CSC591/791: Spatial and Temporal Data Mining
Homework 2. Due: 2/17/15 @ 11.55pm EST.

Student Name: Abhishek Kumar Agrawal
Student ID: 200061445

Notes:

- Most questions are open-ended, so please chose the best solution based on your reading/understanding.
- Some questions require reading papers and web resources. Likewise you can discuss with fellow students. Whatever the case, answers should always be your own.
- You should always **cite** the source of all materials if they are used in your answer/solution/program.
- Answers should be concise and to the point (and within the space provided).
- Word/PDF are the only accepted formats.

#Q	Max Points	Your Score
1	35	
2	30	
3	35	

Q1. For given satellite image (ilk-3b-1024.tif; 3-dimensional, 1024x1024), do the following:

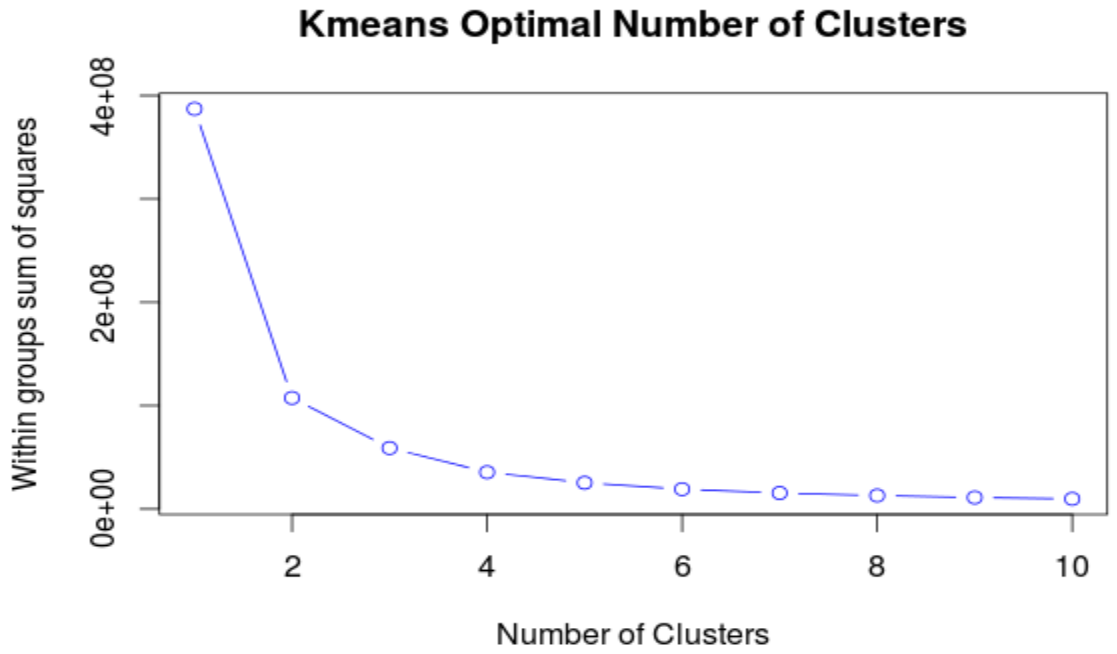
1. **K-Means Clustering**
2. **Model-based Clustering (GMM)**
3. **Compare:**
    a. **Finding optimal clusters**
    b. **Computational Cost (based on execution time)**
    c. **Ease of use**
    d. **Quality of results in terms of visual agreement with thematic classes (e.g., buildings, roads, waters, grass, trees, …)**

Solution :

In order to perform clustering on the given satellite image, the pixel data is converted into raster data using "rgdal" R package. Now this dataset is converted into R data.frame for further operations.

Sampling : A custom function is written to sample a random points from a continuous bin size. Eg. Selecting 1 point randomly from every bin of size 10. [Refer Code cluster.R]

K-Means Clustering : After sampling, the sampled data is used to generate the model of K-Means Clustering algorithm. This model is then used to classify the unlabeled data. In the mean time, the optimal number of clusters is obtained comparing the sum of squared errors for various k clusters as parameter.



From the above plot, using elbow method we can estimate the optimal number of clusters. Hence the parameter k i.e. number of clusters = 6

Now performing Kmeans clustering using k =6,
Code : cluster.R (sample code shown below)

```
kClusters <- 6
km = kcca(sampleData, k=kClusters)
pred <- predict(km, imgData[-sampleIndex,1:3])

imgData$clust <- 0
imgData[sampleIndex, 6] <- km@cluster
imgData[-sampleIndex, 6] <- pred              # Merging the results
```

Computation Time : 29 secs
Ease of Use : Very easy to use as the functions are well explained in the R package "flexclust"

Model-based Clustering (GMM)

Similar to the k-means approach the image data is sampled using the same customized sampling function. This time the bin size is increased to have less samples as GMM clustering method is computationally expensive.

