

# Data Visualization

# Laboratory Work – 1

Ravinthiran Partheepan

# Programming Language

- ▶ Task 1 and 2 – R Script and Shiny Package
- ▶ Task 3,4 and 6/7 – Plotly Package and Python

# Info about the Dataset

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Year	Name	Location	Country	Elevation	Type	Status	Time	DEATHS	DEATHS_DESCRIPTION	INJURIES	INJURI	DAMAGE	DAMAGE	HOUSES_C	HOUSES_C	TOTAL_DE	TOTAL_D
2																		
3	2010	Tungurahu	Ecuador	Ecuador	5023	Stratovolc	Historical	D1										
4	2010	Eyjafjallajökull	Iceland-S	Iceland	1666	Stratovolc	Historical	D1	55		1							2
5	2010	Pacaya	Guatemala	Guatemala	2552	Complex v	Historical	D1	76		1					1	3	1
6	2010	Sarigan	Mariana Is	United Sta	538	Stratovolc	Holocene	U	2									
7	2010	Karangeta	Sangihe Is-	Indonesia	1784	Stratovolc	Historical	D1	29		1	13	1				1	4
8	2010	Sinabung	Sumatra	Indonesia	2460	Stratovolc	Holocene	U	15		1							2
9	2010	Merapi	Java	Indonesia	2947	Stratovolc	Historical	D1	97		3	20	3	600	4		3	367
10	2010	Tungurahu	Ecuador	Ecuador	5023	Stratovolc	Historical	D1	1							1		
11	2010	Tengger	Central Java	Indonesia	2329	Stratovolc	Historical	D1	3							1		
12	2011	Merapi	Java	Indonesia	2947	Stratovolc	Historical	D1	28		1	1	1	1				1
13	2011	Kirishima	Kyushu-Jap	Japan	1700	Shield volc	Historical	D1	7			1	1					
14	2011	Bulusan	Luzon-Philippines	Philippines	1565	Stratovolc	Historical	D1	61		1							1
15	2011	Karangeta	Sangihe Is-	Indonesia	1784	Stratovolc	Historical	D1								1		
16	2011	Tungurahu	Ecuador	Ecuador	5023	Stratovolc	Historical	D1	15							1		
17	2011	Puyehue	Chile-C	Chile	2236	Stratovolc	Holocene	U								2		
18	2011	Nabro	Africa-NE	Eritrea	2218	Stratovolc	Holocene	Unknown	31		1		3		1			31
19	2011	Katla	Iceland-S	Iceland	1512	Subglacial	Historical	D2								1		
20	2011	Lokon-Eme	Sulawesi-I	Indonesia	1580	Stratovolc	Historical	D1	41		1							1
21	2011	Gamalama	Halmahera	Indonesia	1715	Stratovolc	Historical	D1	67		1	1						4
22	2012	Kilauea	Hawaiian I	United Sta	1222	Shield volc	Historical	D1								2		
23	2012	Kilauea	Hawaiian I	United Sta	1222	Shield volc	Historical	D1	18						1	1	1	
24	2012	Tolbachik	Kamchatka	Russia	3682	Shield volc	Historical	D1							1			
25	2013	Merapi	Java	Indonesia	2947	Stratovolc	Historical	D1	1		1	1	1	1				1
26	2013	Paluweh	Lesser Sunda	Indonesia	875	Stratovolc	Historical	D1							1		1	
27	2013	Mayon	Luzon-Philippines	Philippines	2462	Stratovolc	Historical	D1	5		1	8	1					5
28	2013	Paluweh	Lesser Sunda	Indonesia	875	Stratovolc	Historical	D1	5		1							5
29	2013	Ubinas	Peru	Peru	5672	Stratovolc	Historical	D1								1		
30	2013	Sakurajima	Kyushu-Jap	Japan	1117	Stratovolc	Historical	D1	24						1			
31	2013	Sinabung	Sumatra	Indonesia	2460	Stratovolc	Holocene	U							2			
32	2013	Okataina	New Zealand	New Zealand	1111	Lava dome	Historical	D1	1		1							1

# Attribute and it's Type

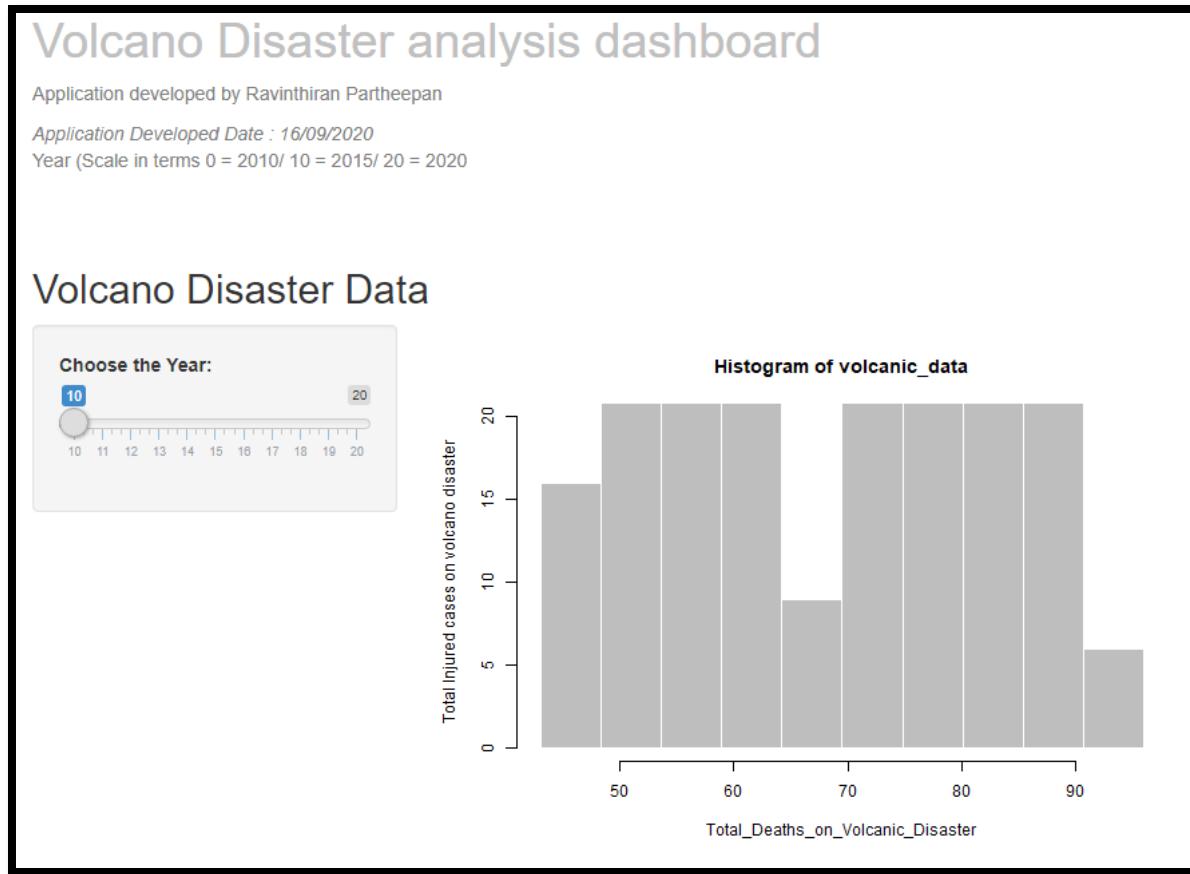
- ▶ This dataset is based on quantitative type.
- ▶ Quantitative attribute – Year,  
Total\_death\_on\_Volcano\_Disaster and  
Total\_Injured\_cases\_on\_Volcano\_Disaster
- ▶ Attribute Type – Ratio type

