Gun Violence in the US. Application of Unsupervised Learning Methods for Trend Exploration

by Sumaira Afzal, Viraja Ketkar, Murlidhar Loka, Vadim Spirkov

Abstract To do

Background

Objective

The objective of this research is to ...

Data Analysis

The data set used for this research contains 260k of gun violence incidents in the US between January 2013 and March 2018. The data has been sourced from Kaggle.

Originally the data set was uploaded to Kaggle from Gun Violence Archive (GVA) Web site gunviolencearchive.org. This is a not for profit corporation formed in 2013 to provide free online public access to accurate information about gun-related violence in the United States. GVA will collect and check for accuracy, comprehensive information about gun-related violence in the U.S. and then post and disseminate it online.

Data Dictionary

Column Name	Column Description
incident_id	Incident ID
date	Date of crime
state	State
city_or_county	City/county of crime
address	Address of the location of the crime
n_killed	Number of people killed
n_injured	Number of people injured
incident_url	URL regarding the incident
source_url	Reference to the reporting source
incident_url_fields_missing	TRUE if the incident_url is present, FALSE otherwise
congressional_district	Congressional district id
gun_stolen	Status of guns involved in the crime (i.e. Unknown, Stolen, etc...)
gun_type	Typification of guns used in the crime
incident_characteristics	Characteristics of the incidence
latitude	Location of the incident
location_description	Description of the location
longitude	Location of the incident
n_guns_involved	Number of guns involved in incident
notes	Additional information of the crime
participant_age	Age of participant(s) at the time of crime (victims and suspects)
participant_age_group	Age group of participant(s) at the time crime
participant_gender	Gender of participant(s)
participant_name	Name of participant(s) involved in crime
participant_relationship	Relationship of participant to other participant(s)

Column Name	Column Description
participant_status	Extent of harm done to the participant
participant_type	Type of participant (victim or suspect)
sources	Participants source
state_house_district	Voting house district
state_senate_district	Territorial district from which a senator to a state legislature is elected.

Data Exploration

Firstly we are going to load and examine content and statistics of the data set

```
data = read.csv("../data/gun-violence-data_01-2013_03-2018.csv", header = T,
                na.strings = c("NA","","#NA"),sep=",")
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu % Date and time: Tue, Mar 05, 2019 - 5:55:52 PM

Table 2

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
incident_id	239,677	559,334.300	293,128.700	92,114	308,545	817,228	1,083,472
n_killed	239,677	0.252	0.522	0	0	0	50
n_injured	239,677	0.494	0.730	0	0	1	53
congressional_district	227,733	8.001	8.481	0.000	2.000	10.000	53.000
latitude	231,754	37.547	5.131	19.111	33.903	41.437	71.337
longitude	231,754	−89.338	14.360	−171.429	−94.159	−80.049	97.433
n_guns_involved	140,226	1.372	4.678	1.000	1.000	1.000	400.000
state_house_district	200,905	55.447	42.048	1.000	21.000	84.000	901.000
state_senate_district	207,342	20.477	14.205	1.000	9.000	30.000	94.000

Initial observation of the data shows that there is a number of features which do not present any analytical value (Table: **??**). They are:

- *incident_id*
- *incident_url*
- *source_url*
- *state_house_district*
- *state_senate_district*
- *congressional_district*
- *sources*
- *incident_url_fields_missing*

We also going to drop *participant_age* feature in favor of the *participant_age_group*. The age group is more suitable for categorization and has much less missing data (16% vs 39%).

The remaining features could be grouped as follows...

Participant Features. This group describes suspects and victims found on the crime scene. The content of the features of this group is structured as follows: *[idx1::value1||idx2::value2]* (see Table: **??**). This is not quite acceptable for the analytics, thus the participant related features would have to be parsed to extract valuable information about the crime.

It is feasible. *utils.R* script contains *parseFeature* function, which parses *[idx1::value1||idx2::value2]* structure and returns a named vector object. For example a *participent_type* could be structured as follows:

0	1	2	3
Victim	Victim	Subject-Suspect	Subject-Suspect

Unfortunately *participant_relationship* feature missing **93%** of values. It is not possible to impute the missing data thus we will drop it. For obvious reasons we are also going to get rid of *participant_name*. The rest of the participant-related features will be parsed and replaced wit the new categorical attributes. In order to do so we have to understand what possible values each participant-related feature can have. for this we will employ text mining technique.

We begin with *participant_type* feature

```
participantType = data %>%  mutate(text = trimws(gsub('\\|\\||:|\\|'," ",participant_type,
        fixed = F))) %>% filter(text != "0" ) %>% select(text)

pCorups = VCorpus(VectorSource(participantType))
pCorups  = tm_map(pCorups , removeNumbers)
pTermMatrix = tm::TermDocumentMatrix(pCorups)
# count frequent words
print(tm::findFreqTerms(pTermMatrix, 10))
```

```
[1] "subject-suspect" "victim"
```

As we can see the *participan type* may have two values *vitim* and *subject-suspect*. If the *participant type* is missing we will consider it as **unknown**. Thus we will be employing *participant type* feature as a basis to impute all other participant stats.

Let's find the possible values of *parctipant_age_group* feature (the coded is omitted).

```
[1] "adult" "child" "teen"
```

Further examination of the feature data shows that there the age group values are:

- Adult 18+
- Teen 12-17
- Child 0-11

Thus using *participant_age_group* feature data we will create two new ones: *vicitm_age_group* and *suspect_age_group*. These new categorical features will be coded as follows:

- 0 - no info
- 1 - all adults
- 2 - children/ teens
- 3 - adults and children/ teens . Adults make majority
- 4 - adults and children/ teens. Children/ teens make majority

participant_gender could also be parsed and replaced with the coded categorical features as described below.

```
participantGender = data %>%  mutate(text = trimws(gsub('\\|\\||:|\\|'," ",participant_gender,
        fixed = F))) %>% filter(text != "0" ) %>% select(text)
pCorups = VCorpus(VectorSource(participantGender))
pCorups  = tm_map(pCorups , removeNumbers)
pTermMatrix = tm::TermDocumentMatrix(pCorups)
print(tm::findFreqTerms(pTermMatrix, 10))
```

```
[1] "female" "male"
```

As a result we will be adding two new features:

- *victim_gender* - gender of the victims
- *suspect_gender* - gender of the suspects

Gender Codes

- 0 - no info
- 1 - male
- 2 - female
- 3 - male dominated group
- 4 - female dominated group

The last feature of the group is *participant_status*. It maintains the outcome of the incident. Let's review the content of the attribute.

```
[1] "arrested"  "injured"   "injured,"  "killed"    "killed,"   "unharmed"
[7] "unharmed,"
```

Based on our findings we will be creating three new numerical features:

- n_victim_killed - number of victims killed
- n_victim_injured - number of victims injured
- n_arrested - number of suspects arrested

Gun Related Features. There are three attributes that describe gun types: *gun_stolen*, *gun_type* and *n_guns_invoved* (Table: **??**) *gun_type* and *gun_stolen* have similar to the participant-related features encoding (*[idx1::value1||idx2::value2]*). Thus they also could be parsed and substituted with the categorical features. We begin with the gun type.

Figure 1: The Most Frequently Used Gun Types

Employing simple text mining techniques we can see that **handgun**, **rifle**, **shotgun** and **auto** make the majority. Thus we will add another new feature *gun_type_involved* to categorize the gun types as follows:

- 0 - unknown
- 1 - handgun
- 2 - shotgun/ rifle
- 3 - automatic
- 4 - mix/other

gun_stolen attribute tells if the gun was stolen or acquired legally. We are going to create a new categorical feature - *gun_origin* which would maintain the following data:

- 0 - unknown
- 1 - all stolen
- 2 - all acquired legally
- 3 - mix of stolen and legal guns

Location Related Features. To analyze geography of the crimes we will be employing *state*, *city_or_county*, *latitude* and *longitude* attribute. since we have the coordinates the *address* feature does not present much value for unsupervised learning. We will be using it though to impute missing latitude and longitude values. This activity will be covered in greater details in **Missing Data** paragraph.

Descriptive Features. *notes*, *location_description* and *incident_characteristics* are free-text features that might provide additional insights about the crime scene. We are going to take a close look at each feature and decide if we could utilize it.

Lets' begin with the *notes*

Figure 2: Most Common Words in Notes

Unfortunately *notes* feature does not provide more knowledge to what the others features already supply. Thus it will be dropped.

location_description on the other hand, could be useful to classify location type. Unfortunately 82% of the data is missing. Nonetheless this feature appears to be too important to ignore. Let's see how much we can salvage.

Figure 3: Most Common Bi-gram Terms in Location Decription

As plot 3 shows we could utilize bi-gram terms if the location description is available. Thus let's add new categorical feature - *place_type*, which would have the following values:

- 0 - unknown
- 1 - school/ university/ college
- 2 - community center/ shopping center/ hospital/ church
- 3 - home invasion
- 4 - street/drive by
- 5 - other public places

The last feature in the group is *incident_characteristics*. The feature is almost 100% populated. As with the previous two we are going to find out what info it maintains.