Code : cluster.R (Sample code shown below)

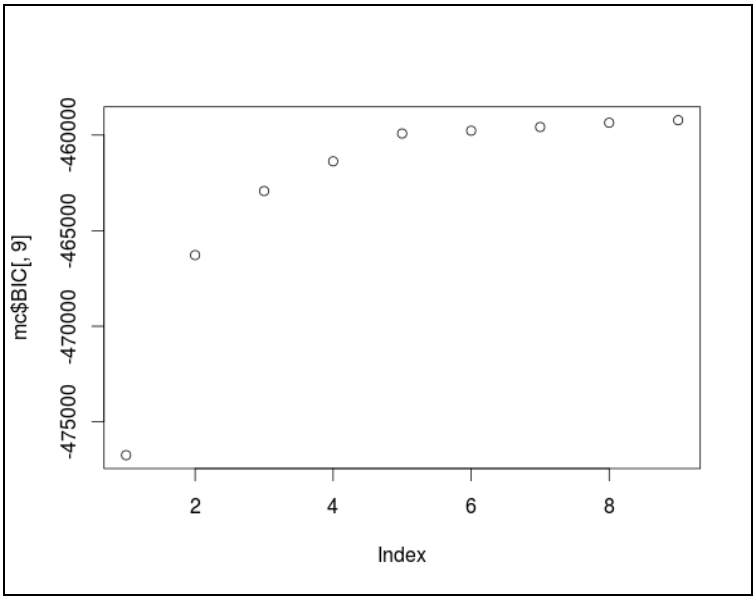
```
sampleIndex <- getSampleIndex(imgData, binSize=50)
sampleData <- imgData[sampleIndex,1:3]
mc = Mclust(sampleData)
predM <- predict(mc, imgData[-sampleIndex,1:3])

imgData$clust <- 0
imgData[sampleIndex, 6] <- mc$classification
imgData[-sampleIndex, 6] <- predM$classification

plot(mc$BIC[,9])  # best
```

Optimal Clusters : It can be verified by plotting BIC (Bayesian Information Criterion) values for various cluster index.

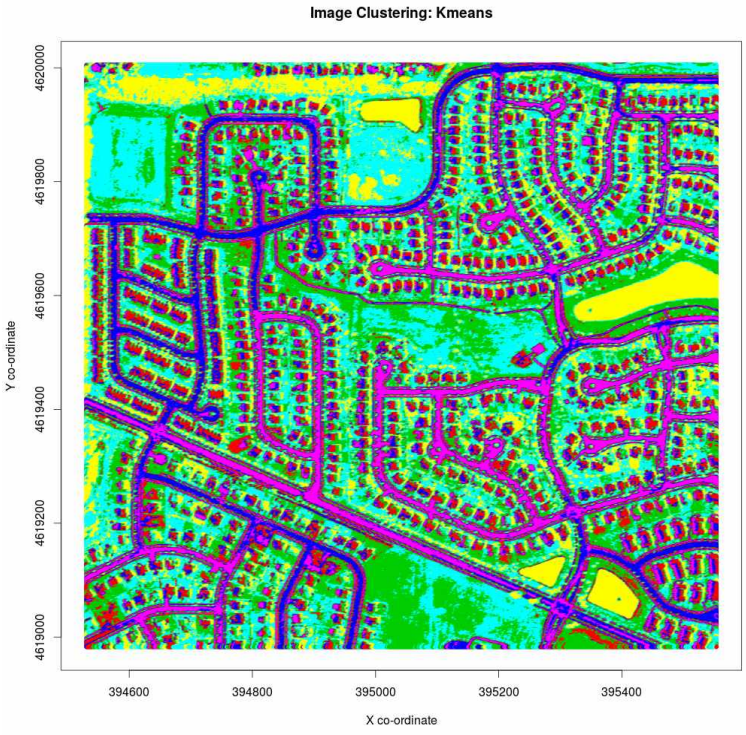
From the plot, we can find the optimal clusters between 6-7.

Computation Time : 61 min 3 sec

Ease of Use : Very easy to use as the functions are well explained in the R package "mclust"



Visual Comparison : **Original Image : Input**



K-means Clustering : Output **GMM Clustering : Output**

a. We can visually conclude that GMM clustering technique is more in visual agreement(better) as compare to Kmeans when compared with original image input.
b. Roads in Kmeans have clustered in 2 different class whereas GMM has clustered roads in single class. But tree layers in upper side of the image is misclassified in both.