# Why it is a Ratio type ?

- ▶ It's a numeric data.
- ▶ Ordered Values – selected attribute “Year”
- ▶ Can compute the variations from a non zero point value.
- ▶ Eg, 2010-2011-2012,.....2019

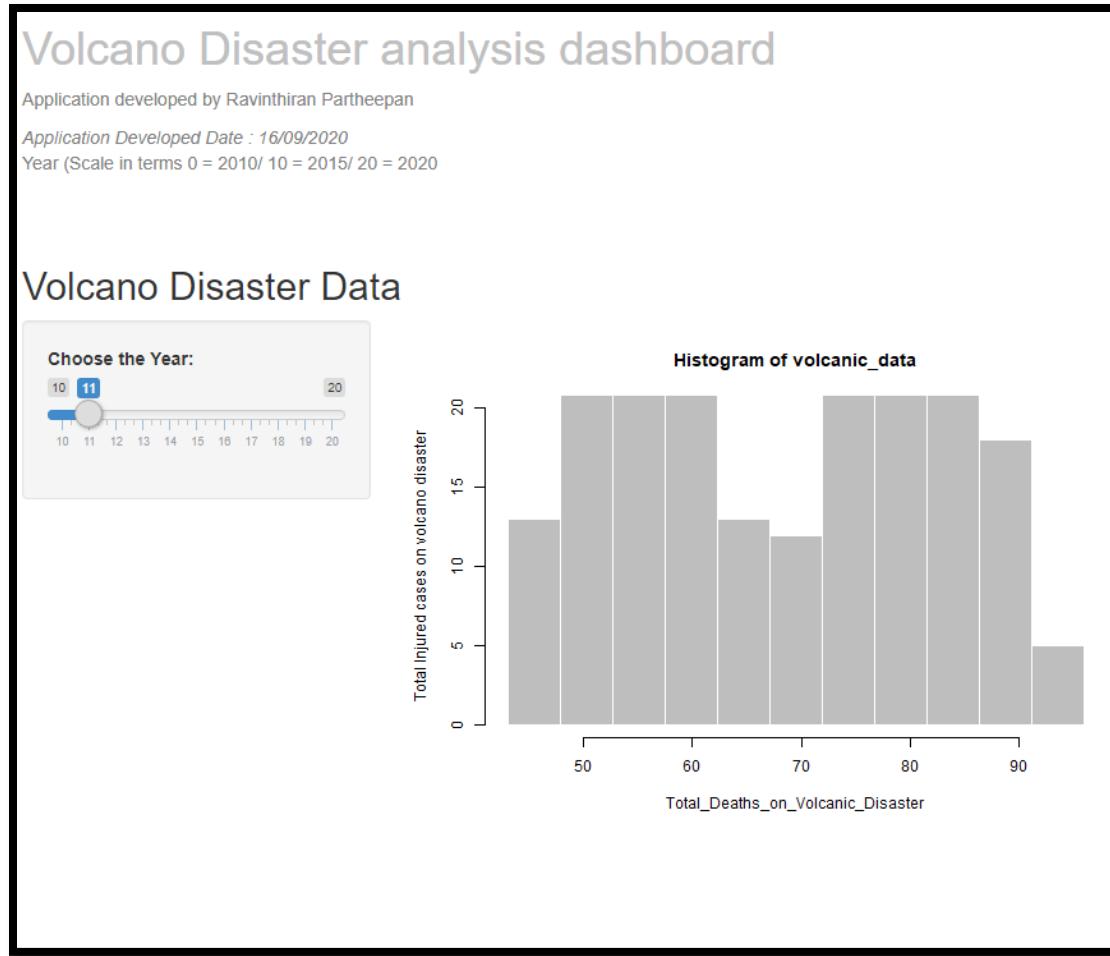
# 2010 Volcano Disaster Analysis

## Visualization with histogram Plot



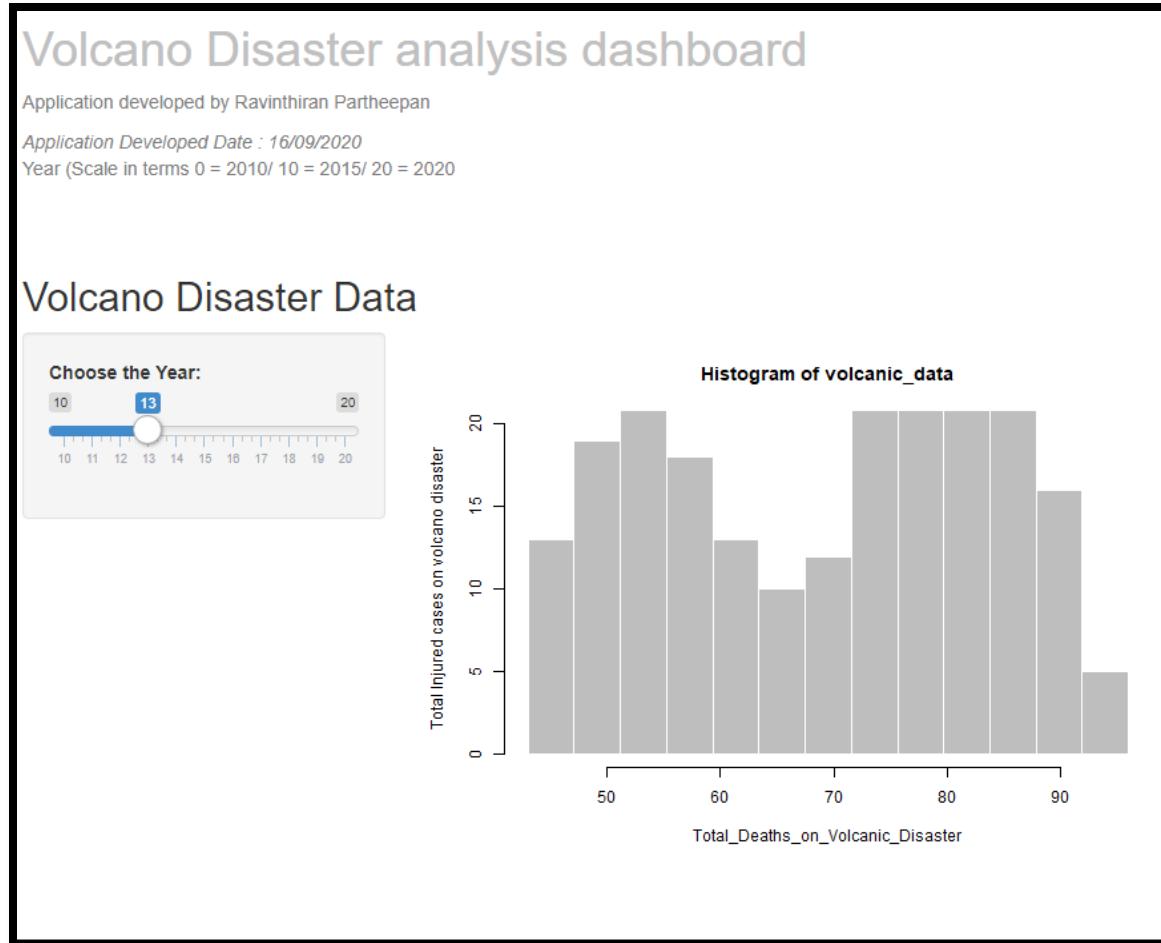
# 2011 Volcano Disaster Analysis

## Visualization with histogram Plot



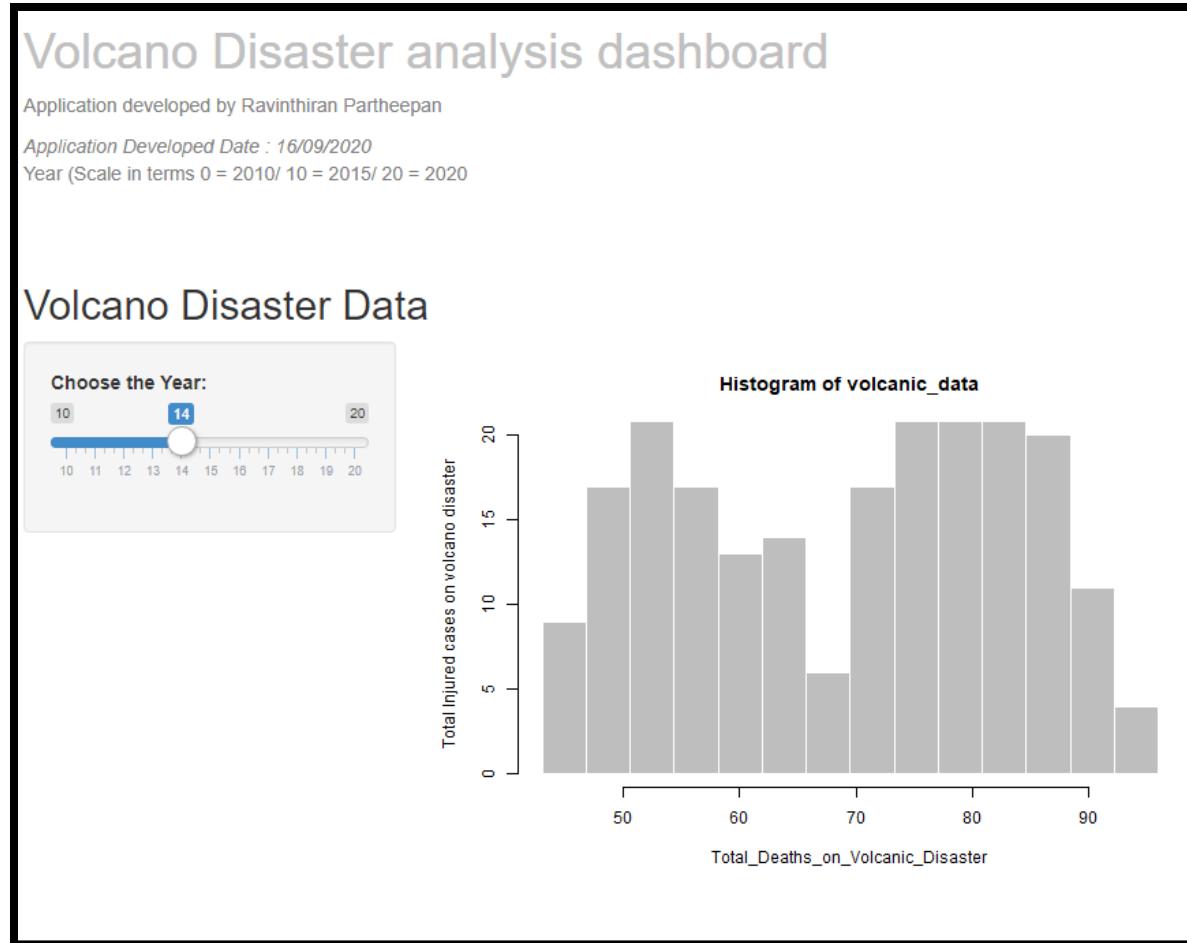
# 2013 Volcano Disaster Analysis

## Visualization with histogram Plot



# 2014 Volcano Disaster Analysis

## Visualization with histogram Plot



# Interactive Data Visualization

- ▶ Programming language Used – R
- ▶ Package Used – Shiny

- ▶ UI.R (Front End) -----→

The screenshot shows the RStudio interface with two tabs open: ui.R and server.R. The ui.R tab contains the following R code:

```
ui.R* library(shiny)
shinyUI(fluidPage(
