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

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

Abstract Gun deaths in US rise to highest level in 20 years. Forty thousand people were killed in shootings in 2017 amid a growing number of suicides involving firearms. Research by the Educational Fund to Stop Gun Violence underlines that the tragedy of gun violence and suicides is not spread randomly across the country, but is concentrated precisely in those places where gun ownership is most prevalent and gun laws at their loosest. Using the data collected by Gun Violence Archive (GVA) and employing unsupervised learning methods we will make an attempt to provide additional insights into the nature of this sad statistics.

Background

Gun violence in the US is a going problem. Mass-shooting, gun-assited suisides, accidental use of guns that cause death have become regular topic of the news agencies.

Objective

The objective of this research is to discover trends and tendencies of the gun violence situation in the US employing unsupervised learning algorithms.

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

Column Name	Column Description
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)
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 2: Gun Violence Dataset Summary

No	Variable	Stats / Values	Freqs (% of Valid)	Missing
1	incident_id [integer]	Mean (sd) : 559334.3 (293128.7) min < med < max: 92114 < 543587 < 1083472 IQR (CV) : 508683 (0.5)	239677 distinct values	0 (0%)
2	date [factor]	1. 2013-01-01 2. 2013-01-05 3. 2013-01-07 [1722 others]	3 (0.0%) 1 (0.0%) 2 (0.0%) 239671 (100.0%)	0 (0%)
3	state [factor]	1. Alabama 2. Alaska 3. Arizona [48 others]	5471 (2.3%) 1349 (0.6%) 2328 (1.0%) 230529 (96.2%)	0 (0%)
4	city_or_county [factor]	1. Abbeville 2. Abbotsford 3. Abbott [12895 others]	37 (0.0%) 3 (0.0%) 1 (0.0%) 239636 (100.0%)	0 (0%)
5	address [factor]	1. 100 block of Kohler Cour 2. 100 block of South Sumne 3. 1000 block of 32nd St [198034 others]	1 (0.0%) 1 (0.0%) 1 (0.0%) 223177 (100.0%)	16497 (6.88%)
6	n_killed [integer]	Mean (sd) : 0.3 (0.5) min < med < max: 0 < 0 < 50 IQR (CV) : 0 (2.1)	16 distinct values	0 (0%)
7	n_injured [integer]	Mean (sd) : 0.5 (0.7) min < med < max: 0 < 0 < 53 IQR (CV) : 1 (1.5)	23 distinct values	0 (0%)

No	Variable	Stats / Values	Freqs (% of Valid)	Missing
8	incident_url [factor]	1. http: //www.gunviolencearc/ 2. http: //www.gunviolencearc/ 3. http: //www.gunviolencearc/ [239674 others]	1 (0.0%) 1 (0.0%) 1 (0.0%) 239674 (100.0%)	0 (0%)
9	source_url [factor]	1. /%20—8newsnow.com 2. /%20%20http%3A//www.wdsu 3. /%20http%3A//blog.al.com/ [213986 others]	1 (0.0%) 1 (0.0%) 1 (0.0%) 239206 (100.0%)	468 (0.2%)
10	incident_url_fields_missing [factor]	1. False	239677 (100.0%)	0 (0%)
11	congressional_district [integer]	Mean (sd) : 8 (8.5) min < med < max: 0 < 5 < 53 IQR (CV) : 8 (1.1)	54 distinct values	11944 (4.98%)
12	gun_stolen [factor]	1. 0::Not-stolen 2. 0::Not-stolen | | 1::Not-sto 3. 0::Not-stolen | | 1::Not-sto [346 others]	1352 (1.0%) 45 (0.0%) 10 (0.0%) 138772 (99.0%)	99498 (41.51%)
13	gun_type [factor]	1. 0::10mm 2. 0::10mm | | 1::22 LR 3. 0::10mm | | 1::223 Rem [AR-1 [2499 others]	32 (0.0%) 1 (0.0%) 1 (0.0%) 140192 (100.0%)	99451 (41.49%)
14	incident_characteristics [factor]	1. Accidental Shooting 2. Accidental Shooting | | Acci 3. Accidental Shooting | | Acci [18123 others]	1 (0.0%) 20 (0.0%) 8 (0.0%) 239322 (100.0%)	326 (0.14%)
15	latitude [numeric]	Mean (sd) : 37.5 (5.1) min < med < max: 19.1 < 38.6 < 71.3 IQR (CV) : 7.5 (0.1)	101240 distinct values	7923 (3.31%)
16	location_description [factor]	1. 'Taste' Dessert Bar 2. "Anderson Island" 3. "Canadian Shores" [27592 others]	1 (0.0%) 1 (0.0%) 1 (0.0%) 42086 (100.0%)	197588 (82.44%)
17	longitude [numeric]	Mean (sd) : -89.3 (14.4) min < med < max: -171.4 < -86.2 < 97.4 IQR (CV) : 14.1 (-0.2)	112347 distinct values	7923 (3.31%)
18	n_guns_involved [integer]	Mean (sd) : 1.4 (4.7) min < med < max: 1 < 1 < 400 IQR (CV) : 0 (3.4)	106 distinct values	99451 (41.49%)
19	notes [factor]	1. ' When asked what was goi 2. 'heard shots, felt pain' 3. 'heard shots, felt pain.' [136649 others]	1 (0.0%) 1 (0.0%) 1 (0.0%) 158657 (100.0%)	81017 (33.8%)
20	participant_age [factor]	1. 0::0 2. 0::0 | | 1::1 | | 2::28 | | 3::24 3. 0::0 | | 1::18 [18948 others]	12 (0.0%) 1 (0.0%) 2 (0.0%) 147364 (100.0%)	92298 (38.51%)
21	participant_age_group [factor]	1. 0::Adult 18+ 2. 0::Adult 18+ | | 1::Adult 18 3. 0::Adult 18+ | | 1::Adult 18 [895 others]	94671 (47.9%) 49273 (24.9%) 13893 (7.0%) 39721 (20.1%)	42119 (17.57%)