Q 2. For given spatial dataset (cancer-data.csv), which contains x and y coordinates (in kilometers) with spatial resolution of 100 meters, do spatial clustering (DBSCAN). Through visual exploration, choose appropriate parameters for DBSCAN clustering. Answer the following:

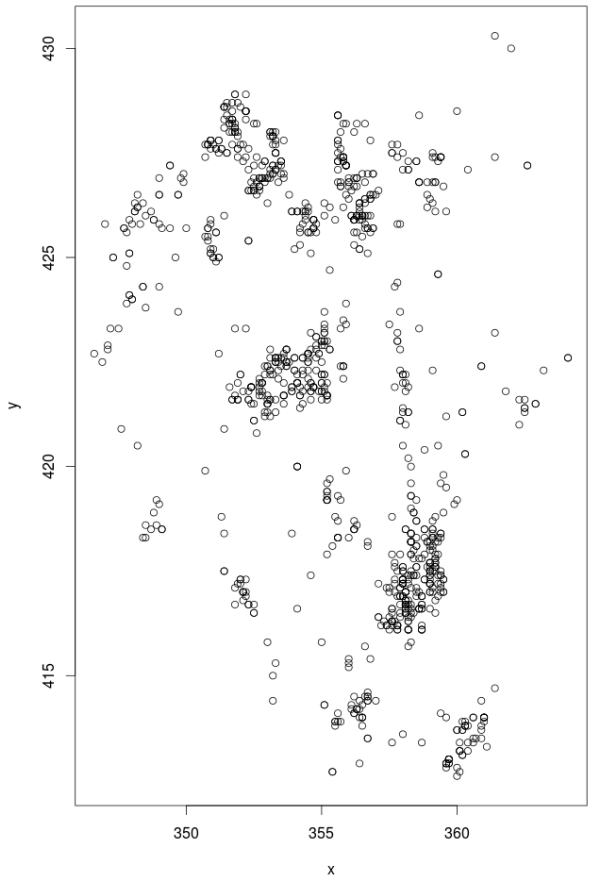
a. **Report parameters**
b. **Generate plot for raw data**
c. **Generate plot for cluster results (mark each cluster with different color; mark noise points with separate color or symbol)**

Solution :

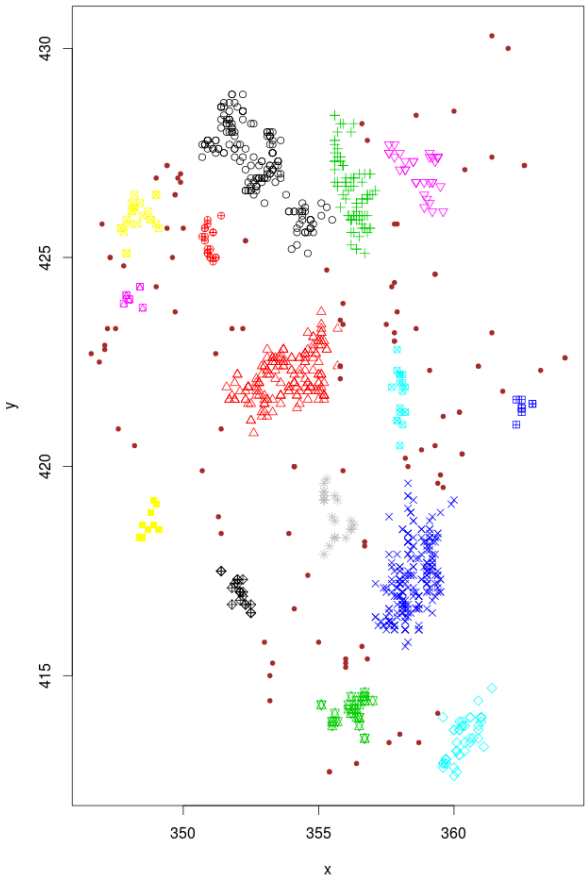
Parameters : Epsilon radius = 0.6 unit and MIN_PTS = 8

Code:

```
library(fpc)
cancerData <- read.csv("hw2/cancer-data.csv")   # Loading Dataset
dd <- dbscan(cancerData, eps=.6, MinPts=8)      # Performing DBScan
par(mfcol=  c(1,2))
plot(cancerData)                                # Raw Data Plot
plot( cancerData, col=dd$cluster, pch=dd$cluster)
points(cancerData[which(dd$cluster == 0),], pch=20, col="brown")
```



Plot : Raw Data Plot



Plot: DBScan Plot

Q3. Compare and contrast C-DBSCAN algorithm [1] and DBCluC algorithm [2]. In addition to similarities and dissimilarities, comment on computational complexity, ease of implementation, ease of use. For each algorithm, list one spatial application that is best suited for the algorithm.