  h1("Volcano Disaster analysis dashboard", style="color:silver", Position="center"),
  p("Application developed by Ravinthiran Partheepan", style = "color:grey"),
  em("Application Developed Date : 16/09/2020", style="color: grey"),
  br(),
  p("Year (Scale in terms 0 = 2010/ 10 = 2015/ 20 = 2020", style= "color:grey"),
  br(),
  br(),
  titlePanel("Volcano Disaster Data"),
  sidebarLayout(
    sidebarPanel(
      sliderInput("bins",
                  "choose the year:",
                  min = 10,
                  max = 20,
                  value = 17
    ),
    mainPanel(
      plotOutput("distPlot")
    )
  )
))
```

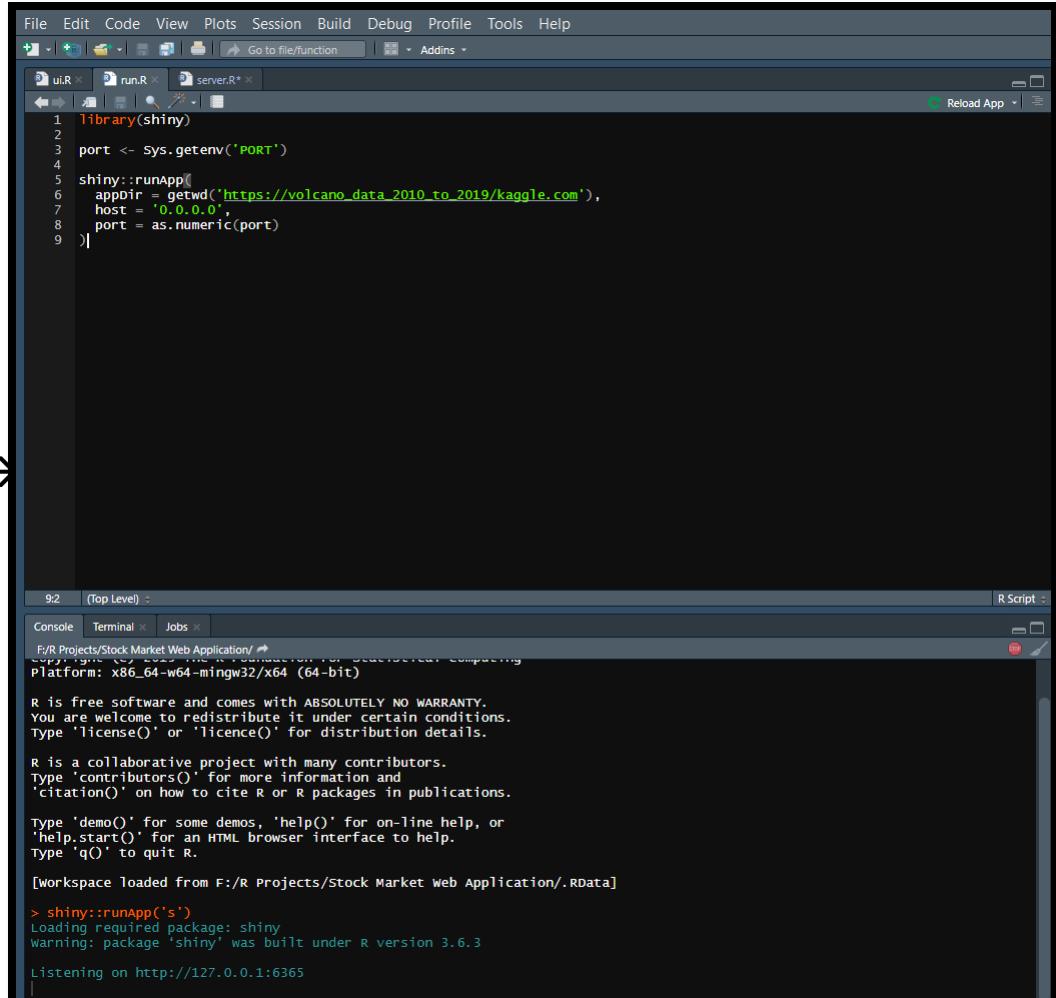
The server.R tab is partially visible at the top. Below the tabs, the R Script pane shows the following terminal output from the R console:

```
16:1 | (Top Level) :
Console Terminal Jobs
F/R Projects/Stock Market Web Application/ ↗
> runApp('s')
Listening on http://127.0.0.1:7565

> |
```

# Interactive Data Visualization

- ▶ Programming language Used-R
- ▶ Package Used – Shiny
  
- ▶ Port Connectivity ----- →



The screenshot shows the RStudio interface with three tabs open: ui.R, run.R, and server.R\*. The ui.R tab contains a single line of code: `library(shiny)`. The run.R tab contains the following R code:

```
1 library(shiny)
2 port <- Sys.getenv("PORT")
3 shiny::runApp(
4   appDir = getwd(),
5   host = '0.0.0.0',
6   port = as.numeric(port)
7 )
```

The server.R\* tab is currently selected. The R console window at the bottom displays the following output:

```
F:/R Projects/Stock Market Web Application/ ↵
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[workspace loaded from F:/R Projects/Stock Market Web Application/.RData]

> shiny::runApp('s')
Loading required package: shiny
warning: package 'shiny' was built under R version 3.6.3

Listening on http://127.0.0.1:6365
```

