Midterm Project
Global Terrorism Dataset.
-achyutganti

The main aim of this project was to visualize those various variables in the Global terrorism dataset onto the US map. I was given four dataset – GTD dataset, Population dataset, County_city dataset and Poverty dataset.

GTD dataset – It gives the information of various attacks, venue of those attacks, groups that controlled those attacks, weapons used, number of people killed etc.

Population Dataset – This dataset has information about the city, state, population in those cities in particular years (2013,2014, 2015,2016).

County_City dataset - gives you information about cities , counties and states respectively.

Poverty Dataset – This dataset gives me the poverty percentages in those areas and the median income for the year 2015.

The first challenge was to merge those four datasets together so that we can use the poverty and population information along with the GTD information to make informative visualizations.

The way I tackled was as follows :

My main aim was to merge the remaining three datasets with the GTD dataset. I did not want to mess with the GTD dataset's observations nor its variables.

So Here is what I did.

Dealing with the Population dataset

I tired to merge the population dataset with the GTD dataset. For that I needed unique keys that would identify each observations uniquely. So my keys were **Year, State and City**. But to do that I had to do a bit of cleaning in the Population dataset. The problems I faced were

- I had to gather the column names of the population dataset (POPESTIMATE2014,POPESTIMATE2015,POPESTIMATE2016) into a column named year and the respective values into another columns and name them as cases.

```
# Gathering the dataset with gather() so that year becomes a column.
pop <- pop%>%
  gather('POPESTIMATE2013','POPESTIMATE2014','POPESTIMATE2015','POPESTIMATE2016',key='year',value='cases')%>%
  mutate(year=fct_collapse(year,
                           '2013'='POPESTIMATE2013',
                           '2014'='POPESTIMATE2014',
                           '2015'='POPESTIMATE2015',
                           '2016'='POPESTIMATE2016'))
```

Now the dataset looks pretty much like this

	SUMLEV	STATE	COUNTY	PLACE	city	provstate	iyear	cases
1	50	4	1	0	Apache	Arizona	2013	72021
2	162	4	0	2830	Apache Junction	Arizona	2013	36723
3	157	4	21	2830	Apache Junction city (pt.)	Arizona	2013	36417
4	40	4	0	0	Arizona	Arizona	2013	6624617
5	162	4	0	4720	Avondale	Arizona	2013	79561
6	157	4	1	99990	Balance of Apache	Arizona	2013	61603
7	157	4	3	99990	Balance of Cochise	Arizona	2013	50352
8	157	4	5	99990	Balance of Coconino	Arizona	2013	53226
9	157	4	7	99990	Balance of Gila	Arizona	2013	25407

- The population dataset had all the states in it while the GTD dataset had only 35 or 36. So, I filtered out the required states from the population dataset so that I would not have to deal with the big dataset.
- The next problem was at the time of merging. The city names in the population dataset and the GTD dataset differ. For example: Population dataset had Abbeville city in it while the GTD dataset had he same city listed as Abbeville. So had to remove the ‘city’ words in the end. Not only that, there are some other ctites where this problem existed. Those are the cities that had County, Town, Village, Township, Borough in the end of their names.
- But that was not it. Removing **‘city’** from the city name in the end would work for all the cities expect for one. **New York City.** So, this had to be taken care of first. So, I have changed all the **New York city** names to New York City. After doing that change the column names so that they are same in both the datasets and then merge them .

```
# Cleaning the pop$city column so that merging is possible with the GTD city column.
pop$city<-str_replace_all(pop$city,"New York city",'New York City')
pop$city<-str_replace_all(pop$city," city$",'')
#pop$city<-str_replace_all(pop$city,"\\(pt\\.\\)$",'')

pop$city<-str_replace_all(pop$city,' (c|C)ounty$','')
pop$city<-str_replace_all(pop$city,' (T|t)own$','')
pop$city<-str_replace_all(pop$city,' village$','')
pop$city<-str_replace_all(pop$city,' township$','')
pop$city<-str_replace_all(pop$city,' borough$','')
```

- Finally after merging the two datasets I found that there are some cities like **‘Ingelwood’** that were misspelled in the datasets. So, had to change the names individually so that they don’t clash and give me NA values.

Merging the County_city dataset.