```
print(xtable(tmp[1:50,1:2],
      caption = "\\tt Most Frequently Used Incident Characteristics  Bi-grams"),
      row.names = FALSE,  include.rownames = FALSE)
```

word	freq
shot wounded	93926.00
wounded injured	93313.00
dead murder	53409.00
murder accidental	53409.00
shot dead	53272.00
accidental suicide	52967.00
shots fired	45895.00
nonshooting incident	44761.00
officer involved	38229.00
injured shot	37426.00
fired no	35750.00
no injuries	35552.00
found commission	30863.00
guns found	30863.00
commission crimes	30720.00
possession guns	30646.00
involved incident	23860.00
subject suspect	21886.00
suspect perpetrator	21881.00
home invasion	21244.00
injury death	20540.00
incident officer	20166.00
death evidence	19723.00
evidence dgu	19723.00
robbery injury	19723.00
armed robbery	19502.00
dgu found	19383.00
carry lost	19017.00
flourishing open	19017.00
open carry	19017.00
lost found	18975.00
brandishing flourishing	18519.00
confiscation raid	17991.00
le confiscation	17991.00
atf le	17966.00
raid arrest	17959.00
felon prohibited	17165.00
gun felon	17165.00
prohibited person	17158.00
suicide shot	17153.00
possession gun	17137.00
drug involvement	16976.00
defensive use	16824.00
found shot	16689.00
involved shooting	16658.00
accidental shooting	15641.00
arrest possession	14942.00
car street	13655.00
street car	13655.00
car car	13630.00

Information the *incident_characteristics* provides proved to be useful. It can support two features: *place_type*, which was introduced above and *inceident_type*. The *incident_type* is going to be a categorical attribute with the following codes:

- 0 - unknown
- 1 - accidental
- 2 - defensive use
- 3 - armed robbery
- 4 - suicide
- 5 - raid/ arrest/ warrant
- 6 - domestic violence
- 7 - gun brandishing, flourishing, open demonstration _ We are also be adding a feature that indicates if drugs or alcohol was involved: *is_drug_alcohol*

Date Feature In addition to *date* of incident attribute we add *month* and *day_of_week* to identify any seasonal patterns.

Data Preparation

Prior to generating new features as discussed in the previous paragraph we would need to impute missing latitude and longitude data. To do so we employ OpenCage forward geocoding API. Unlike Google this company offers a free tier. To save time we imputed the missing geo-coordinates and saved the result in the file. The code below is submitted for demonstration purpose only.

```
imputeCoordinates()
```

Now we are going to remove the features identified as redundant

```
data = subset(data, select = c(-incident_id, -incident_url, -source_url,
  -state_house_district, -state_senate_district,-sources, -incident_url_fields_missing,
  -congressional_district, -address, -participant_age, -participant_name,
  -participant_relationship, -notes))
```

Lastly we are going to loop through the entire data fame imputing missing data and adding new features. Again this is a lengthy process that takes about 1.5 hours to finish. The code is also quite long. Thus in order not to clutter the report we submit the code just in the script to illustrates the process, but will not output it into the report.

After we added new feature it is time to remove the columns that are no longer relevant and save the result into a file to be used for unsupervised learning

```
data = subset(data, select = c(-participant_age_group, -participant_type,
      -participant_gender, -participant_status, -location_description,
      -incident_characteristics, -gun_stolen,-gun_type))
```

Resulting Dataset

After a rather lengthy process, we finaly have reached the stage when our data set is eady to be used for exploration by clustering algorithms. All missing features have been imputed and free-format text columns have been relaced with the categorical attributes. This is the summary of the resulting data.

```
print(dfSummary(data, valid.col = F, max.distinct.values = 4, headings = F),
      caption = "\\tt Engineered Gun Violence Dataset Summary")
```

Table 4: `Engineered Gun Violence Dataset Summary`

No	Variable	Stats / Values	Freqs (% of Valid)	Missing
1	date	1. 2013-01-01	3 (0.0%)	0
	[factor]	2. 2013-01-05	1 (0.0%)	(0%)
		3. 2013-01-07	2 (0.0%)	
		4. 2013-01-19	1 (0.0%)	
		[1721 others]	239670 (100.0%)	
2	state	1. Alabama	5471 (2.3%)	0
	[factor]	2. Alaska	1349 (0.6%)	(0%)
		3. Arizona	2328 (1.0%)	
		4. Arkansas	2842 (1.2%)	
		[47 others]	227687 (95.0%)	
3	city_or_county	1. Abbeville	37 (0.0%)	0
	[factor]	2. Abbotsford	3 (0.0%)	(0%)
		3. Abbott	1 (0.0%)	
		4. Abbott Township	1 (0.0%)	
		[12894 others]	239635 (100.0%)	
4	n_killed	Mean (sd) : 0.3 (0.5)	16 distinct values	0
	[integer]	min < med < max:		(0%)
		0 < 0 < 50		
		IQR (CV) : 0 (2.1)		
5	n_injured	Mean (sd) : 0.5 (0.7)	23 distinct values	0
	[integer]	min < med < max:		(0%)
		0 < 0 < 53		
		IQR (CV) : 1 (1.5)		

No	Variable	Stats / Values	Freqs (% of Valid)	Missing
6	latitude [numeric]	Mean (sd) : 37.5 (5.2) min < med < max: -39 < 38.6 < 71.3 IQR (CV) : 7.5 (0.1)	107051 distinct values	0 (0%)
7	longitude [numeric]	Mean (sd) : -89.2 (15) min < med < max: -171.4 < -86.2 < 176.2 IQR (CV) : 14.1 (-0.2)	118198 distinct values	0 (0%)
8	n_guns_involved [integer]	Mean (sd) : 0.8 (3.6) min < med < max: 0 < 1 < 400 IQR (CV) : 1 (4.5)	107 distinct values	0 (0%)
9	month [integer]	Mean (sd) : 6.4 (3.4) min < med < max: 1 < 6 < 12 IQR (CV) : 6 (0.5)	12 distinct values	0 (0%)
10	day_of_week [integer]	Mean (sd) : 4.1 (2) min < med < max: 1 < 4 < 7 IQR (CV) : 4 (0.5)	7 distinct values	0 (0%)
11	victim_gender [integer]	Mean (sd) : 0.7 (0.8) min < med < max: 0 < 1 < 4 IQR (CV) : 1 (1.1)	5 distinct values	0 (0%)
12	suspect_gender [integer]	Mean (sd) : 0.6 (0.7) min < med < max: 0 < 1 < 4 IQR (CV) : 1 (1.1)	5 distinct values	0 (0%)
13	victim_age_group [integer]	Mean (sd) : 0.6 (0.7) min < med < max: 0 < 1 < 4 IQR (CV) : 1 (1.1)	5 distinct values	0 (0%)
14	suspect_age_group [integer]	Mean (sd) : 0.6 (0.7) min < med < max: 0 < 1 < 4 IQR (CV) : 1 (1.1)	5 distinct values	0 (0%)
15	n_victim_killed [integer]	Mean (sd) : 0.2 (0.5) min < med < max: 0 < 0 < 49 IQR (CV) : 0 (2.2)	15 distinct values	0 (0%)
16	n_victim_injured [integer]	Mean (sd) : 0.5 (0.7) min < med < max: 0 < 0 < 53 IQR (CV) : 1 (1.6)	23 distinct values	0 (0%)
17	n_victims [integer]	Mean (sd) : 0.8 (0.8) min < med < max: 0 < 1 < 102 IQR (CV) : 1 (1.1)	26 distinct values	0 (0%)
18	n_suspects [integer]	Mean (sd) : 0.8 (1) min < med < max: 0 < 1 < 63 IQR (CV) : 1 (1.2)	33 distinct values	0 (0%)
19	n_arrested [integer]	Mean (sd) : 0.4 (0.8) min < med < max: 0 < 0 < 63 IQR (CV) : 1 (2)	31 distinct values	0 (0%)
20	gun_type_involved [integer]	Mean (sd) : 0.3 (0.8) min < med < max: 0 < 0 < 4 IQR (CV) : 0 (3)	5 distinct values	0 (0%)
21	gun_origin [integer]	Min : 0 Mean : 0 Max : 1	0 : 230802 (96.3%) 1 : 8875 (3.7%)	0 (0%)

No	Variable	Stats / Values	Freqs (% of Valid)	Missing
22	place_type [integer]	Mean (sd) : 0.8 (1.7) min < med < max: 0 < 0 < 5 IQR (CV) : 0 (2)	6 distinct values	0 (0%)
23	incident_type [integer]	Mean (sd) : 1.3 (2.2) min < med < max: 0 < 0 < 7 IQR (CV) : 2 (1.7)	6 distinct values	0 (0%)
24	is_drug_alcohol [integer]	Min : 0 Mean : 0.1 Max : 1	0 : 210310 (87.8%) 1 : 29367 (12.2%)	0 (0%)

Modeling and Evalutation

In this section we will apply various clustering methods to explore gun violence trends in the US. We will use parallel, hierarchical and density-based clustering approaches.

In general the clustering algorithms take time to compute the result. Thus we will be employing a smaller dataset to find and visualize the clusters. We will be exploring the incidents, that have at least three victims and where a place type was recorded.