# Interactive Data Visualization

- ▶ R Script
- ▶ Package Used – Shiny
- ▶ Port Connectivity ---→

The screenshot shows the RStudio interface. The top panel displays an R script with code for a shiny application. The bottom panel shows the R console output.

```
File Edit Code View Plots Session Build Debug Profile Tools Help
ui.R run.R server.R*
< Go to file/function Addins Reload App < > < > < > < >
1
2
3 library(shiny)
4
5 v <- c()
6
7 shinyServer(function(input, output) {
8
9   output$distPlot <- renderPlot({
10
11     app$column <- column(year)$column(total_deaths+total_injuries)
12     volcanic_data <- faithful[, 2]
13     bins <- seq(min(volcanic_data), max(volcanic_data), length.out = input$bins + 1)
14
15     hist(volcanic_data, xlab="Total_Deaths_on_Volcanic_Disaster", ylab=" Total Injured cases on volcano disaster ", y
16   })
17 }
18 )
19
20 })
21
```

11:13 <function>(input, output) : R Script

Console Terminal Jobs

F/R Projects/Stock Market Web Application/

R version 3.6.3 (2019-12-12) -- "Kite and Sword" Copyright (C) 2019 The R Foundation for Statistical Computing Platform: x86\_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.  
R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.  
Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.

[workspace loaded from F:/R Projects/Stock Market Web Application/.RData]

> shiny::runApp('s')

Loading required package: shiny  
Warning: package "shiny" was built under R version 3.6.3

Listening on http://127.0.0.1:6365

# Heroku (PaaS)

- ▶ Link to the web application :-
- ▶ <https://volcano-disaster-analysis-rp.herokuapp.com/>

# Task No. 8

## Text Visualization

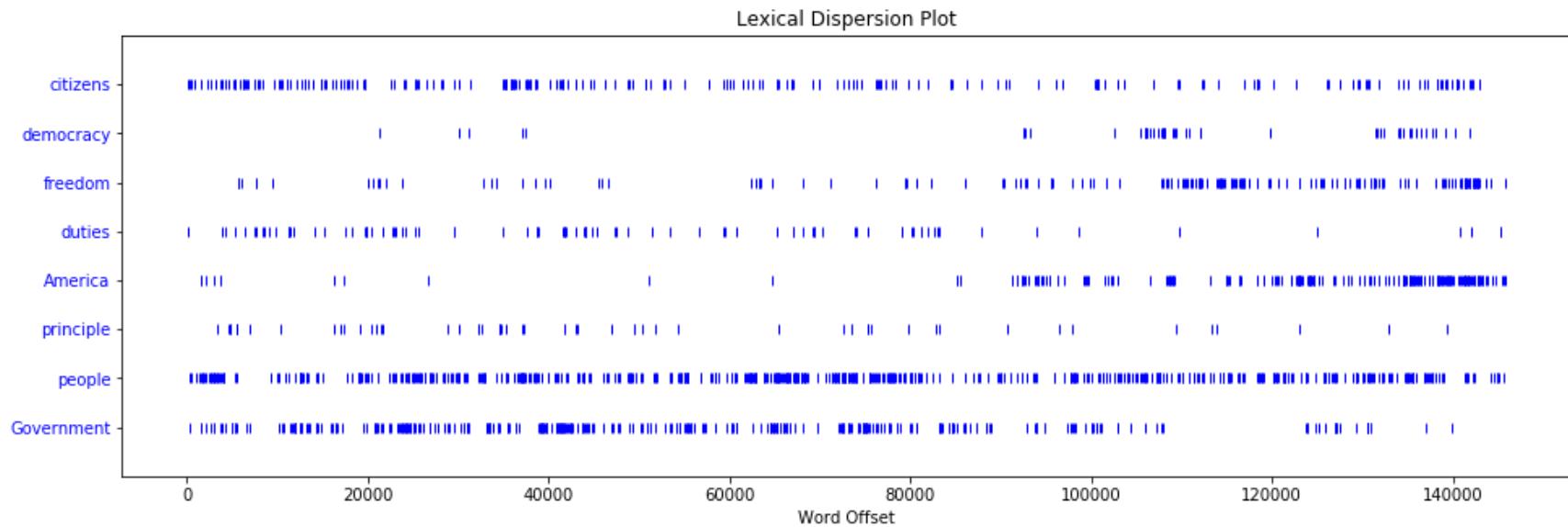
## About Dataset - Presidential Inaugural Addresses

The screenshot shows a Microsoft Excel spreadsheet titled "inaug\_speeches - Microsoft Excel". The ribbon menu is visible at the top, with tabs for Home, Insert, Page Layout, Formulas, Data, Review, View, Developer, and Team. The Home tab is selected.

The main content area displays a table with 32 rows of data, starting from row 1. The columns are labeled A through Q. Column A contains row numbers from 1 to 32. Column B is labeled "Name" and lists the presidents. Column C is labeled "Inaugural Address" and lists the locations and dates. Column D is labeled "Date" and lists the specific dates. Column E is labeled "text" and contains the full text of each inaugural speech.