No	Variable	Stats / Values	Freqs (% of Valid)	Missing
22	participant_gender [factor]	1. 0::Female 2. 0::Female ||1::Female 3. 0::Female ||1::Female ||2:: [870 others]	7791 (3.8%) 918 (0.5%) 102 (0.1%) 194504 (95.7%)	36362 (15.17%)
23	participant_name [factor]	1. 0::"Bear" ||1::Cardell Mon 2. 0::"Bo" 3. 0::"Chicago" [113485 others]	1 (0.0%) 1 (0.0%) 1 (0.0%) 117421 (100.0%)	122253 (51.01%)
24	participant_relationship [factor]	1. 0::Aquaintance 2. 0::Aquaintance ||1::Aquain 3. 0::Aquaintance ||1::Aquain [281 others]	63 (0.4%) 8 (0.1%) 1 (0.0%) 15702 (99.5%)	223903 (93.42%)
25	participant_status [factor]	1. 0::Arrested 2. 0::Arrested ||1::Arrested 3. 0::Arrested ||1::Arrested | [2147 others]	2739 (1.3%) 487 (0.2%) 175 (0.1%) 208650 (98.4%)	27626 (11.53%)
26	participant_type [factor]	1. 0::Subject-Suspect 2. 0::Subject-Suspect ||1::Su 3. 0::Subject-Suspect ||1::Su [256 others]	44914 (20.9%) 8922 (4.2%) 3040 (1.4%) 157938 (73.5%)	24863 (10.37%)
27	sources [factor]	1. http://1160wccs.com/blair/ 2. http://1160wccs.com/polic/ 3. http://1160wccs.com/polic/ [217277 others]	1 (0.0%) 1 (0.0%) 1 (0.0%) 239065 (100.0%)	609 (0.25%)
28	state_house_district [integer]	Mean (sd) : 55.4 (42) min < med < max: 1 < 47 < 901 IQR (CV) : 63 (0.8)	275 distinct values	38772 (16.18%)
29	state_senate_district [integer]	Mean (sd) : 20.5 (14.2) min < med < max: 1 < 19 < 94 IQR (CV) : 21 (0.7)	68 distinct values	32335 (13.49%)

Initial observation of the data shows that there is a number of features which do not present any analytical value (Table: 2). They are:

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

We are going to get rid of them. We will also 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: 2). This is not quite acceptable for the analytic, 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. To do so we will employ text mining technique (Ref: Ingo Feinerer). We will analyze term frequencies to make conclusions about the content of the features.

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)
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 related 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: 2) *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.

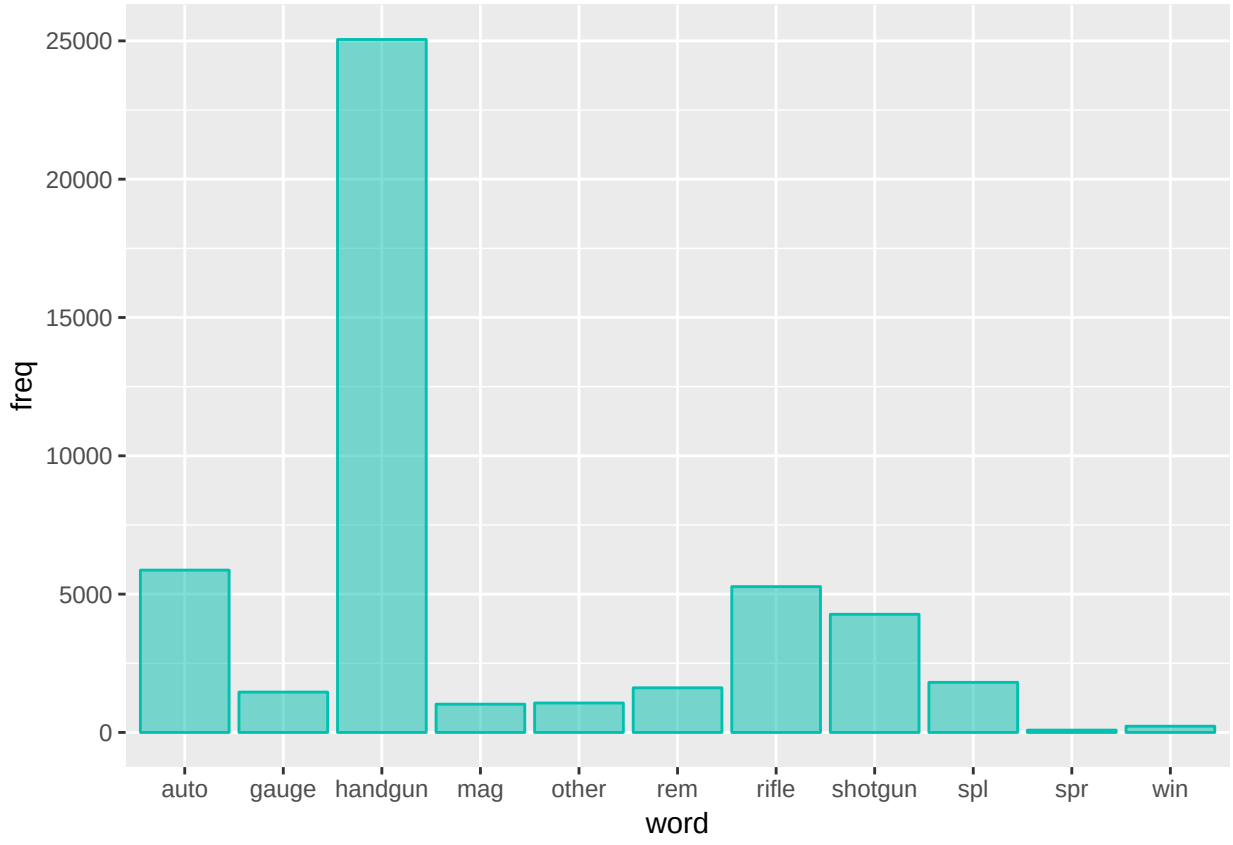

Figure 1: The Most Frequently Used Gun Types

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.