```
victimStats = data %>% filter(n_victims >= 3 & place_type>0) %>% arrange(date)
```

Before we apply clustering models to the dataset we should assess clustering tendency. In order to do so we will employ **Hopkins** statistics.

Hopkins Statistics

Hopkins statistic is used to assess the clustering tendency of a dataset by measuring the probability that a given dataset is generated by a uniform data distribution. Let's calculcate Hopkins (**H**) statistics for some continuous variables:

- n_guns_involved
- n_victims
- n_suspects
- n_killed
- n_injured

The **H** value close to zero indicates very good clustering tendency. The **H** value around or greater than 0.5 denotes poor clustering tendency. We

```
stats = victimStats[c("n_victims","n_suspects","n_guns_involved","n_killed","n_injured")] %>% scale()

H =  get_clust_tendency(stats,n = 100, graph = F, seed = 6709)
print(H[["hopkins_stat"]])
```

```
[1] 0.03212907
```

Outsanding! **H** value is very close to 0. Let' calculate Hopkins statistics on some categorical features

```
stats = victimStats[c("gun_type_involved","victim_gender","place_type","victim_age_group","suspect_gender")]

H =  get_clust_tendency(stats, n = 100, graph = F, seed = 6701)
print(H[["hopkins_stat"]])
```

```
[1] 0.3370471
```

Well, the categorical data do not seem to be so suatable for clustering, **H** greater than 0.3 is too high. Let's examine the combination of geo-coordinates and some continuous and categorical features.

```
stats = victimStats[c("n_victims","n_suspects","n_guns_involved","n_killed","n_injured",
        "gun_type_involved","victim_gender","place_type","victim_age_group",
        "suspect_gender")] %>% scale()
```

```
H =  get_clust_tendency(stats, n = 100, graph = F, seed = 6701)
print(H[["hopkins_stat"]])
```

```
[1] 0.0502742
```

The **H** number is very encouraging again. It appears we would have to combine continuous with one or too categorical fetatures to get dense, well separated clusters. Now it is time to explore the data

Partitioning Clustering Approach Using CLARA

We decide to select CLARA algorithm because it scales well and can deal with continuos and categorical data. It is based on The **Partitioning Around Medoids** (*PAM*) algorithm, which is a popular realization of k-medoids clustering.We start with univariate data set for simplicty. Then we will take a look at silhouette plot to analyze the result. The silhouette coefficient meauser how well the clusters are seprated. The silhouette analysis also provides data on the cluster density.

Let's see how *n_victim* feature could be split by *CLARA*. The method also scales the data

```
stats = victimStats[c("n_victims")]
clara = clara(stats, k = 4, metric = "euclidean",stand = T, samples = 100, sampsize =500)
fviz_silhouette(clara, title = "")
```

```
  cluster size ave.sil.width
1       1  313          1.00
2       2  102          1.00
3       3   61          0.73
4       4   24          0.28
```

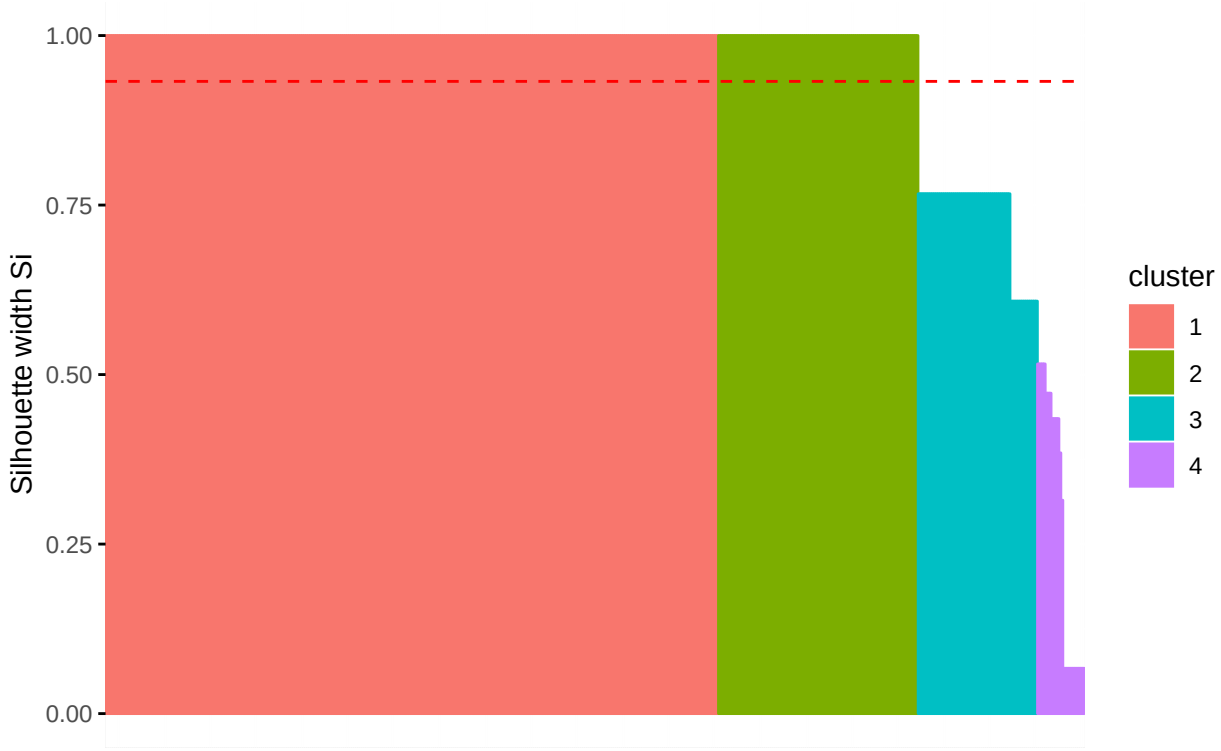

Figure 4: CLARA Approach. Univariate Feature, 4 Cluster Silhouette