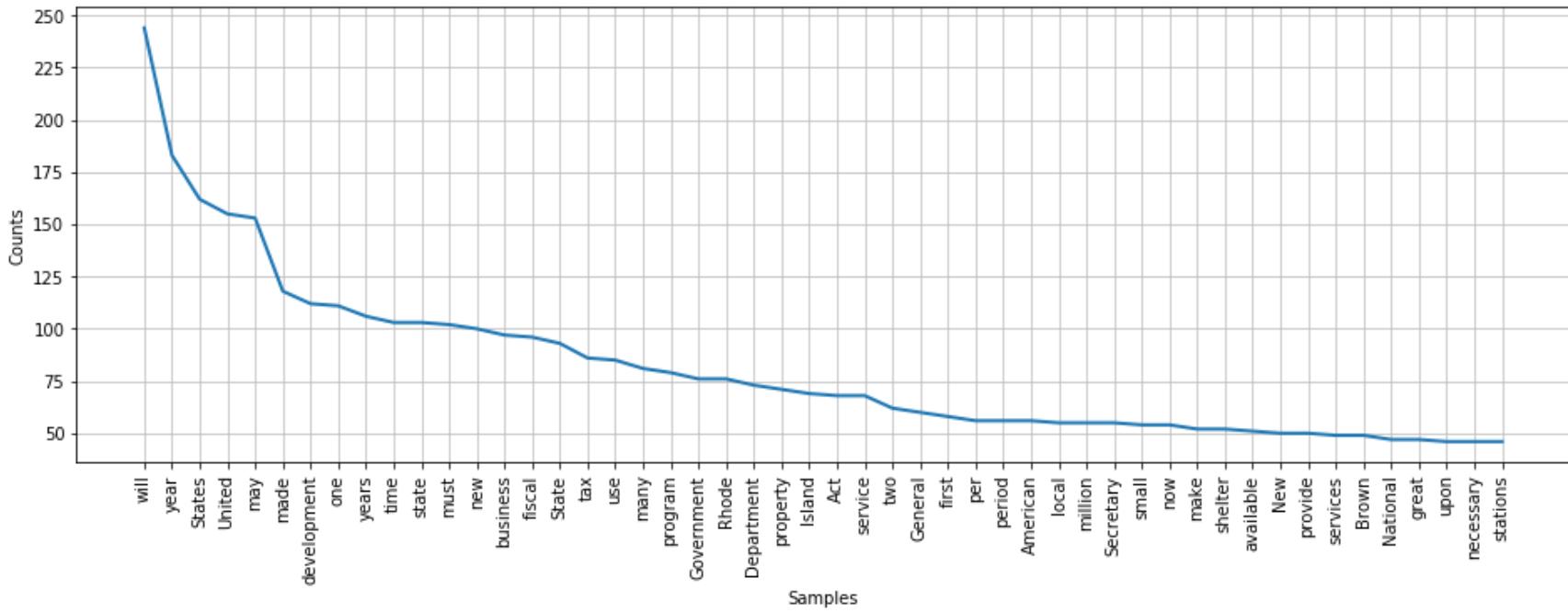
On the left side of the screen, there is a sidebar titled "Document Recovery" which lists several recovered files, and a "Available files" section which lists the same files again. At the bottom left, there is a link "Which file do I want to save?"

# Lexical Dispersion



Lexical diversity lets us what is the percentage of the unique words in the text corpus

# Word Counts

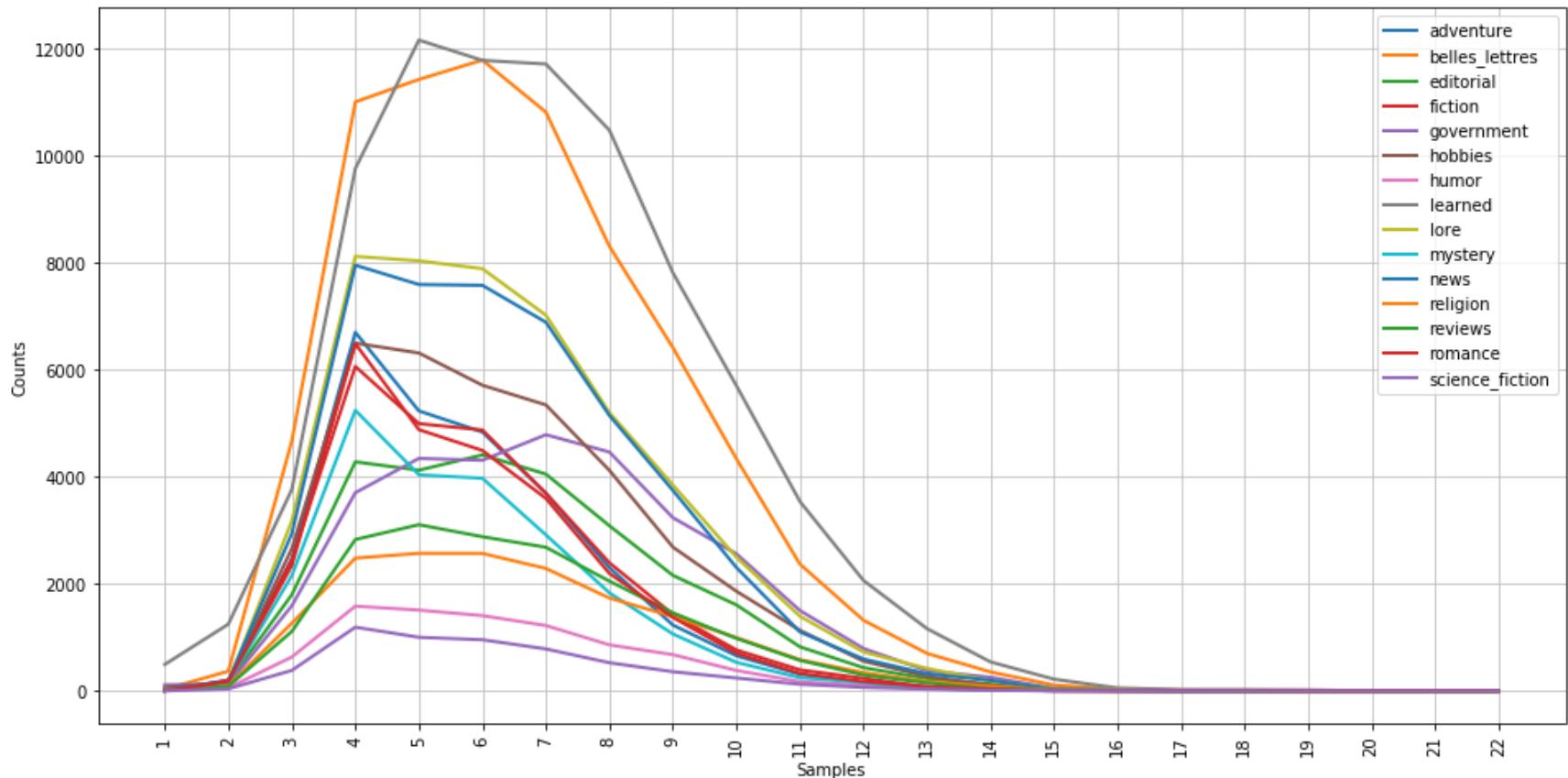


# Word Cloud



WordClouds help in detecting the words that occur frequently.