C-DBScan	DBCluc
Similarities: 1. C-DBScan is a clustering algorithm that enforces instance level constraints on the data to drive cluster construction and hence falls under the hood of semi-supervised learning. 2. This algorithm is an extension of original DBScan[3] algorithm that devises neighbour based density heuristics to form clusters. For clustering purposes it uses the same density reachability and connectivity rules of DBScan, 3. At the implementation level, C-DBScan partition data into small grids and performs the neighbourhood query from an advanced tree data structure called kd-tree. 4. Since the clustering happens in the density based hierarchical fashion i.e. from local clusters to merge into neighbourhood clusters, it is not sensitive to the order of input.	1. DBCluc is also a clustering algorithm that takes advantage of constraint modeling to efficiently cluster data while considering all physical constraints. As a constraint based clustering method, it also falls in the category of semi-supervised learning. 2. This algorithm also extends the density notion of DBScan that involves correlation between a data-point and its neighbours. 3. Similar to C-DBScan, this algorithm also adopts the neighbourhood query approach from similar data structure called SR-tree. The difference lies on the fact that it implements range neighbourhood queries rather than nearest neighbour query. 4. Once the algorithm have modeled obstacles using the polygon reduction algorithm, DBCluC starts the clustering procedure from an arbitrary data point. Hence it is also not sensitive to an order of the data input.
Dissimilarities: 1. This algorithm can be used to detect constraint based clusters for multi-dimensional data.	1. The algorithm is constrained for two dimensional dataset and thus more suitable for spatial datasets with physical constraints.

2. The constrains are captured at the instance level applying the constraints of **"must link"** connectivity and "**cannot link**" connectivity. 3. The clustering merging technique is based on hierarchical clustering approach where the adjacent clusters are merged at top level while keeping the constraint of must link and cannot link. 4. The constraints of must link and cannot link are external to the system and hence the effectiveness of the method is based on the quality of these external constraints. In the spatial domain, capturing such constraints are computationally expensive and equally challenging.	2. Here the constraints are captured as obstacles and the modeling of such physical obstacles is done using **polygons** as obstacles and the object visibility the clustering criterion. 3. The clustering technique here merges the neighbour based on range queries and visibility properties. It does not follow any hierarchical order of local clusters to merging of clusters. 4. The constraints imposed for the clustering purpose are internal to the system. It performs obstacle identification and its maintains its visibility properties with data points internally. Hence the quality of the results can be controlled by maintaining the efficient obstacle constraints. Due to its internal constraints maintenance property it is more suitable for spatial clustering.
Computation Complexity: C-DBScan uses Kd-tree for its data point comparison and clustering. The creation complexity of this data structure is in order of O(kNlogN) and then for constraints comparison and merging steps are of complexity O(N) . Hence overall complexity O(kNlogN)	DBCluC claims the time complexity to be O(NlogN) where N is the number of data points. However this linearithmic complexity favors only if the obstacles known are indexed.
Ease of Implementation: The core complexity lies in this algorithm is due to must link and cannot link constraints maintenance while clustering. It checks cannot link constraints and create local clusters. Followed by the creation step, it performs clusters merging and then	DBCluC is divided into three parts with its own complexity. In the first part, the polygon reduction takes place which replaces polygon edges with a loss-set set of primitive edges with respect to visibility. Then it combines the neighbourhood based

separating based on must link and cannot link constraints. Since these constraints computation are external to the algorithm, hence implementation wise it follows comparatively less number of steps, making it somewhat easy to implement.	on range neighbourhood queries while having visibility and DBScan constraints. In short, the complexity is less but the amount of work for the implementation is more, hence comparatively hard to implement. Also it cannot extends for higher dimensional data, so mostly suitable for spatial databases.
Spatial Application: 1. Due to its hierarchical nature of density based clustering, it can be used for navigation and track finding. Also it can be extended to find trajectories.	1. Clustering of Satellite images of land or field types having multiple obstacles as roads, rivers, buildings, mountain etc.modeled as polygons.

References:

[1] Carlos Ruiz., et. al. C-DBSCAN: Density-Based Clustering with Constraints.
[2] Osmar R. Zaiane, et. al. Clustering Spatial Data in the Presence of Obstacles: a Density-Based Approach.
[3] Ester, Martin, et al. "A density-based algorithm for discovering clusters in large spatial databases with noise." Kdd. Vol. 96. No. 34. 1996.