```
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank())
```

```
List of 2
 $ axis.text.x : list()
```

```
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ axis.ticks.x: list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 - attr(*, "class")= chr [1:2] "theme" "gg"
 - attr(*, "complete")= logi FALSE
 - attr(*, "validate")= logi TRUE
```

The result looks very good. The clasters are dence and well separated. The last cluster doe not look perfect, probably it contains the incidents, where number of victims is very high. To better interprete the cluster content let's merge the clustering result with the original data set and render a scatter plot.

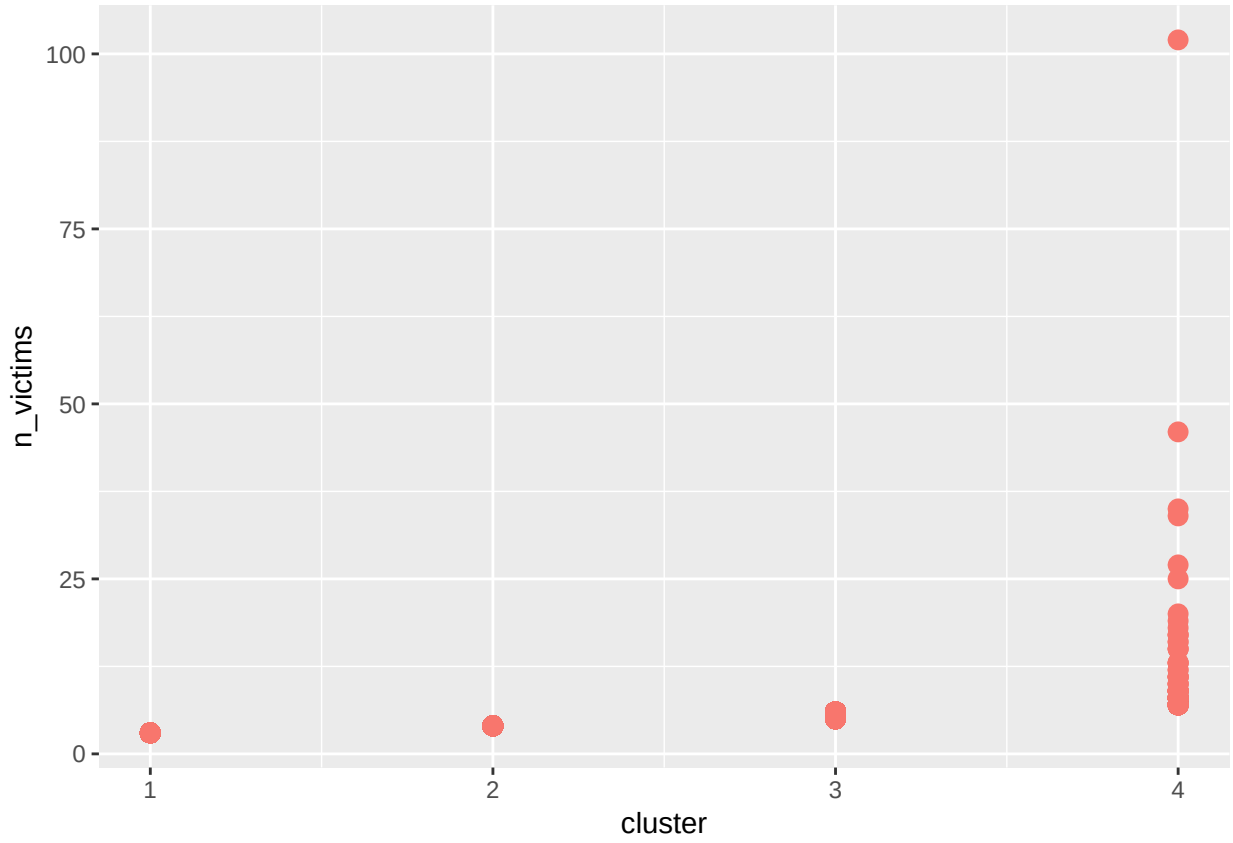

Figure 5: CLARA Approach. Number of Victims/Cluster Scatter Plot

As we can see the cluster number one groups the incidents where number of victims is around 3. The cluster number 2 contains the incidents with 4 victims, the # 3 combines the incidents with the number of victims between 5 and 6. And the rest goes to the cluster #4. Here is the golocation of the clustered incidents

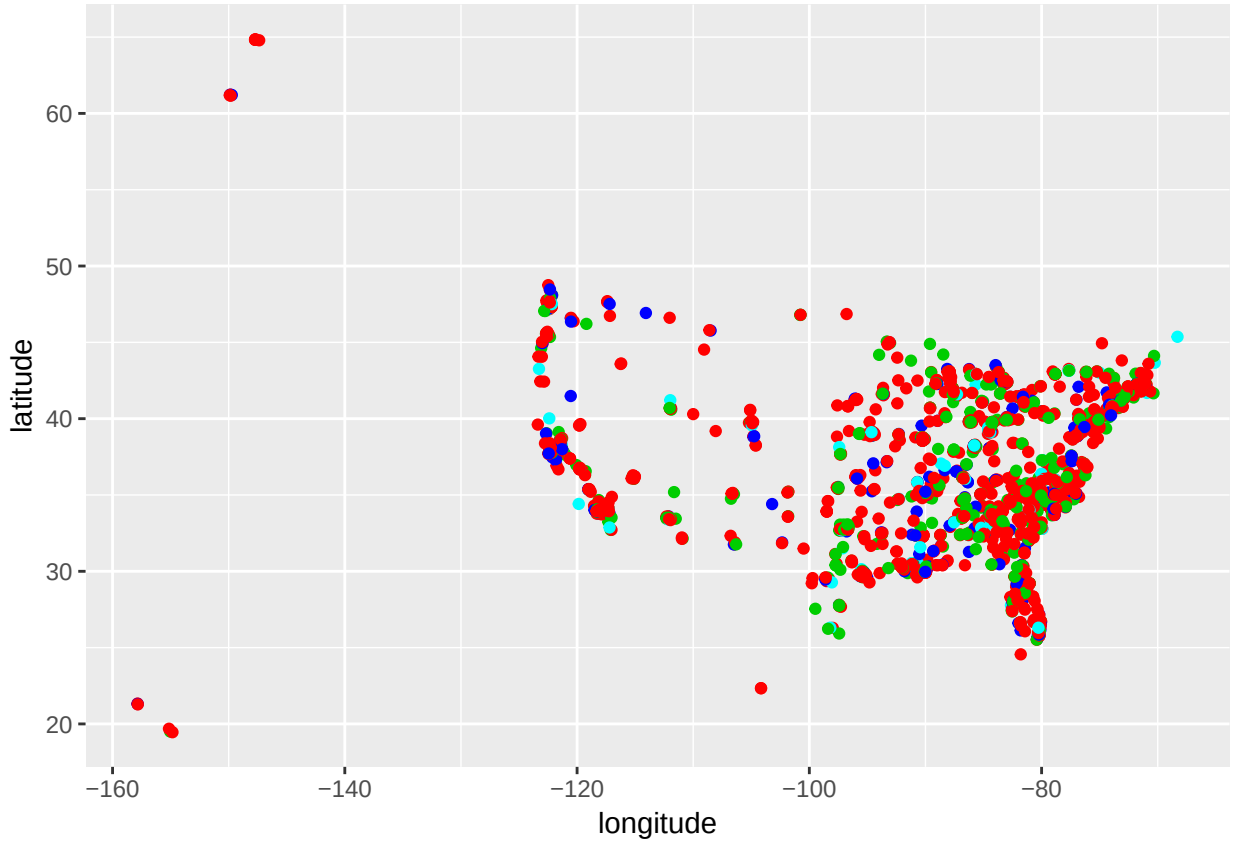

Figure 6: CLARA Approach. Number of Victims Clusters on Map

Very well! Now we are going to add a categorical feature and see if could get meaningful clusters

```
stats = victimStats[c("n_victims","place_type")]
clara = clara(stats, k = 4, metric = "euclidean",stand = T, samples = 100, sampsize =500)
fviz_silhouette(clara, title = "") +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank())
```

```
  cluster size ave.sil.width
1       1  141          0.74
2       2  133          0.54
3       3   87         -0.27
4       4  139          0.75
```

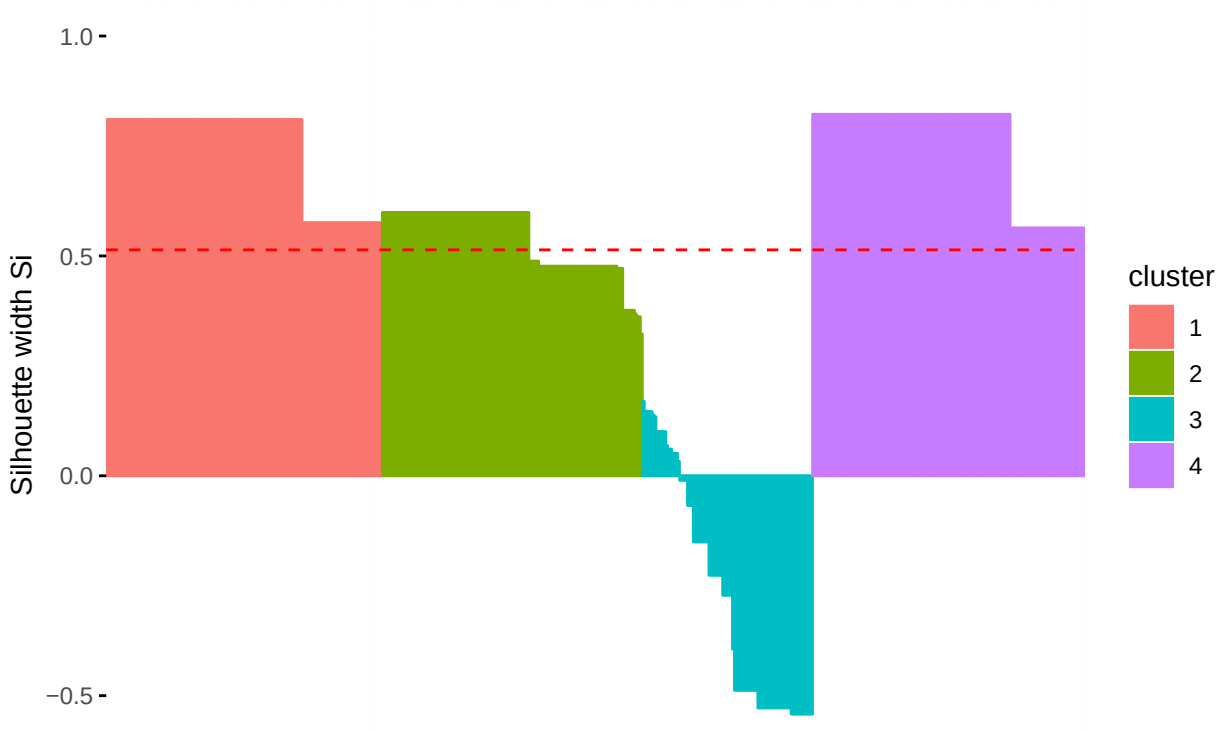

Figure 7: CLARA Approach. Two Feature, 4 Cluster Silhouette

Unfortunately the result is not great. Average silohuette coefficient is about 0.5; one cluster has a negative value. We have made quite a few experints with the various number of clusters and metrics without much success (the results of the experiments are not published in the report). Some sources suggested to use dummy encoding for the categorical values. We tried - this approach did not help either. It looks like the categorical features are not very well suited for clustering analysis (this is what **H** stats actually hinted).

Let's do another attempt this time we combine the number of victims and the number of gun used

```
victimStats = data %>% filter(n_victims >= 3 & n_guns_involved>0) %>% arrange(date)
stats =  victimStats %>% select(n_victims, n_guns_involved)

clara = clara(stats, k = 10, metric = "euclidean",stand = T, samples = 100, sampsize =500)
fviz_silhouette(clara, title = "") +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank())
```