# Word Length Distribution



This plot helps to visualize the composition of different word length in the text corpus.

# Program Code - Python and NLTK

```
In [1]: import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
from wordcloud import WordCloud,STOPWORDS
```

```
In [4]: import nltk  
nltk.download('inaugural')  
  
[nltk_data] Downloading package inaugural to  
[nltk_data]     C:\Users\RAVINTHIRAN\AppData\Roaming\nltk_data...  
[nltk_data]  Unzipping corpora\inaugural.zip.
```

Out[4]: True

```
In [5]: from nltk.corpus import inaugural
data = pd.read_csv('F:/Applied Informatics/Semester III/Data Visualization/Homeworks/Task8/archive.csv')
text = inaugural.raw()
wordcloud = WordCloud(max_font_size=60).generate(text)
plt.figure(figsize=(16,12))

plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.show()
```



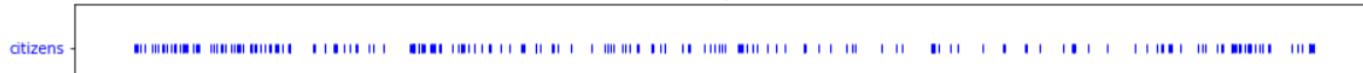
# Program Code

```
In [14]: nltk.download('gutenberg')
nltk.download('treebank')
from nltk.book import text4 as inaugural_speeches
plt.figure(figsize=(16,5))
topics = ['citizens', 'democracy', 'freedom', 'duties', 'America', 'principle', 'people', 'Government']
inaugural_speeches.dispersion_plot(topics)
```

```
[nltk_data] Downloading package gutenberg to
[nltk_data]      C:\Users\RAVINTHIRAN\AppData\Roaming\nltk_data...
[nltk_data] Package gutenberg is already up-to-date!
[nltk_data] Downloading package treebank to
[nltk_data]      C:\Users\RAVINTHIRAN\AppData\Roaming\nltk_data...
[nltk_data]  Unzipping corpora\treebank.zip.
```

```
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
```

Lexical Dispersion Plot



# Program Code

```
In [16]: nltk.download('brown')
from nltk.corpus import brown
stop_words = set(STOPWORDS)
topics = ['government', 'news', 'religion','adventure','hobbies']
for topic in topics:
    # filter out stopwords and punctuation mark and only create array of words
    words = [word for word in brown.words(categories=topic)
             if word.lower() not in stop_words and word.isalpha() ]
    freqdist = nltk.FreqDist(words)
    # print 5 most frequent words
    print(topic,'more :', ' ', '.join([ word.lower() for word, count in freqdist.most_common(5)]))
    # print 5 least frequent words
    print(topic,'less :', ' ', '.join([ word.lower() for word, count in freqdist.most_common()[-5:]]))
```

```
[nltk_data] Downloading package brown to
[nltk_data]     C:\Users\RAVINTHIRAN\AppData\Roaming\nltk_data...
[nltk_data]     Unzipping corpora\brown.zip.
```

```
government more : will , year , states , united , may
government less : load , cadre , perception , lengthened , shadow
news more : said , will , one , last , two
news less : pupils , render , vitally , richer , fuller
religion more : god , world , one , may , new
religion less : medium , sat , ivory , velvet , sadness
adventure more : said , back , man , one , time
adventure less : snick , fisted , overhand , plunge , insanely
hobbies more : will , one , may , time , two
hobbies less : explore , hinterlands , bride , winner , sweepstakes
```

```
In [17]: corpus_genre = 'government'
words = [word for word in brown.words(categories=corpus_genre) if word.lower() not in stop_words and word.isalpha() ]
freqdist = nltk.FreqDist(words)
```

# Task No.2

- ▶ A box plot is used for summarizing a set of data measured on an interval scale.
- ▶ It is often used in explanatory data analysis.
- ▶ This type of graph is used to show the shape of the distribution, its central value, and its variability.

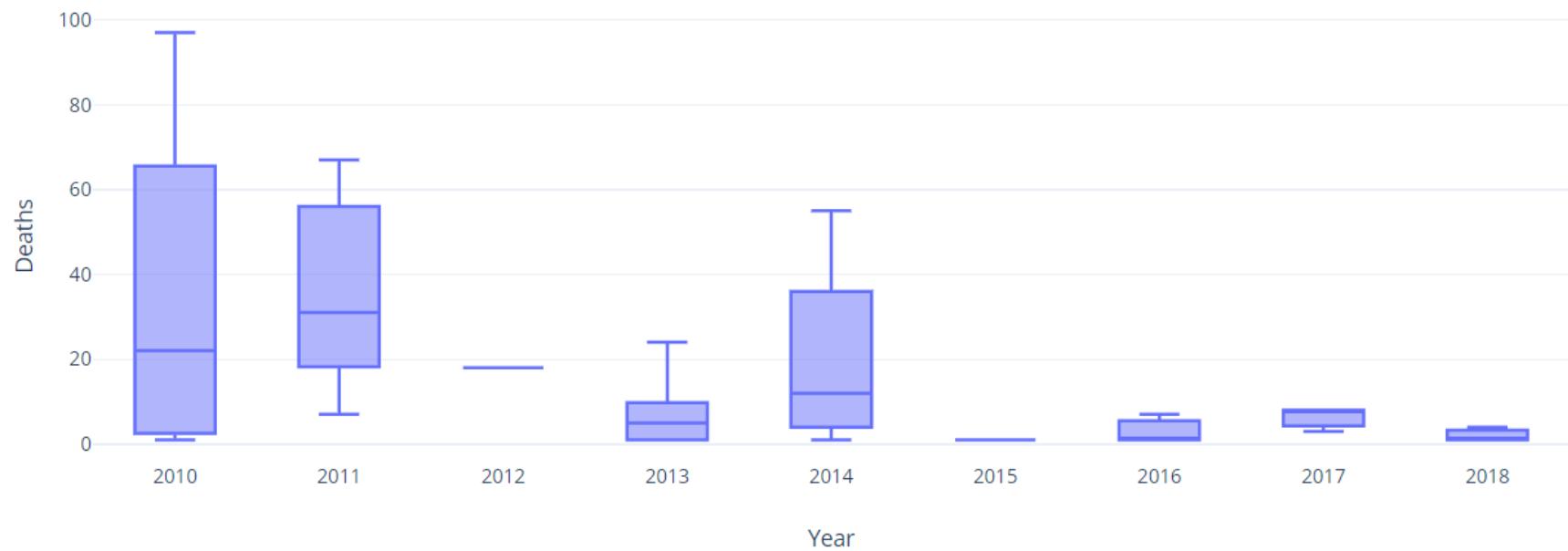
# Programming Language

- ▶ Task 1 and 2 – R Script and Shiny Package
- ▶ Task 3,4 and 6/7 – Plotly Package and Python

