My Comprehensive Evaluation

A Comprehensive Evaluation Report
Presented to
The Statistics Faculty
Amherst College

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Arts
in
Statistics

Kaitlyn E. Haase

February 2019

Acknowledgements

I want to thank my family.

Table of Contents

List of Tables

List of Figures

Abstract

In recent years, the amount of geographic data has increased immensely. With new technology, the accuracy and complexity of data has also improved. This has provoked statisticians to create techniques to best analyze and draw conclusions from this new-found data. Earlier techniques of spatial data were not equipped to handle the complexity and amount of present data. This project first explores how and why we analyze data based on geographic information. The project will then focus on the CLARANS (Clustering Large Applications based on RANdomized Search) algorithm, which is an extension of both the PAM (Partitioning Around Medoids) and CLARAS (Clustering LARge Applications). Example data will be used to demonstrate CLARA, and the project will conclude with both an evaluation of the CLARA method and a model to predict cluster.

In recent years, the amount of geographic data has increased immensely. With new technology, the accuracy and complexity of data has also improved. This has provoked statisticians to create techniques to best analyze and draw conclusions from this new-found data. Earlier techniques of spatial data were not equipped to handle the complexity and amount of present data. This project first explores how and why we analyze data based on geographic information. The project will then focus on the CLARANS (Clustering Large Applications based on RANdomized Search) algorithm, which is an extension of both the PAM (Partitioning Around Medoids) and CLARAS (Clustering LARge Applications). Example data will be used to demonstrate CLARA, and the project will conclude with both an evaluation of the CLARA method and a model to predict cluster.

Introduction

Spatial analysis is analyzing data based on gepgraphic information. This includes topological, geometric, and geographic information. For example, spatial data may include latitude and longtitude, zip code, or street address.

0.1 Why Analyze Spatial Data?

We want to analyze spatial data because there is so much of it available. Analyzing spatial data can help us find dissimilarities and similarities amoung people, places, and locations. Spatial data can help our society allocate resources to areas that need them most, discover changes over time, and BLANK.

0.2 Big Picture Analyzing Spatial Data Algorithms

There are many algorithms out there that handle spatial data.

0.2.1 Classification vs. Clustering

Two categories of how to analyze spatial data include classification and clustering. Classification groups data items together into categories according to their properties. It is considered supervised classification because it needs a training dataset to fit the classification model and a test dataset to evaluate the model.

Clustering is organizing a set of data items into groups so that items in the same group are similar to each other and different from those in other groups [Rec 1]. Clustering is helpful in finding patterns and similarities/differences between data points and groups; however it can be quite subjective, as we will discuss later on in the project.

Chapter 1

Spatial Clustering Methods

There are many factors to consider when chosing a clustering algorithm, such as the application of the problem (what do you want to find out about this data?), quality vs speed trade off (size of data plays a role), characteristics of the data (i.e. numeric distance measures), dimensionality (typically as dimension increases the time it takes to run the method increases and quality of the data clusters decrease), and outliers (some methods are very sensitive to outliers) [Rec 2].

1.1 Types of Clustering: Partitioning and Hierarchical

Two of the main types of clustering are partitioning and hierarchial.

Hierarchial clustering organizes data items into a hierarchy with a sequence of nested partions or groupings [Rec 1, p. 405]. There is the bottom-up approach: There is also the top-down approach:

Partitioning cluster methods divide a set of data items into a number of non-overlapping clusters. A data item is typically assigned to a cluster based on a proximity or dissimilarity measure [Rec 2, p. 405]. Partitioning clustering algorithms classifies the data into K groups by satisfying both that each group has at least one data point, and that each data point belongs to exactly one group. [Rec 5, p. 18].

1.2 How to Create Clusters: *K*-Means vs *K*-Medoids

K-means algorithm and k-medoid algorithm are two examples of partitioning algorithms. They both ise iterative processes to find K clusters; however, they use different ways to represent these clusters.

1.2.1 *K*-Means

K-means algorithm represents its n observations in k groups, with the center of the groups being the mean/average observation. The goal of the algorithm is to find k centroids, one for each cluster. In order to do this, we must minimize an *objective function*, which is the squared error function for k means. The objective function is:

$$O = \sum_{j=1}^{k} \sum_{i=1}^{j} ||X_i^{(j)} - C_j||^2$$

Where $|X_i^{(j)} - C_j|$ is an indicator of the distance of the data points from their cluster centers.

The steps of the algorithm are as follows:

1. Choose K points in the space to represent the centroid. This works best if they are chosen to be far apart from eachother.
2. Assign each object to the cluster with the closest centroid.
3. When all of the clusters have been made, recalculate the positions of the K centroids.
4. Repeat steps 2 and 3 until the centroids no longer move.

This algorithm always terminates; however, it is sensitive both to outliers and to the initial randomly selected K cluster centers. Therefore, the algorithm should be run multiple times to reduce the effects from this sensitivity. [Rec 5, p. 18].

1.2.2 *K* Medoids

On the contrary, instead of taking the mean value of the objects in a cluster, the k-medoid method uses the most centrally located object in a cluster to be the cluster center [Rec 2]. This causes the method to be less sensitive to outliers, but also requires more time to run.

same steps as K-means except BLANK (p. 6 in Rec 2) steps in Rec 5, p. 19 for k-medoids

Rec 5 p. 18-19 ***

```
#How to insert a figure, make sure amherst.png is in main directory
#label(path = "figure/amherst.png", caption = "Amherst logo",
    #  label = "amherst", type = "figure")
```

1.3 How to Choose *K*

Many ways to choose k, which is why these methods are so subjective.