County city dataset had abbreviated state names but the GTD dataset had full state names. So the first task was to add a column to the GTD dataset that contained abbreviated state names.

```
#Collect the state names into a variable.
state_information <- new_GTD$provstate
# here i matched the full names with the respective abbreviated names
#except DC. Will deal with that in the next step.
abrevated_states=state.abb[match(state_information,state.name)]
new_GTD$abrevated_states<- abrevated_states
```

But the list did not have DC in it and when I added that column to the GTD dataset I had NA values in place of all the DC entries. But it was not a huge task. I just manually added DC to the list in the next step.

```
#DC was not in the list. So i manually had to replace all teh NA values with DC
new_GTD$abrevated_states[is.na(new_GTD$abrevated_states)] <- 'DC'
```

Some city names in the county_city dataset misspelled and I had to deal with them. You can find that information in the snippet given below.

```
# Clean the County_city's City  column so that merging with the GTD dataset is possible
filtered_county$city[filtered_county$city=='Mc Cook']<-'McCook'
filtered_county$city<- str_replace_all(filtered_county$city,'New York','New York City')
filtered_county$city<- str_replace_all(filtered_county$city,'Saint Cloud','St. Cloud')
filtered_county$city<- str_replace_all(filtered_county$city,'Saint Louis','St. Louis')
filtered_county$city<- str_replace_all(filtered_county$city,'Tyngsboro','Tyngsborough')
```

Variables used as Keys for merging:
Finally the GTD dataset and the County_city dataset were joined by abbreviated state column and the City column.

I still had a problem. Some of the Cities had were absent in the County_city dataset and I had no County information merged into the GTD dataset. Those were.

Overland Park – KS
Blooming Groove – PA
Corinth – TX
Colerain – PA
Lake Los Angeles – CA

Luckily these were only a few in number. So a simple Google search gave me the required county information and a small piece of R code completed my merging.

Dealing with the Poverty dataset

As always I filtered out the states that I wanted and simplified the dataset. Just like the population dataset, the poverty dataset too had county names with County, Parish etc in the end. I had to deal with them because Poverty dataset had to be merged with the now GTD dataset (that had population and county_city merged with it) with **County and State as keys**.

```
#Remove the tails at the end eg., PArish and County
pov$Area_Name<-str_replace_all(pov$Area_Name,' (c|C)ounty$','')
pov$Area_Name<-str_replace_all(pov$Area_Name,' (P|p)arish$','')
```

Apart from that the two datasets had some county names different. They had some upper case and lower case problems. So I had to deal with them individually. Those were as follows.

```
pov$Area_Name<-str_replace_all(pov$Area_Name,'District of Columbia','District Of Columbia')
pov$Area_Name<- str_replace_all(pov$Area_Name,'McLennan','Mclennan')
pov$Area_Name<- str_replace_all(pov$Area_Name,'DuPage','Du Page')
pov$Area_Name<- str_replace_all(pov$Area_Name,'Falls Church city','Falls Church City')
pov$Area_Name<- str_replace_all(pov$Area_Name,'St. Louis city','Saint Louis City')
pov$Area_Name<- str_replace_all(pov$Area_Name,'Roanoke city','Roanoke City')
pov$Area_Name<- str_replace_all(pov$Area_Name,'St. Lucie','Saint Lucie')
```

After that merging the two datasets was possible. And the final dataset was almost a success.

The next task was to create a column called Date which had dates in yyyy-mm-dd format.

```
# Now creating a yyyy-mm-dd format for year, month and day
finaldata<- finaldata %>%
  mutate(date = make_date(iyear, imonth, iday))
```

The above code allowed me to create a new variable called date which had year,month,date in the yyyy-mm-dd format.

Finally I have recoded the missing values as -99.

```
#Recoding the NA values as -99
finaldata[is.na(finaldata)]<- -99
```

Univariate summaries

For **attack types1_txt**, I have made a tabular representation of the different categories in the variable and the count with which they occurred.

	attacktype1_txt	count
1	Armed Assault	47
2	Assassination	2
3	Bombing/Explosion	26
4	Facility/Infrastructure Attack	61
5	Hijacking	1
6	Hostage Taking (Barricade Incident)	2
7	Unarmed Assault	7

Summary for Population (cases)

```
> summary(finaldata$cases)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
     153    39924   184111  1096815   650609 10066615
>
```

From this, we can see that the Min of the data is 153, the maximum value is 10066615. We can also see the median and the mode of the data as well.