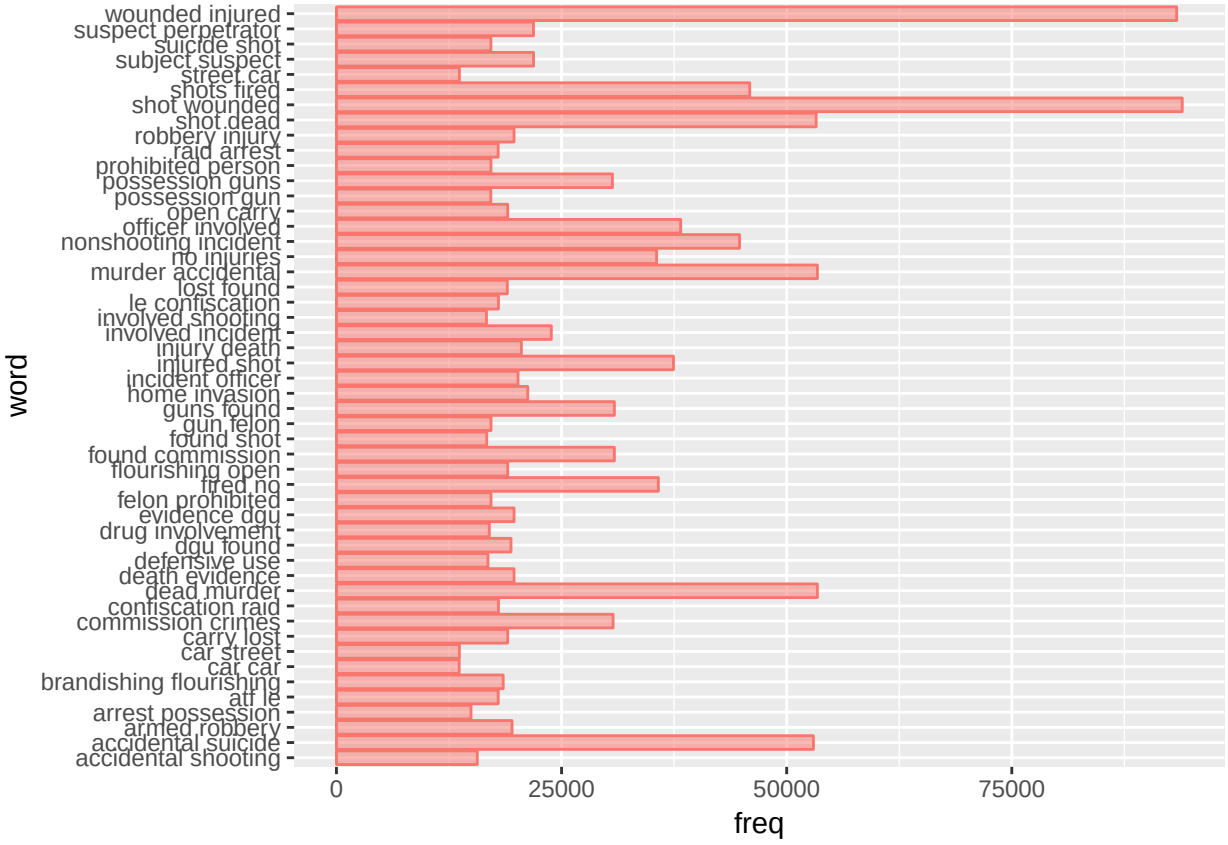

Figure 4: Most Common Bi-gram Terms in Incident Characteristics Column

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 the 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 finally have reached the stage when our dataset is ready to be used for exploration by clustering algorithms. All missing features have been imputed and free-format text columns have been replaced with the categorical attributes. This is the summary of the resulting data.

```
print(dfSummary(data, valid.col = F, max.distinct.values = 3, headings = F),
      caption = "\\tt Engineered Gun Violence Dataset Summary", scalebox = .9)
```

Table 4: `Engineered Gun Violence Dataset Summary`

No	Variable	Stats / Values	Freqs (% of Valid)	Missing
1	date [factor]	1. 2013-01-01 2. 2013-01-05 3. 2013-01-07 [1722 others]	3 (0.0%) 1 (0.0%) 2 (0.0%) 239671 (100.0%)	0 (0%)
2	state [factor]	1. Alabama 2. Alaska 3. Arizona [48 others]	5471 (2.3%) 1349 (0.6%) 2328 (1.0%) 230529 (96.2%)	0 (0%)
3	city_or_county [factor]	1. Abbeville 2. Abbotsford 3. Abbott [12895 others]	37 (0.0%) 3 (0.0%) 1 (0.0%) 239636 (100.0%)	0 (0%)
4	n_killed [integer]	Mean (sd) : 0.3 (0.5) min < med < max: 0 < 0 < 50 IQR (CV) : 0 (2.1)	16 distinct values	0 (0%)
5	n_injured [integer]	Mean (sd) : 0.5 (0.7) min < med < max: 0 < 0 < 53 IQR (CV) : 1 (1.5)	23 distinct values	0 (0%)
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%)

No	Variable	Stats / Values	Freqs (% of Valid)	Missing
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%)
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. The focus of our interests will be 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 calculate 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(Ref: Alboukadel Kassambara).

```
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
```

Outstanding! **H** value is very close to 0. Let's 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 suitable 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 features to get dense, well separated clusters. Now it is time to explore the data

Partitioning Clustering Approach Using CLARA

We decided to select CLARA method because it scales well and can deal with continuous 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 simplicity. Then we will take a look at silhouette plot (Ref: Alboukadel Kassambara) to analyze the result. The silhouette coefficient measure how well the clusters are separated. The silhouette analysis also provides insights into the cluster density.