```
   cluster size ave.sil.width
1        1   22          1.00
2        2   14          0.55
3        3   88          1.00
4        4   38          0.68
5        5    4          0.02
6        6   12          0.65
7        7    9          0.64
8        8  306          1.00
9        9    5          0.62
10      10    2          0.73
```

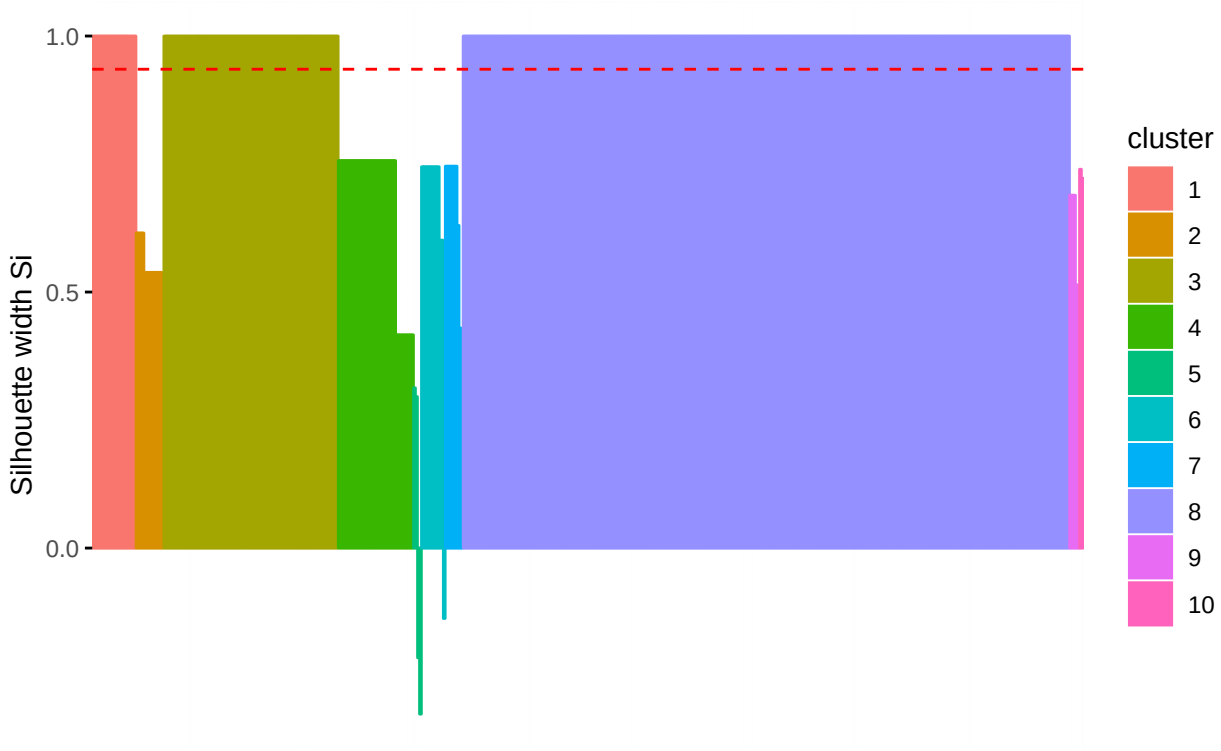

Figure 8: CLARA Approach. Two Continuous Feature, 10 Cluster Silhouette

We have played with the number of clusters and found the one, which produces the most optimal result. Ten cluster give quite balance split. The only exception is cluster # 4 which seems to contained misplaced observations. But if we employ different metrics would it help? Let us see. We use *manhattan* this time

```
clara = clara(stats, k = 10, metric = "manhattan",stand = T, samples = 100, sampsize =500)
fviz_silhouette(clara, title = "")
```

```
   cluster size ave.sil.width
1        1   27          1.00
2        2   17          0.33
3        3   94          1.00
4        4   35          0.70
5        5    3          0.45
6        6   11          0.53
7        7    9          0.67
8        8  295          1.00
9        9    7          0.43
10      10    2          0.63
```

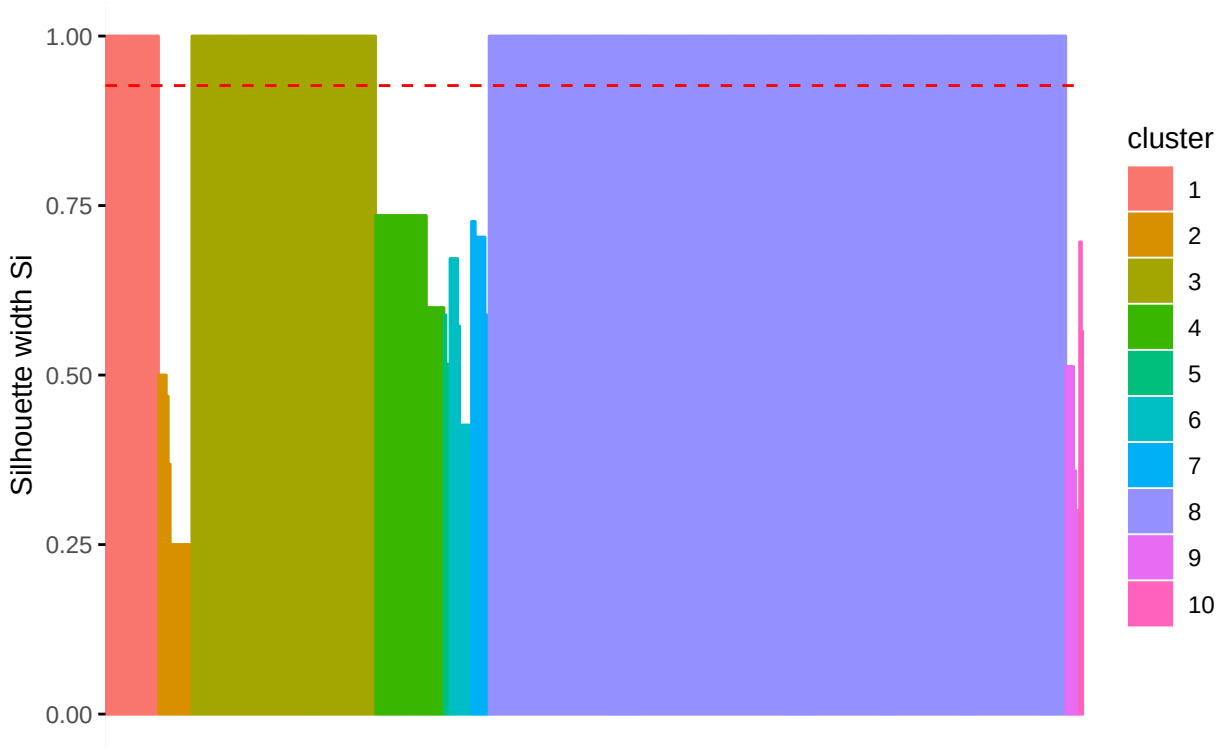

Figure 9: CLARA Approach. Two Continuous Feature, 10 Cluster Silhouette. Manhattan Metrics

```
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank())
```

```
List of 2
 $ axis.text.x : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ axis.ticks.x: list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 - attr(*, "class")= chr [1:2] "theme" "gg"
 - attr(*, "complete")= logi FALSE
 - attr(*, "validate")= logi TRUE
```

Voilà! It worked.

```
fviz_cluster(clara, stats, stand = F,  geom = "point", main="",
   axes = c(1,2), xlab = "N of Victims", ylab = "Number of Guns")
```

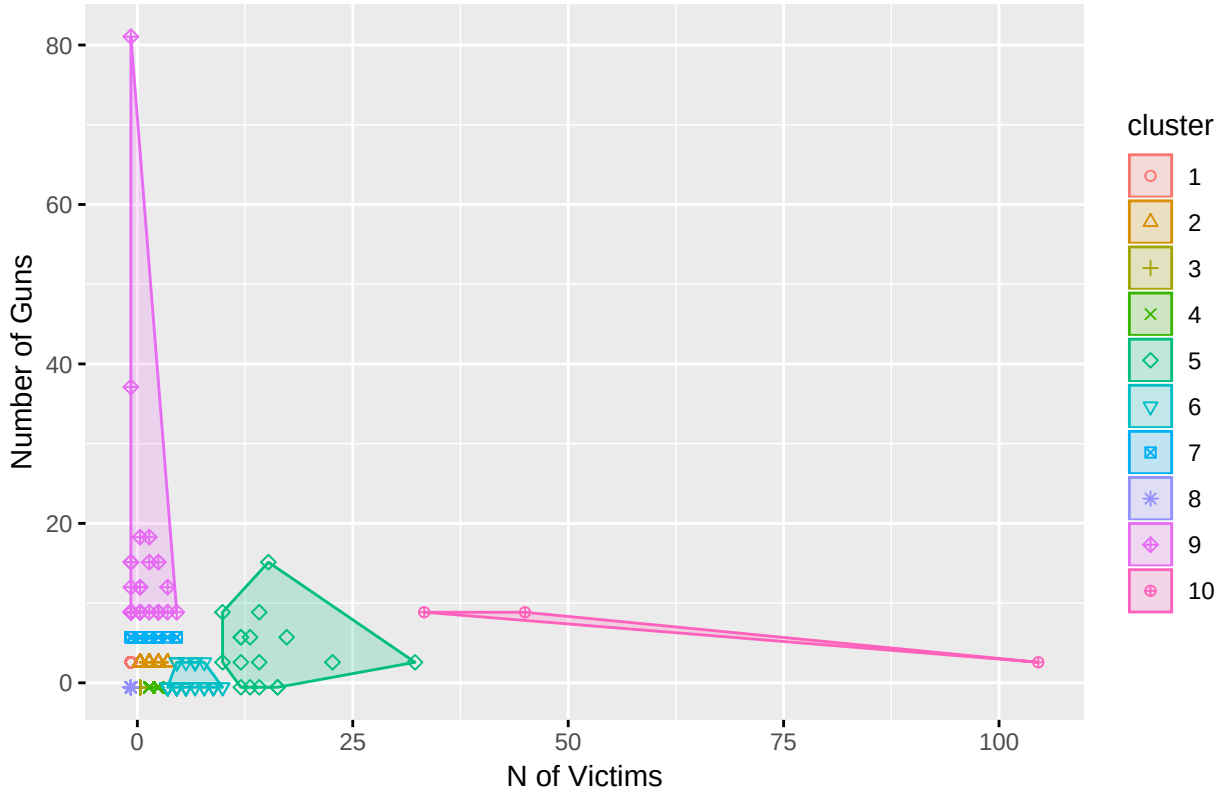

Figure 10: CLARA Approach. Two Continuous Feature, 10 Cluster Plot. Manhattan Metrics

Geo-location of the clasters.