Summary for the weaptype_txt variable

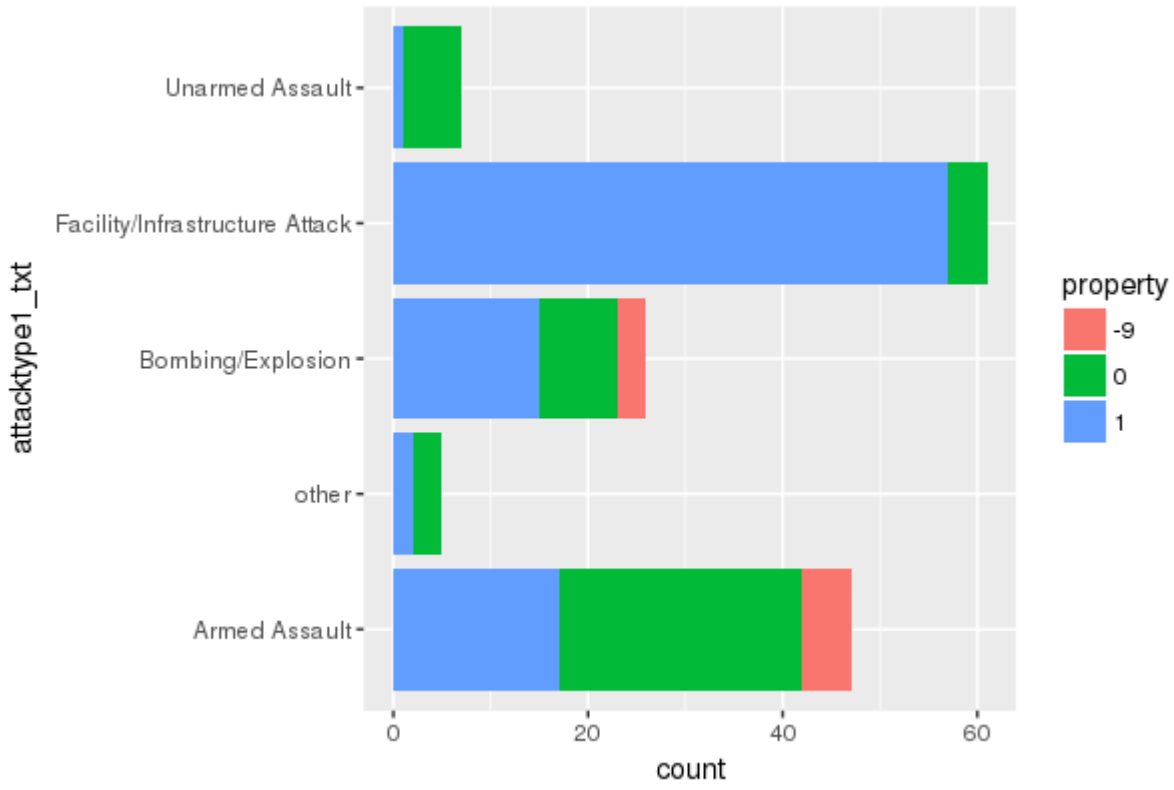
	weaptype1_txt	count
1	Biological	3
2	Chemical	2
3	Explosives/Bombs/Dynamite	26
4	Firearms	43
5	Incendiary	60
6	Melee	11
7	Other	1

Golden words to the reader:

The project was fun to do and I've made a lot of mistakes and learned a lot along the way. Few things that I felt mentioning are:

- Always save your progress. I had to write the same code thrice because I did not save my work.
- As the elders say, always write good comments. My work would have been twice as difficult if I did not write comments that were not meaningful.
- Assign variable names that makes sense. Or else, you'll have hard time figuring out what is what.
- This is what I found most useful. ** Always create a copy of your original dataset whenever you feel like exploring it more. Or, when you want to change some variables, assign the resultant dataset to a new variable. ** If you know what you are doing, then fine. If not this is my advice.

HAPPY CODING.!!!..


Unarmed Assault
Facility/Infrastructure Attack
Bombing/Explosion
other
Armed Assault
attacktype1_txt
0
20
40
60
count
property
-9
0
1