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

```
  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
```

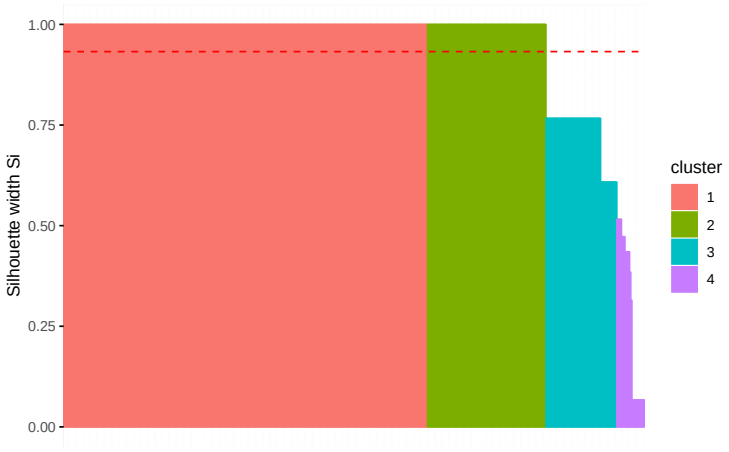

Figure 5: CLARA Approach. Univariate Feature, 4 Cluster Silhouette

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

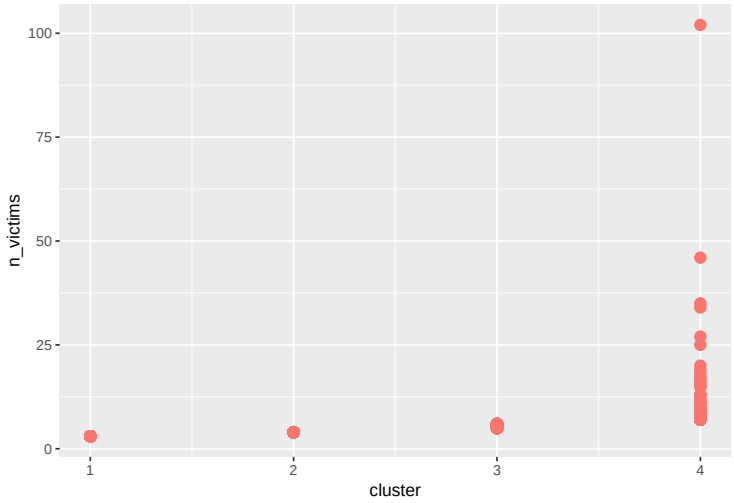

Figure 6: CLARA Approach. Number of Victims/Cluster Scatter Plot

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 geo-location of the clustered incidents

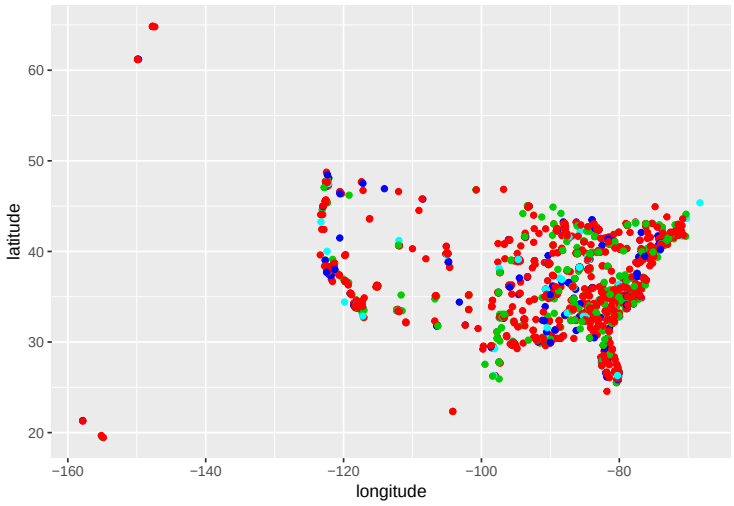

Figure 7: CLARA Approach. Number of Victims Clusters on Map

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

```
  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
```