```
combined = cbind(victimStats, cluster=c(clara$clustering))
ggplot(victimStats, aes(x=longitude, y=latitude )) +
  geom_point( color = clara$clustering+2L)
```

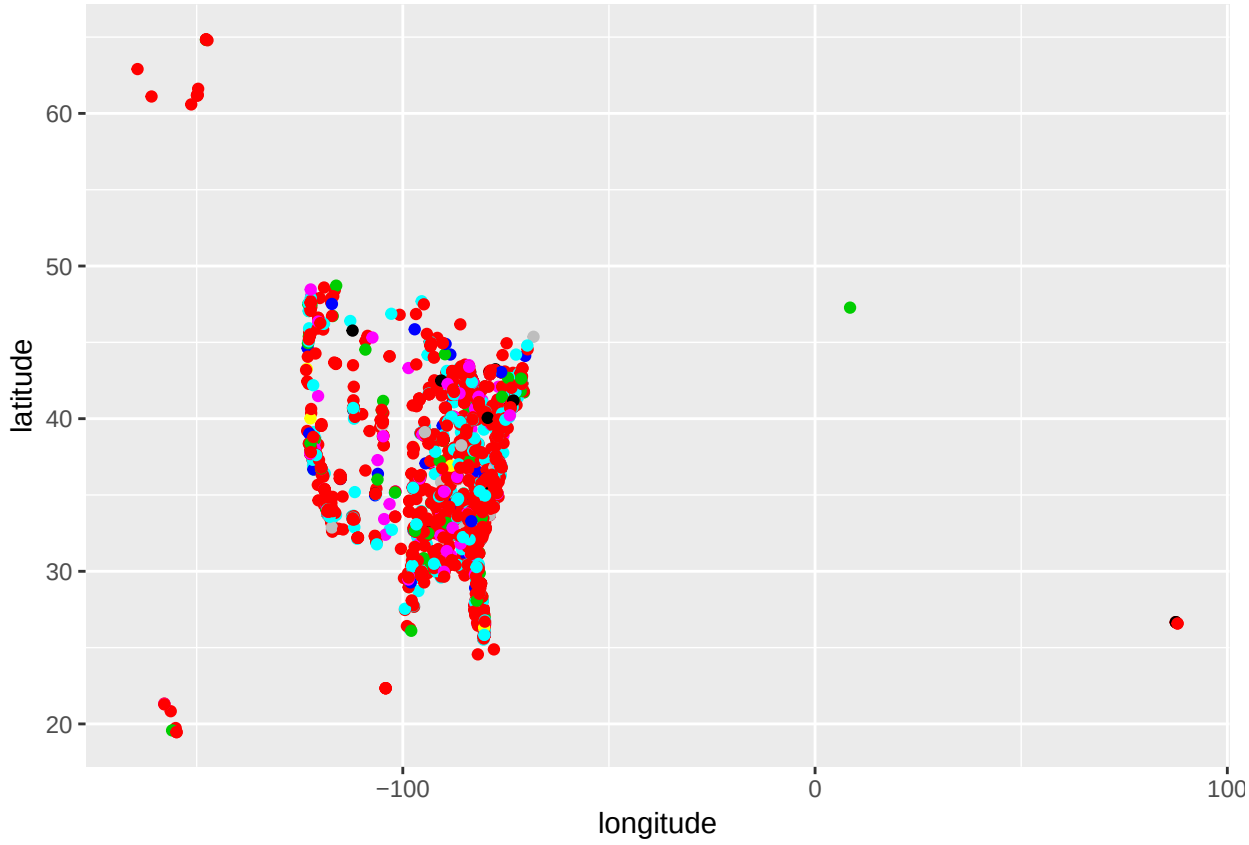

Figure 11: CLARA Approach. Two Continuous Feature, 10 Clusters. Manhattan Metrics. Number of Victims and Guns Clusters on Map

So how do we interprete the clusters?

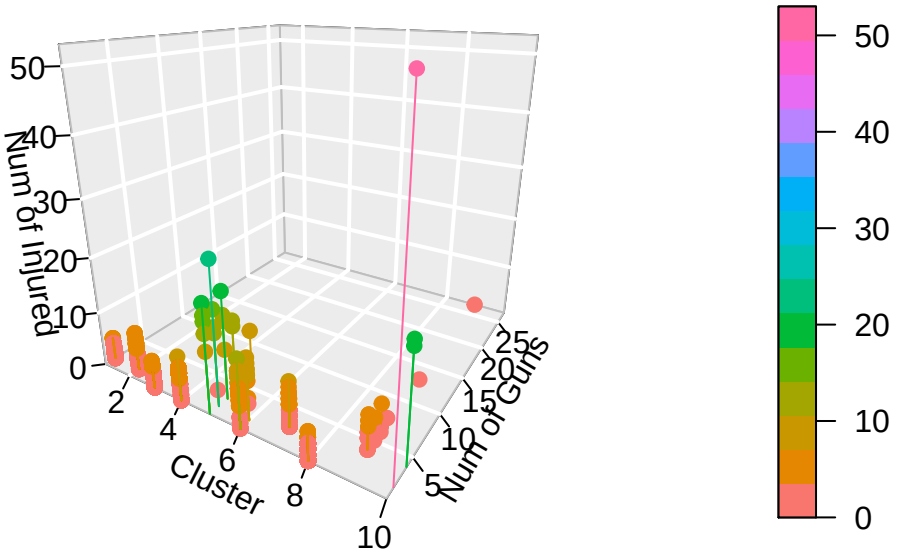

Figure 12: CLARA Approach. Number of Victims and Guns Cluster 3D Plot

Figure 12 shows that the hedious crimes that number dozens of people injured are committed with greater number of the guns (cluster # 10). Another extreme is when relatively low number of people is hurt but the number of guns found in the crime scene is extremely high. This could be attributed to a police raid.

Agglomerative Hierarchical Clustering Approach (AGNES).

Next we are going to examine the same feature combinations using agglomerative clustering. We decided to choose the agglomerative clustering (AGNES) vs devisive method because the former generally has less challanges than the latter. In the case of the devisive method it is not always clear how to partition a large cluster into a smaller one. The agglomerative method start with the bottom and increase the cluster size based on the selected metrics.

As in the case of *CLARA* we start with the univariate feature set, employing *euclidean* metrics, cutting the tree at 4 clusters

```
  cluster size ave.sil.width
1       1 2307          0.80
2       2  200          0.61
3       3   16          0.33
4       4    1          0.00
```

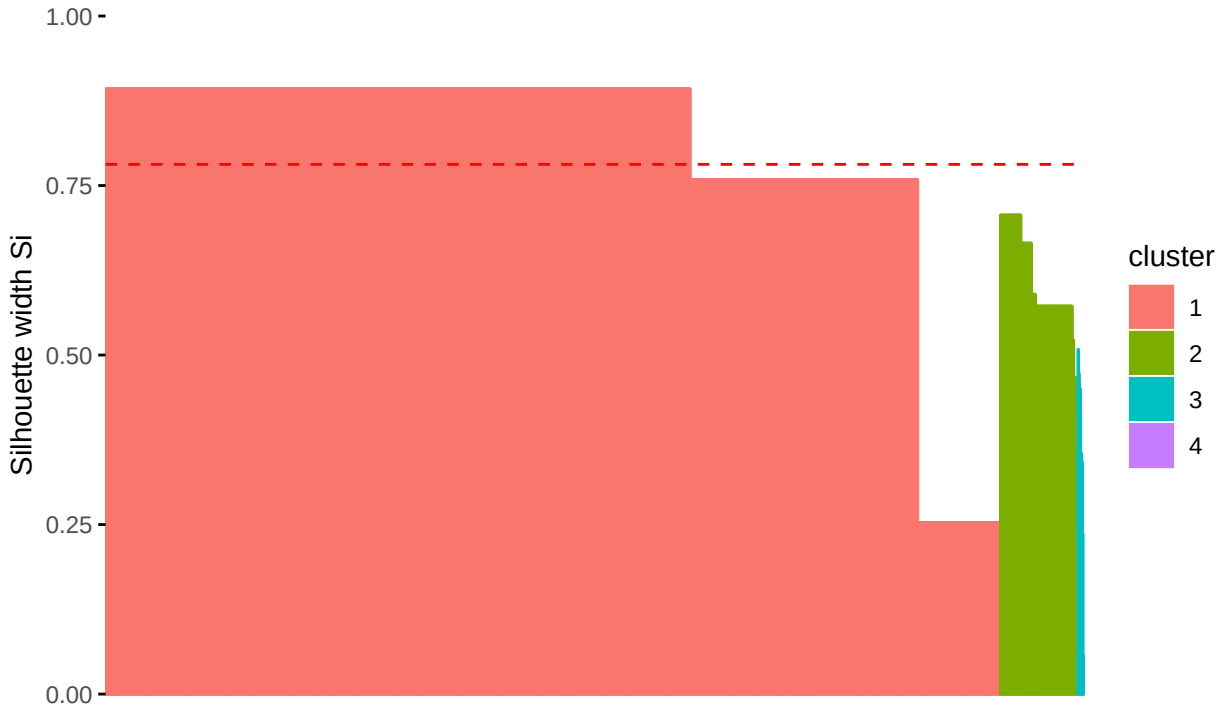

Figure 13: AGNES Approach. 4 Cluster, Univariate Feature Silhouette

Following our routine we review the silhouette coeffisietns. They are less satisfactory than the ones produced by *CLARA*. Further experimentations with the number of the clusters and distamce metrics did yield better result. So we are moving on to the nex feature set: *n_victims* and *n_guns_involved*

