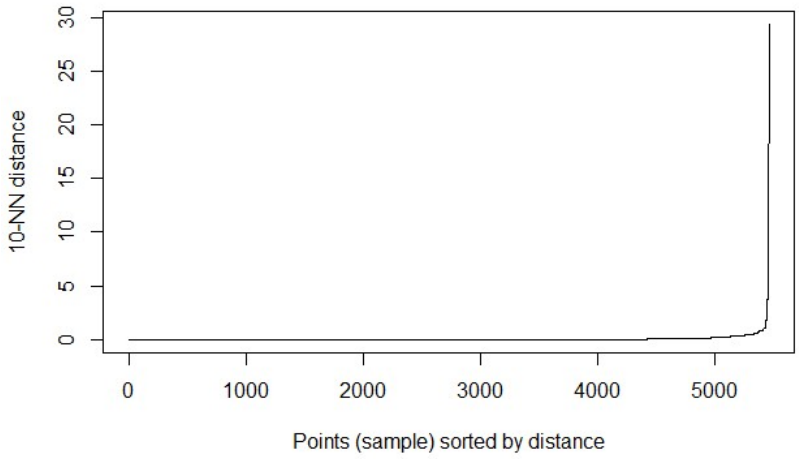
Using value from the data, the K-Means clustering was performed. The parameters for Kmeans function in R include the dataset, value of k (which was plotted above) and nStart value (Running the algorithm for multiple times). Value for k was set as 3 and nstart as 100, for reaching convergent result. The clusters can be seen in the plot below:



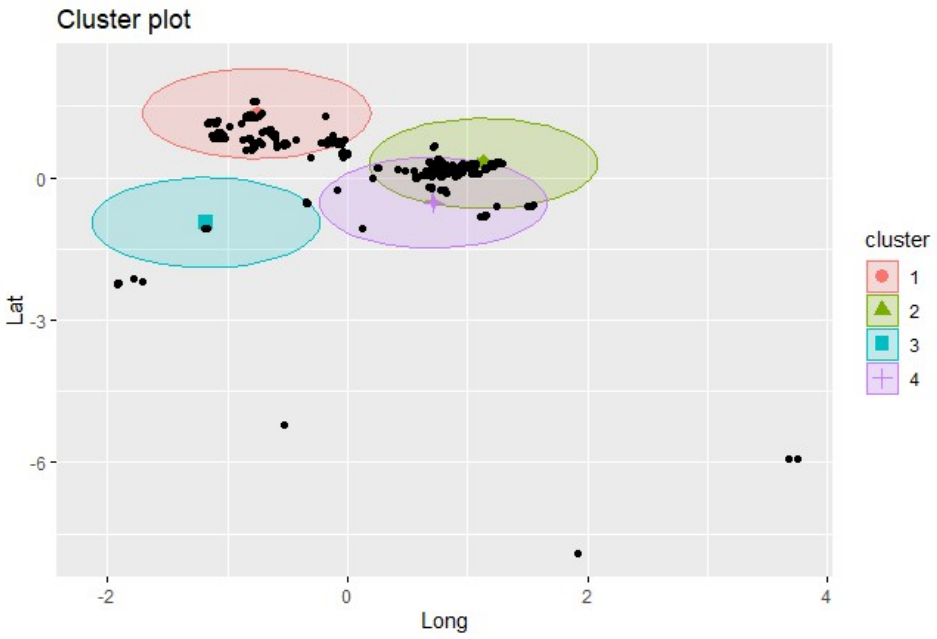
- **DBSCAN Clustering** – DBSCAN Stands for Density-Based Spatial Clustering of Applications with Noise. It is the first density based clustering algorithm (Khan et al., n.d.). It was first proposed during 1996, designed to cluster data of arbitrary shapes in the presence of noise in spatial and non-spatial high dimensional databases. For each object of a cluster, what DBSCAN does is to check whether the neighborhood of a given radius (Eps) must contain at least a minimum number of objects (MinPts), which means that the cardinality of the neighborhood has to exceed some threshold.

  DBSCAN has the ability in discovering clusters with arbitrary shape such as linear, concave, oval, etc., and comparing to some clustering algorithms, it does not have the requirement of the number of clusters. It has proven its ability of processing very large databases as well (Birant et al., n.d.).
  This algorithm is based on the idea that objects which form a dense region should be grouped together into one cluster, by using a fixed threshold value to determine denser regions. They search for regions of high density in a feature space that are separated by regions of lower density.

Using Function kNNdistplot in R, we first determine the eps value. This function takes the data and the value of k, which is number of nearest neighbors used for the distance calculation. Plot of this function run can be seen below:



After getting the eps value using kNNdistplot, we can use it in the dbscan function parameter, which also has a minPoints parameter, which is the minimum number of neighbors within "eps" radius. After checking with different minPoints to get the correct clusters, it was selected as 375. And the cluster result is shown below:



Conclusion:

The GROW observatory data, after some data cleaning, was used to plot the locations of the sensors and then for cluster analysis using K-Means and DBSCAN clustering.

In the process of getting the correct clusters, in K-Means clustering it does detect two clusters correctly, but it also includes the outlier data in it, as shown in the plot. It can be known which data are the outliers but still nothing much about it can be predicted here. But in DBSCAN, unlike K-Means, you can see the outliers in the data, and it the clustered data as well. These are the points which was plotted outside the EU regions, and it can be see away from a cluster here as well. So, it can be said that DBSCAN does a better job in spatial data clustering as compared to K-Means.

References

- Cluster in K-Means Clustering. International Journal of Advance Research in Computer Science and Management Studies, 1(6). www.ijarcsms.com

- Lletí, R., Ortiz, M. C., Sarabia, L. A., & Sánchez, M. S. (2004). Selecting variables for k-means cluster analysis by using a genetic algorithm that optimises the silhouettes. Analytica Chimica Acta, 515(1), 87–100. https://doi.org/10.1016/j.aca.2003.12.020

- Khan, K., Rehman, S., Aziz, K., … S. F.-T. fifth international, & 2014, undefined. (n.d.). DBSCAN: Past, present and future. Ieeexplore.Ieee.Org. Retrieved May 27, 2021, from https://ieeexplore.ieee.org/abstract/document/6814687/?casa_token=2VSFqcmdUlwAAAAA:x5cGOg6sVJfVUkkoOXo-lM2RX56LY7c31q1GYqHfDRnn3arawYsdhjfN2cc1Ej-vgg3NkJR-wiMP0w

- Birant, D., engineering, A. K.-D. & knowledge, & 2007, undefined. (n.d.). ST-DBSCAN: An algorithm for clustering spatial–temporal data. Elsevier. Retrieved May 27, 2021, from https://www.sciencedirect.com/science/article/pii/S0169023X06000218?casa_token=09CKpKDqtMsAAAAA:YBYXoizulk6Aj5nlmoOwaS99kH05AYNnyMa6siXTPrT3zAnbTbuTPhixLvM76PhWLIkiA_k6dPg