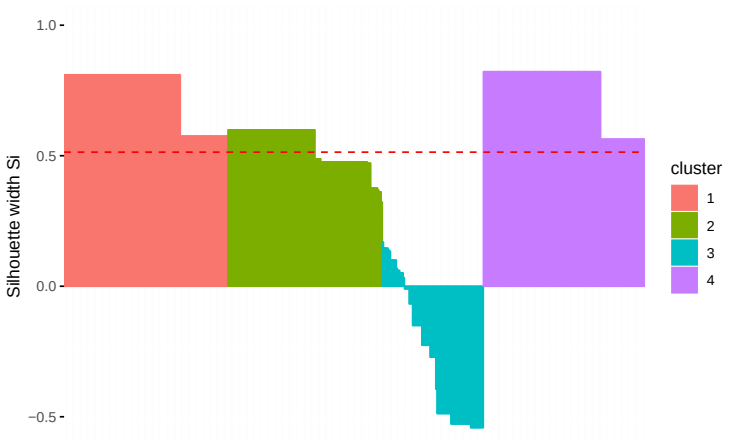

Figure 8: CLARA Approach. Two Feature, 4 Cluster Silhouette

Unfortunately the result is not great. Average silhouette coefficient is about 0.5; one cluster has a negative value. We have made quite a few experiments 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
```

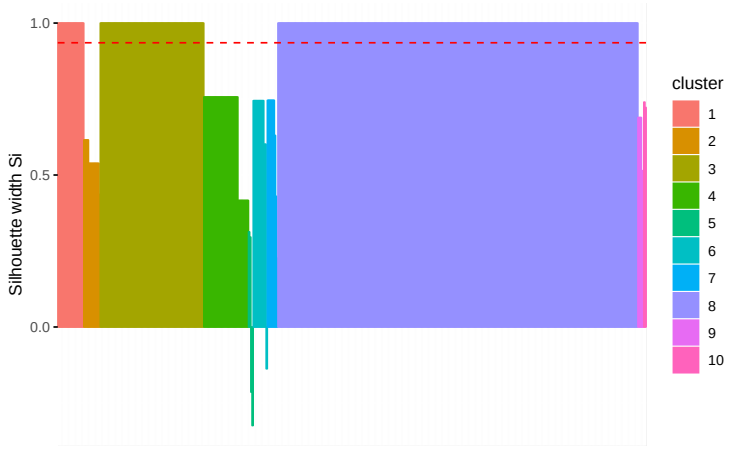

Figure 9: CLARA Approach. Two Continuous Feature, 10 Cluster Silhouette

We have played with the number of clusters and found the one, which produces the most optimal result. Ten cluster give quite balanced 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
```

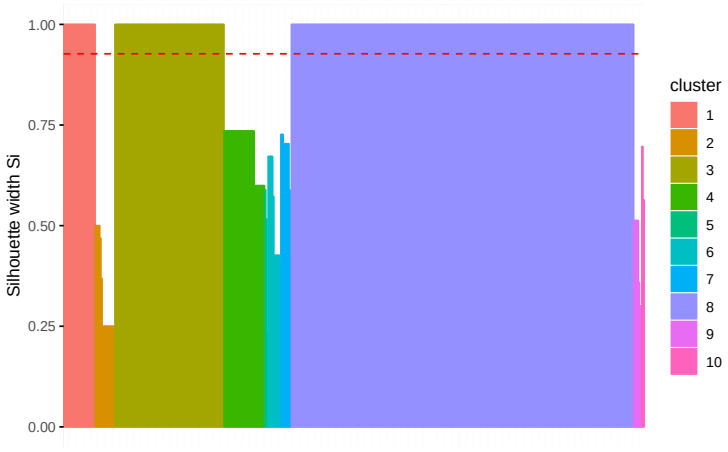

Figure 10: CLARA Approach. Two Continuous Feature, 10 Cluster Silhouette. Manhattan Metrics

```
  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")
```

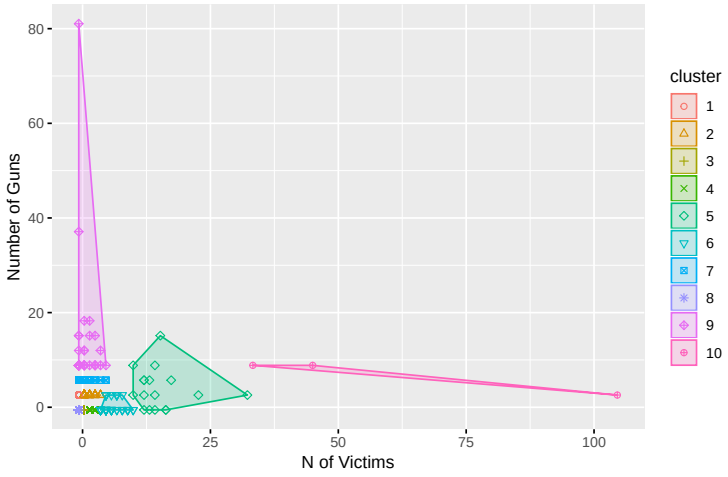

Figure 11: CLARA Approach. Two Continuous Feature, 10 Cluster Plot. Manhattan Metrics

Geo-location of the clusters.

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

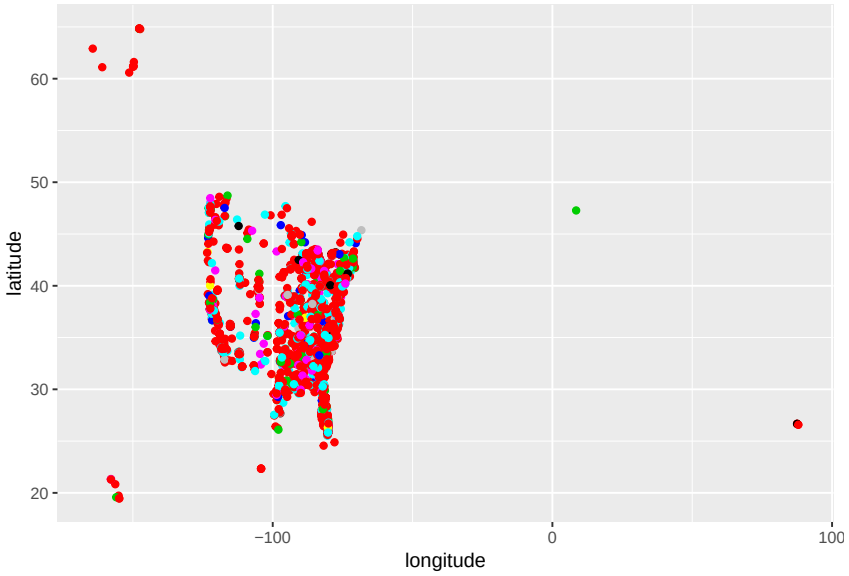

Figure 12: CLARA Approach. Two Continuous Feature, 10 Clusters. Manhattan Metrics. Number of Victims and Guns Clusters on Map

So how do we interpret the clusters?