```
  cluster size ave.sil.width
1       1  375          0.17
2       2 2850          0.87
3       3    1          0.00
4       4    1          0.00
```

Figure 14: AGNES Approach. 4 Cluster, Two Feature Silhouette

Just like the previous attempt *AGNES* failed to provide any meaningful clusters.

Density-based Clustering Approach (DBSCAN)

Unlike the hierarchical and parallel methods the densicyt-based approach can deal with the clustered data of any shape. The dencity-based method do not require specification of number of clusters either. It is quite interesting to see if *DBSCAN* would be able to discover data patterens overlooked byt the previous two approaches. In the case of **DBSCAN** we need at least two-dimentinal data. Thus we will again be using *n_victims* and *n_guns_involved*

Priot to apllyng the algorithm we have to find the hyper parametrs of the *DBSCAN*, namely:

- the minimum distance between the point (**eps**).
- the minimum number of points to form a dense region (**minPoints**).

For that we will employ *k nearest neighbors* method. The idea is to calculate, the average of the distances of every point to its k nearest neighbors. The value of k will be specified by us and corresponds to **minPoints**. We will start with *k* to 5

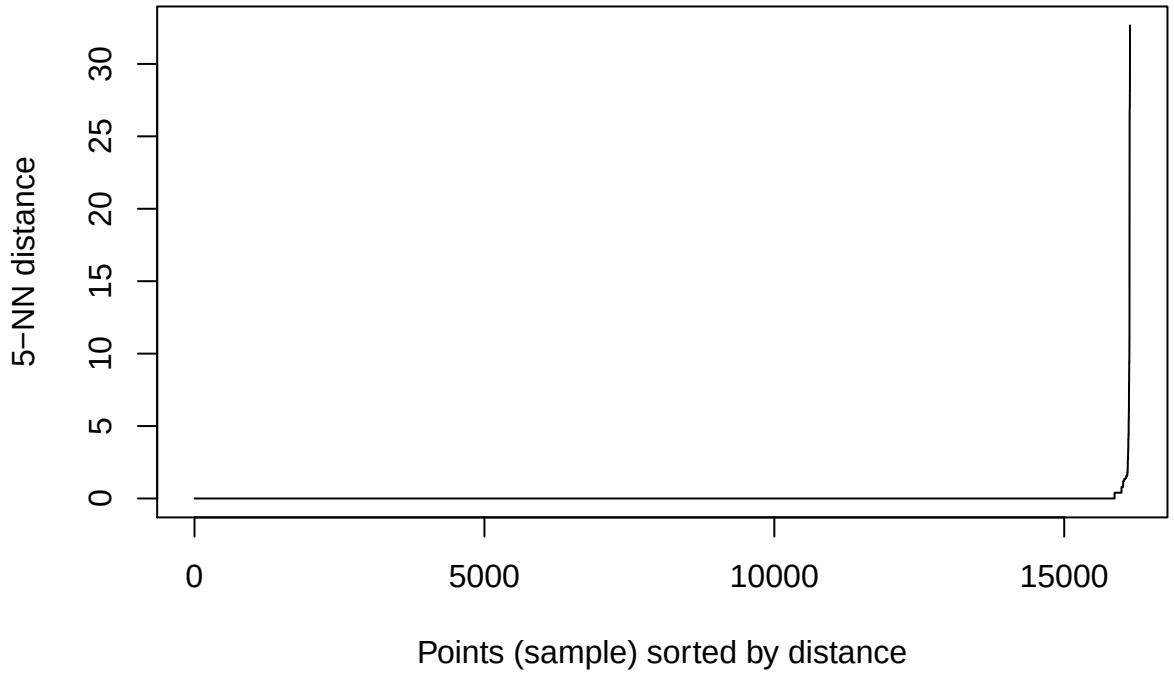

Figure 15: DBSCAN Approach. Finding Hyper-Parameters

The plot (Figure: **??**) has quite sharp elbow. It appers that **eps** = 1.5 would be a good starting value for 5 minimum points. Let's run dbscan method