I will describe two of the many ways to determine K, both of which use visuals to determine what value of k is appropriate for the data.

1.3.1 Elbow Method

To start, the elbow method looks at the total within-cluster sum of squares (WSS) and determines when there are enough clusters so that the next cluster does not improve the total WSS very much. This would be the appropriate K.

The steps for this algorithm are as follows:

1. Compute the clustering algorithm for different values of k (i.e. k from 1 to 10).
2. For each k, calculate the total WSS.
3. Plot the curve of the total WSSs according to the number of clusters (k).
4. The location of the bend in the plot is generally considered an indicator for the appropriate number of clusters.

1.3.2 Silhouette Method

The Silhouette Method focuses on the quality of clustering. A high average silhouette width indicates a good clustering (how well each object lies within its cluster).

The steps of the Silhouette Algorithm are:

1. Compute clustering algoritm for different values of k (i.e. k from 1 to 10).
2. For each k, calculate the average silhouette of observations.
3. Plot the curve of the average silhouettes according to the number of clusters (k).
4. The location of the maximum is considered the appropriate number of clusters.

–> datanovia website

1.4 PAM

Partitioning Around Medoids (PAM) is a k-medoid method that iterates through all the k cluster centers and tries to replace the center with one of the other objects (n-k possibilities). [rec 2]. For a replacement to occur, the squared error function must decrease (if it does not decrease, there is no replacement). The algorithm eventually terminates with a local optimum.

The total complexity of PAM in one iteration is **formula: O(k(n-k)ˆ2) (o= each non-medoid data point, k= # of cluster centers, (n-k) objects to compare to, and (n-k) operations for calculating E). This makes for a costly computation when n is large. Works best for n= 100, k=5.

Explanation of PAM, REC 6, P. 146–> 4 cases, and algorithm

1.5 CLARA

Because PAM does not scale well to large data sets, Clustering LARge Applications (CLARA) was developed to deal with larger data sets.

CLARA is a sampling based method, meaning a sample of the data is used to represent the entire data set. Medoids are chosen from this sample data using PAM and then "the average dissimilarity is computed using the whole dataset" (**don't know what "average dissimilarity" means or how it is calculated). If a new set of medoids gives a lower dissimilarity than a previous best solution, then the best solution is replaced with a new set of medoids [Rec 2, p. 7].

1.6 CLARANS (?)

Chapter 2

Example

2.1 Exploring the Data

Data came from Stat 495 final project. (use info from project...). Needed a sample of 1000...

Importing the data:

```
#using data from final stat 495 project
library(readr)
data_subset <- read_csv("CopyOfdata_subset.csv")

set.seed(1)
#getting a sample of 1000 observations
mysample <- data_subset[sample(1:nrow(data_subset), 1000,
   replace=FALSE),]
```

Picking variables to focus on–> expanding conclusions from Stat 495 project

```
#only keeping the variables I want to look at
myvars <- c("Latitude_tri", "Longitude_tri", "poor_or_fair_health", "poor_physical
smallsample <- mysample[myvars]
```

2.2 Applying CLARA

Step 1: finding k

```
#finding k with project data, using Elbow Method
pkgs <- c("factoextra",  "NbClust")
install.packages(pkgs)

library(factoextra)
library(NbClust)
library(ggplot2)

# Elbow method
fviz_nbclust(new, kmeans, method = "wss") +
    geom_vline(xintercept = 4, linetype = 2)+
  labs(subtitle = "Elbow method")
```

Step 2: Run CLARA function

```
new<- na.omit(smallsample)

## run CLARA
clarasamp <- clara(new[1:6], 4)
```

```
## print components of clarax
print(clarasamp)
summary(clarasamp)
```

```
## plot clusters
plot(new, col = clarasamp$cluster)
## plot centers
points(clarasamp$centers, col = 1:2, pch = 8)
```

```
#plotting clara
factoextra::fviz_cluster(clarasamp)
```

2.3 Evaluation of CLARA

2.3.1 Model to Predict Cluster

First, had to include a cluster variable in the original data set, using the data provided by the CLARA function.

```
#adding each data point's cluster #
cluster<- clarasamp$clustering
cluster_data<- cbind(new, cluster)
```

Conclusion

If we don't want Conclusion to have a chapter number next to it, we can add the `{.unnumbered}` attribute. This has an unintended consequence of the sections being labeled as 3.6 for example though instead of 4.1. The LaTeX commands immediately following the Conclusion declaration get things back on track.

More info

And here's some other random info: the first paragraph after a chapter title or section head *shouldn't be* indented, because indents are to tell the reader that you're starting a new paragraph. Since that's obvious after a chapter or section title, proper typesetting doesn't add an indent there.

Appendix A

The First Appendix

This first appendix includes all of the R chunks of code that were hidden throughout the document (using the `include = FALSE` chunk tag) to help with readibility and/or setup.

In the main Rmd file:

```
# This chunk ensures that the acstats package is
# installed and loaded. This acstats package includes
# the template files for the thesis and also two functions
# used for labeling and referencing
if(!require(devtools))
  install.packages("devtools", repos = "http://cran.rstudio.com")
if(!require(acstats)){
  library(devtools)
  devtools::install_github("Amherst-Statistics/acstats")
}
library(acstats)
```

In :

Appendix B

The Second Appendix, for Fun

References

Angel, E. (2000). *Interactive computer graphics : A top-down approach with opengl*. Boston, MA: Addison Wesley Longman.

Angel, E. (2001a). *Batch-file computer graphics : A bottom-up approach with quicktime*. Boston, MA: Wesley Addison Longman.

Angel, E. (2001b). *Test second book by angel*. Boston, MA: Wesley Addison Longman.