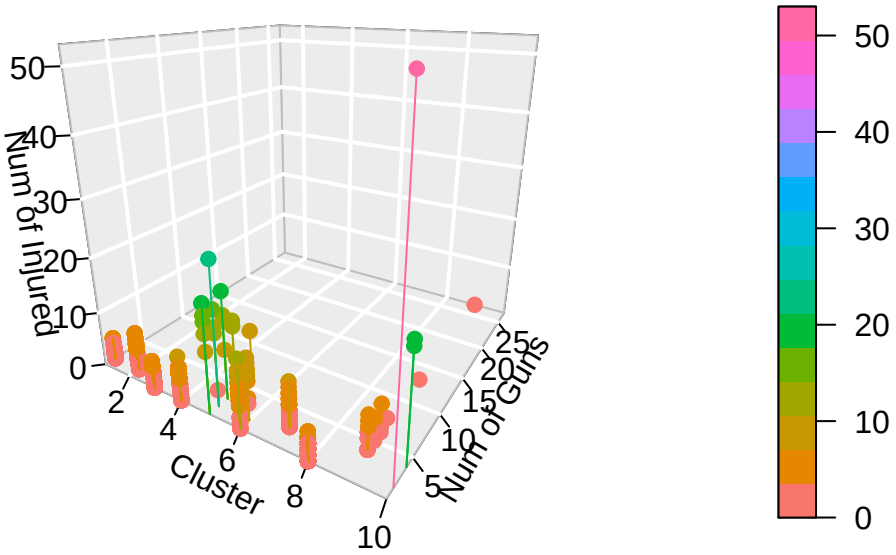

Figure 13: CLARA Approach. Number of Victims and Guns Cluster 3D Plot

Figure 13 shows that the hideous crimes that number dozens of the injured victims are committed with greater number of the guns (cluster # 10). Another extreme is when a 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.

What if try to cluster dataset that includes geo-coordinates and categorical feature? Let's select

observations that have number of victims killed greater or equal to 1, number of guns involved greater than zero and known place type.

```
victimStats = data %>% filter(n_victim_killed >= 1 & n_guns_involved>0 & place_type >0) %>% arrange(date)
stats =  victimStats %>% select(longitude, latitude, n_victims, place_type)

clara = clara(stats, k = 5, 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   41          0.09
2       2  138          0.45
3       3  161          0.22
4       4   74          0.27
5       5   86          0.41
```

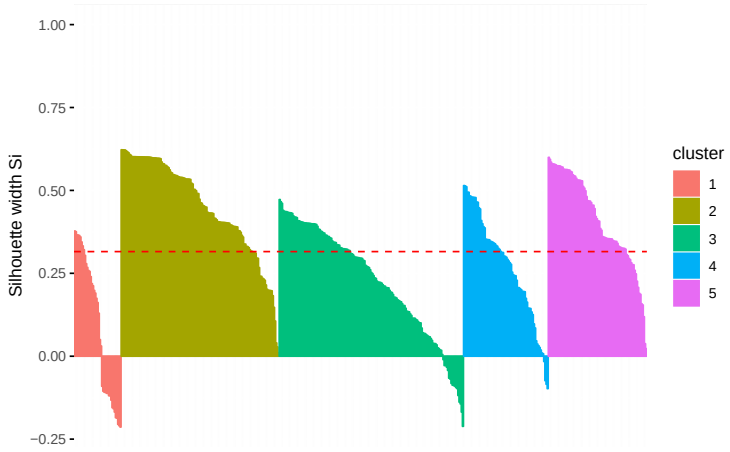

Figure 14: CLARA Approach. Continuous and Categorical Feature, 5 Cluster Silhouette

Again the parallel clustering method does not yield the satisfactory result.

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 divisive method because the former generally has less challenges than the latter. In the case of the divisive method it is not always clear how to partition a large cluster into a smaller one (Ref: Jiawei Han (2012)). The agglomerative method starts from the bottom and increases 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 15: AGNES Approach. 4 Cluster, Univariate Feature Silhouette

Following our routine we review the silhouette coefficients. They are less satisfactory than the ones produced by *CLARA*. Further experimentation with the number of the clusters and distance metrics did not produce better result. So we are moving on to the next 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 16: 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 density-based approach can deal with the clustered data of any shape. The density-based method do not require specification of number of clusters either. It is quite interesting to see if *DBSCAN* is able to discover data patterns overlooked by the previous two approaches (Ref: Michael Hahsler) . In the case of **DBSCAN** we need at least two-dimensional data. Thus we will again be using *n_victims* and *n_guns_involved*

Prior to applying the algorithm we have to find the hyper parameters 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

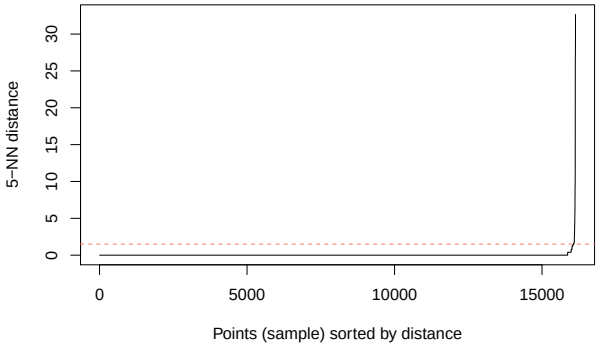

Figure 17: DBSCAN Approach. Finding Hyper-Parameters

```
integer(0)
```

The plot (Figure: 17) has quite sharp elbow. It appears that **eps** = 1.5 would be a good starting value for 5 minimum points. Let's run dbscan method