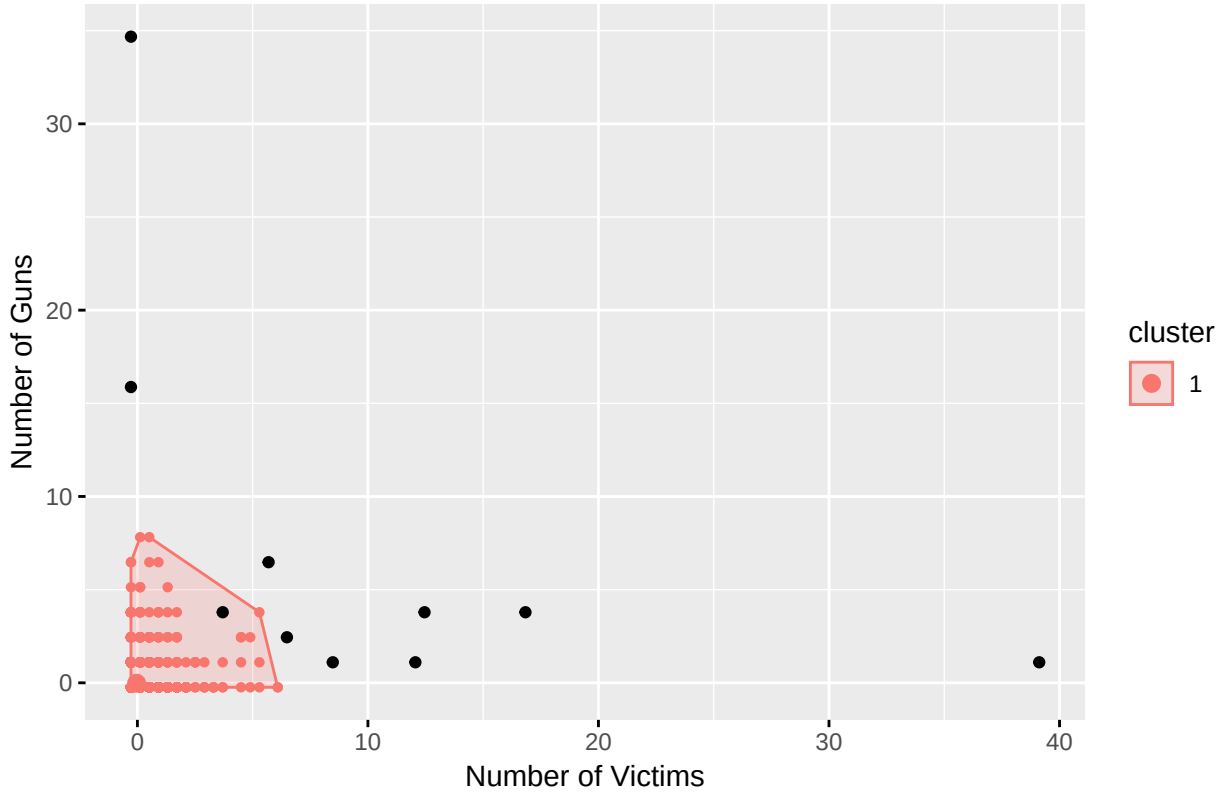

Figure 16: DBSCAN Approach. Clusters

The result of the first attempt is not satisfactory. The method identified one cluster and lots outliers. Let add geo-coordinates. Spatial data should introduce more dense data, which hopefully will be picked by the algorithm

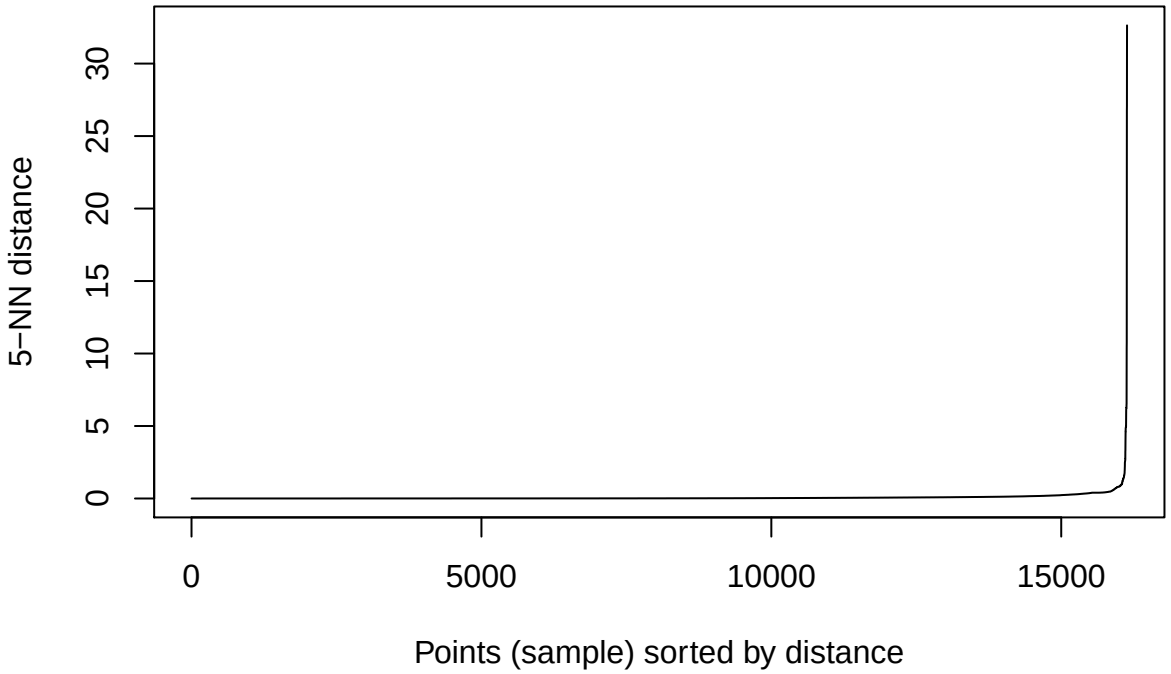

Figure 17: DBSCAN Approach. Finding Hyper-Parameters for High-dimentional Dataset

As per Figure 17 we choose **1** for *eps*

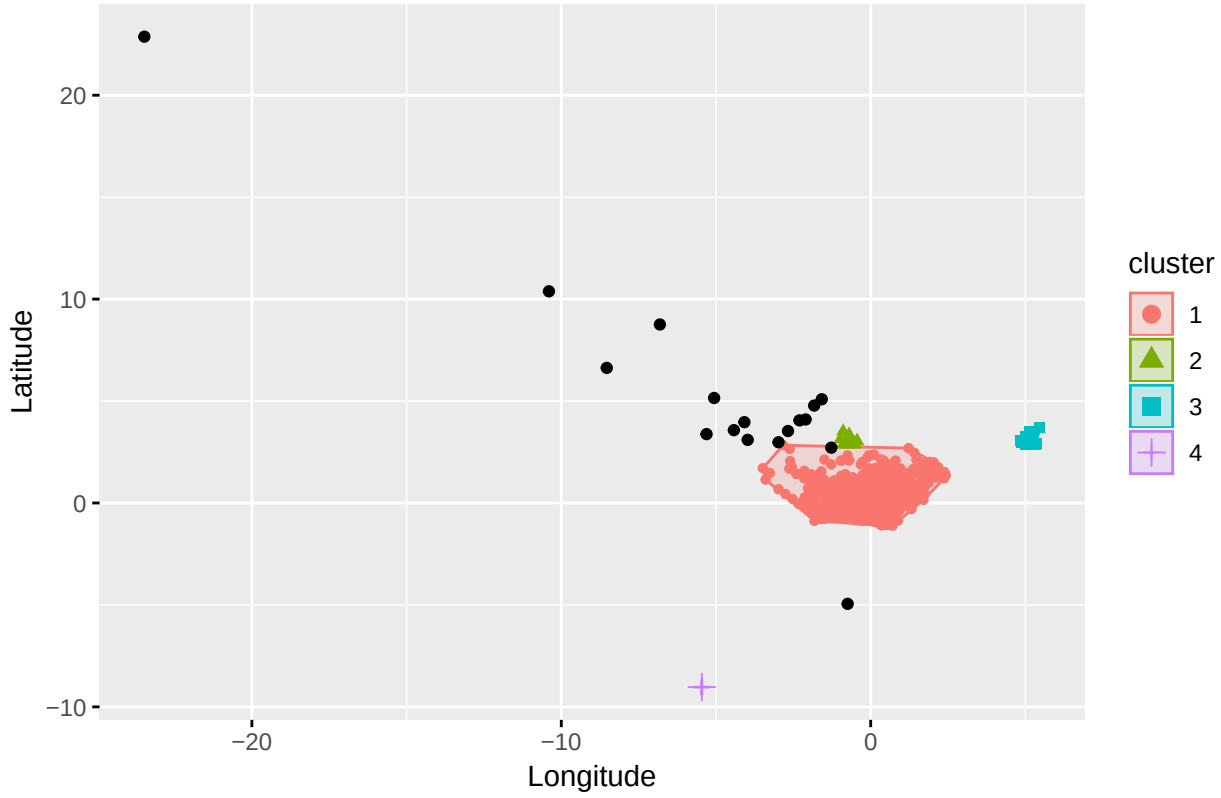

Figure 18: DBSCAN Approach. Clusters

This approach has identified four clusters of violence. Let's view how the incidents are spread geographically

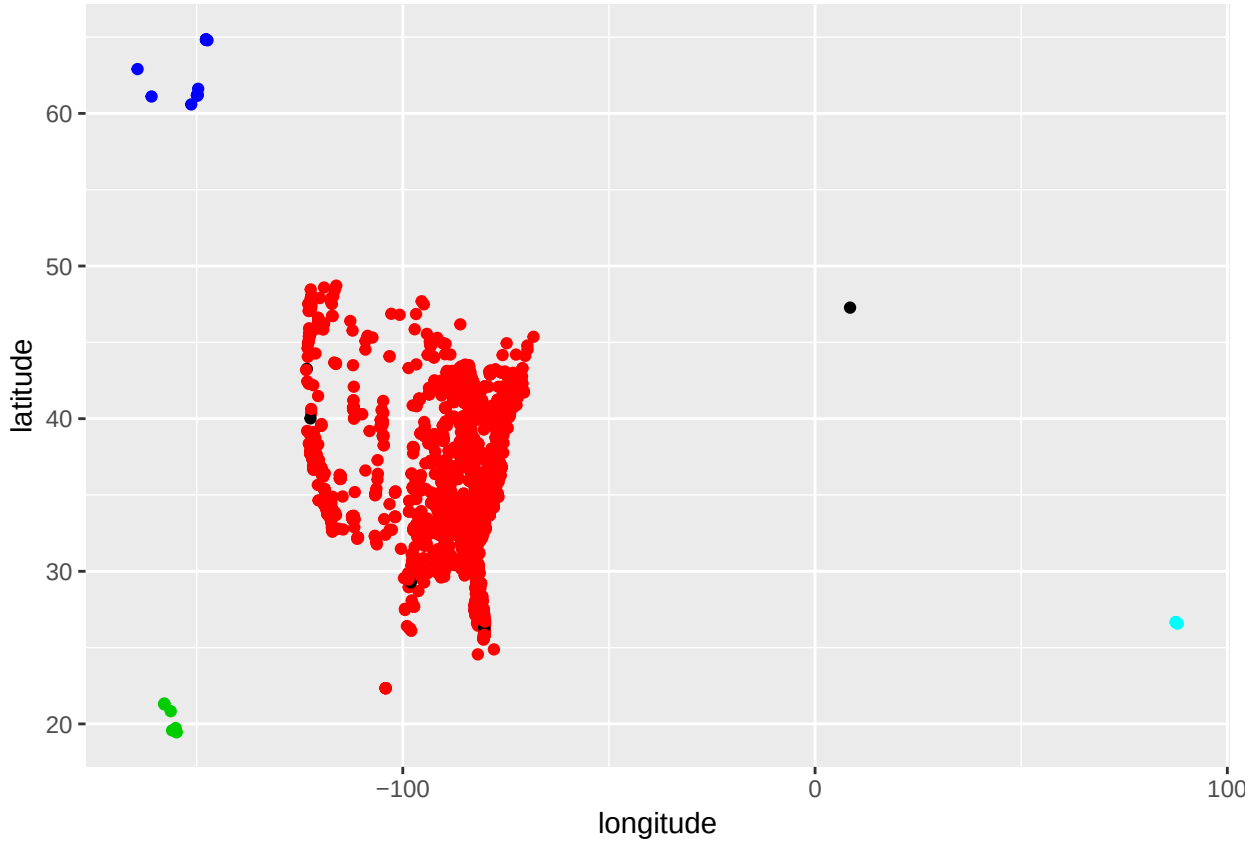

Figure 19: DBSCAN Approach. Victims of Gun Violence on the Map

Clustering Method Evaluation

Model Deployment

Conclusion

Note from the Authors

This file was generated using *The R Journal* style article template, additional information on how to prepare articles for submission is here - Instructions for Authors. The article itself is an executable R Markdown file that could be downloaded from Github with all the necessary artifacts.

Sumaira Afzal
York University School of Continuing Studies

Viraja Ketkar
York University School of Continuing Studies

Murlidhar Loka
York University School of Continuing Studies

Vadim Spirkov
York University School of Continuing Studies