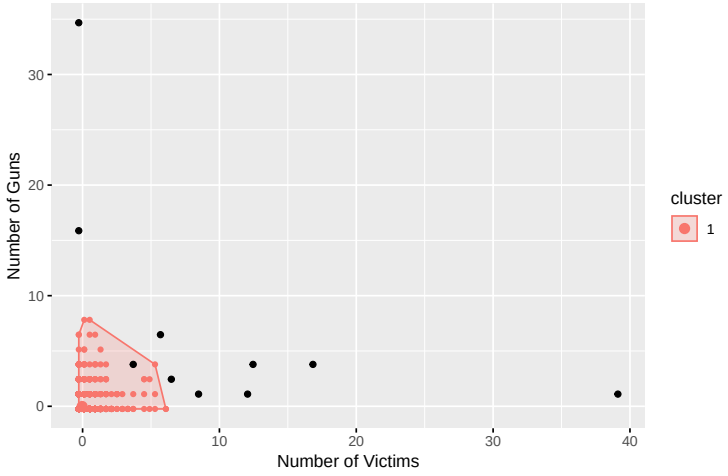

Figure 18: DBSCAN Approach. Clusters

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

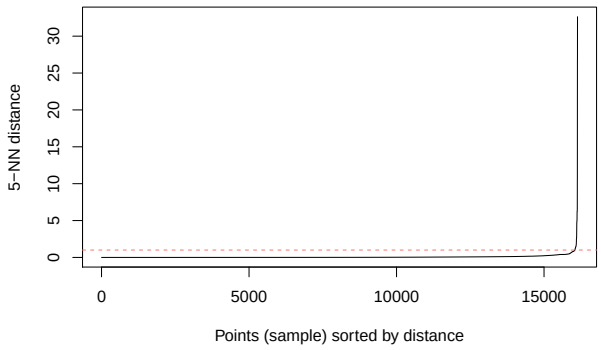

Figure 19: DBSCAN Approach. Finding Hyper-Parameters for High-dimentional Dataset

```
integer(0)
```

As per Figure 19 we choose **1** for *eps*

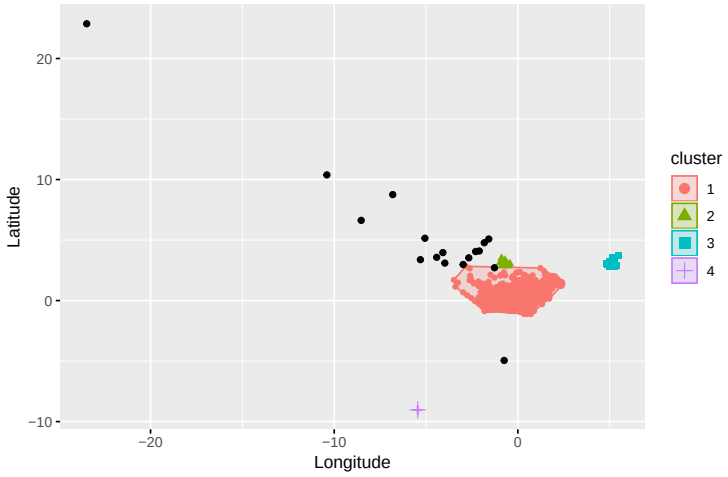

Figure 20: DBSCAN Approach. Clusters

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

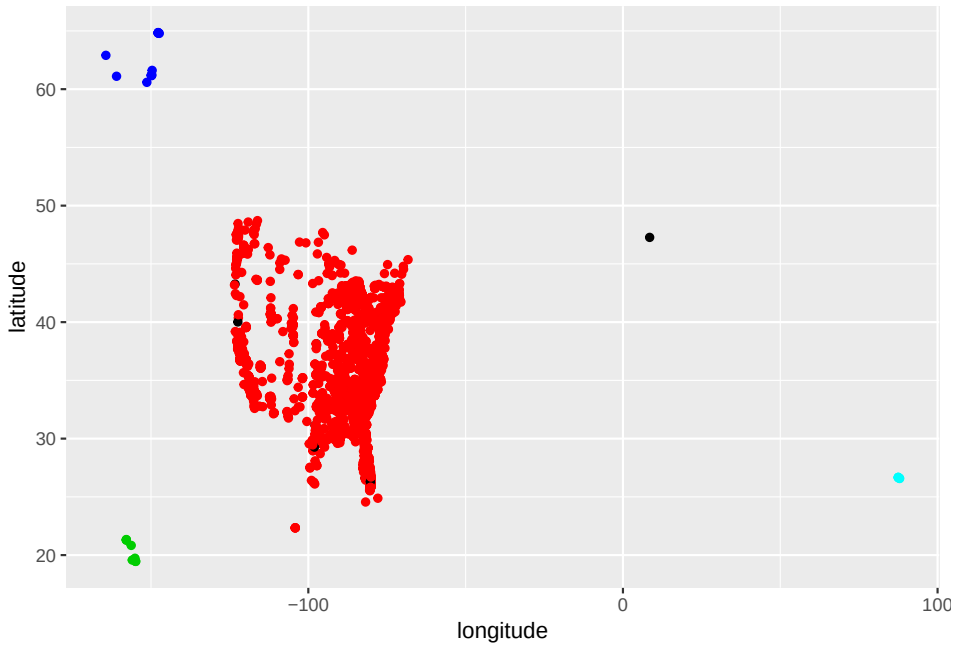

Figure 21: DBSCAN Approach. Victims of Gun Violence on the Map

Lastly we are going to increase dimentionality of the dataset adding **place_type** categorical feature. So we will be clustering data set that have geo-coordinates, number of victims and a place type.

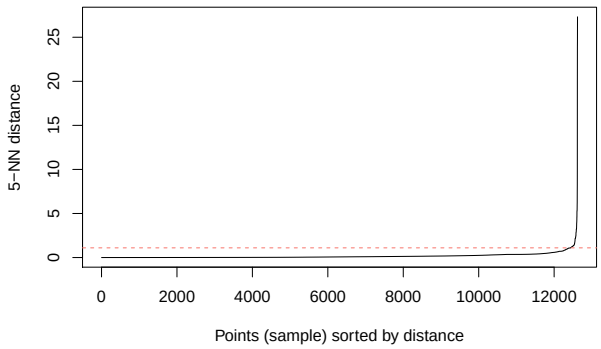

Figure 22: DBSCAN Approach. Finding Hyper-Parameters for High-dimentional Dataset with Categorical Feature

```
integer(0)
```

We will be using *eps* = 1.1

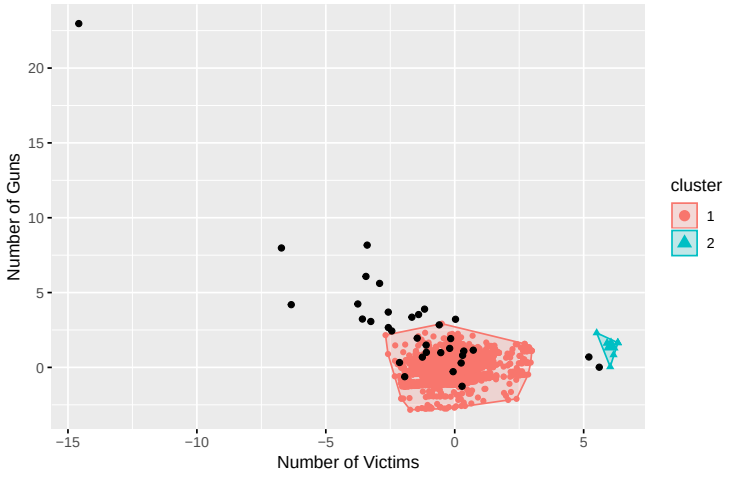

Figure 23: DBSCAN Approach. Clusters with Categorical features

Let's review the cluster content.

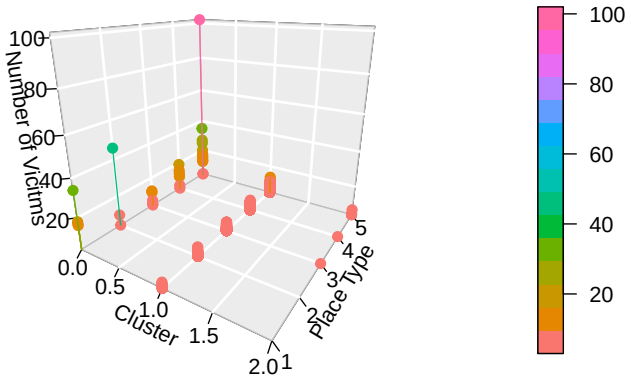

Figure 24: DBSCAN Approach. Number of Victims and lace Type Cluster 3D Plot

As per Figure 24 the latest **DBSCAN* run grouped the data geographically. It did not find *place type* and *number of victims* significant enough to form clusters based on those two features.

Clustering Method Evaluation

We have applied three different clustering algorithm. The parallel clustering method **CLARA** performed the best on both univariate and multi-dimensional datasets. Tweaking the number of clusters and distance metrics we were able to find meaningful clusters with good separation and density. (see Figure 5 and 10). This approach did fail to cluster categorical data.

Neither hierarchical nor density-based approach performed particularly well. *DBSCAN* produced slightly better result when we added geo-coordinates, but the result was pretty meaningless. The density-based approach grouped data by longitude and latitude drawing the map of the USA…

Model Deployment

Unfortunately we can state that the clustering methods were not effective for the selected dataset. Apparently hat frequency based analysis would yield much better result.

Conclusion

We selected **Gun Violence in the US** dataset hoping to discover new relationship between various attributes, which would provide new insights into the goring problem of gun violence in the states.

We spent significant efforts parsing and cleaning the data. We started with imputing missing geographic coordinates employing *OpenCage* geocoding service. Then we separated redundant and useful features. Majority of the useful features contained delimiter-separated, free-text values. We figured out the pattern and came up with the parsing method. We parsed the free-text data and added many new categorical.

We also processed descriptive features applying data mining techniques. We counted the most frequently used terms to understand the content of the features. We counted the words, bi-grams and tri-grams. we successfully identified the most common words and phrase and used them to add even more categorical features, enriching the dataset with meaningful data.

When the data preprocessing was done we measured Hopkins statistics to evaluate cluster tendency of the data set. The result was satisfactory; we proceeded with the clusterization.

We applied various clustering approaches trying to find hidden trends and patterns. During this exercise we realized that the clustering methods did not work as well as we expected. We used *Sighloutte* test to validate the quality of the clusters. **CLARA** approach was the most successful among the three methods we picked (they were *CLARA*, *AGNES* and *DBSCAN*). We tried univariate and multivariate datasets, different metrics and number of clusters. Conducting our research we concluded that categorical features do not cluster well. We tried scaled features and dummy-encoded ones with the same less-than-satisfactory result.

Overall we were not able apply unsupervised learning to reach our goal, but we did learn about clustering approach and developed intuition in which situation it is the best to employ them.

Bibliography

F. M. Alboukadel Kassambara. Extract and visualize the results of multivariate data analyses. URL https://cran.r-project.org/web/packages/factoextra/factoextra.pdf. [p12]

K. H. Ingo Feinerer. Text mining package. URL https://cran.r-project.org/web/packages/tm/tm.pdf. [p5]

J. P. Jiawei Han, Micheline Kamber. *Data Mining. Concepts and Techniques*. The Morgan Kaufmann Series in Data Management Systems, 225 Wyman Street, Waltham, MA 02451, USA, 2012. ISBN 978-0-12-381479-1. [p18]

M. P. Michael Hahsler. Density based clustering of applications with noise (dbscan) and related algorithms. URL https://cran.r-project.org/web/packages/dbscan/dbscan.pdf. [p19]

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