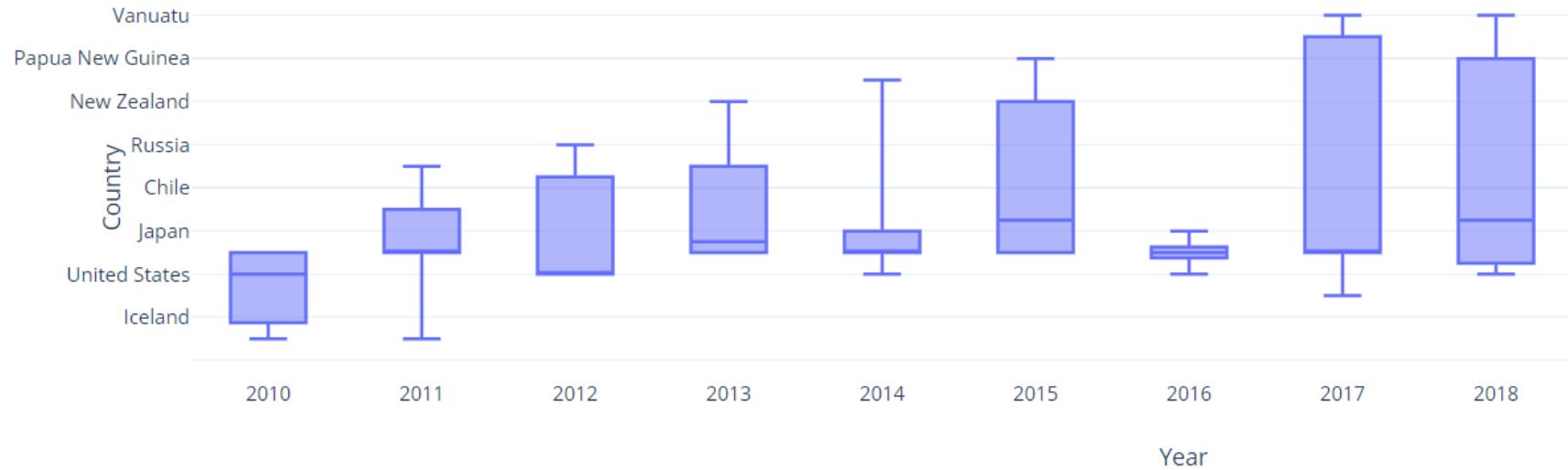
# Info about the Dataset

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Year	Name	Location	Country	Elevation	Type	Status	Time	DEATHS	DEATHS_DESCRIPTION	INJURIES	INJURI	DAMAGE	DAMAGE	HOUSES_C	HOUSES_C	TOTAL_DE	TOTAL_D
2																		
3	2010	Tungurahu	Ecuador	Ecuador	5023	Stratovolc	Historical	D1										
4	2010	Eyjafjallajökull	Iceland-S	Iceland	1666	Stratovolc	Historical	D1	55		1							2
5	2010	Pacaya	Guatemala	Guatemala	2552	Complex v	Historical	D1	76		1					1	3	1
6	2010	Sarigan	Mariana Is	United Sta	538	Stratovolc	Holocene	U	2									
7	2010	Karangeta	Sangihe Is-	Indonesia	1784	Stratovolc	Historical	D1	29		1	13	1				1	4
8	2010	Sinabung	Sumatra	Indonesia	2460	Stratovolc	Holocene	U	15		1							2
9	2010	Merapi	Java	Indonesia	2947	Stratovolc	Historical	D1	97		3	20	3	600	4		3	367
10	2010	Tungurahu	Ecuador	Ecuador	5023	Stratovolc	Historical	D1	1							1		
11	2010	Tengger	Central Java	Indonesia	2329	Stratovolc	Historical	D1	3							1		
12	2011	Merapi	Java	Indonesia	2947	Stratovolc	Historical	D1	28		1	1	1	1				1
13	2011	Kirishima	Kyushu-Jap	Japan	1700	Shield volc	Historical	D1	7			1	1					
14	2011	Bulusan	Luzon-Philippines	Philippines	1565	Stratovolc	Historical	D1	61		1							1
15	2011	Karangeta	Sangihe Is-	Indonesia	1784	Stratovolc	Historical	D1								1		
16	2011	Tungurahu	Ecuador	Ecuador	5023	Stratovolc	Historical	D1	15							1		
17	2011	Puyehue	Chile-C	Chile	2236	Stratovolc	Holocene	U								2		
18	2011	Nabro	Africa-NE	Eritrea	2218	Stratovolc	Holocene	Unknown	31		1		3		1			31
19	2011	Katla	Iceland-S	Iceland	1512	Subglacial	Historical	D2								1		
20	2011	Lokon-Eme	Sulawesi-I	Indonesia	1580	Stratovolc	Historical	D1	41		1							1
21	2011	Gamalama	Halmahera	Indonesia	1715	Stratovolc	Historical	D1	67		1	1						4
22	2012	Kilauea	Hawaiian I	United Sta	1222	Shield volc	Historical	D1								2		
23	2012	Kilauea	Hawaiian I	United Sta	1222	Shield volc	Historical	D1	18						1	1	1	
24	2012	Tolbachik	Kamchatka	Russia	3682	Shield volc	Historical	D1							1			
25	2013	Merapi	Java	Indonesia	2947	Stratovolc	Historical	D1	1		1	1	1	1				1
26	2013	Paluweh	Lesser Sunda	Indonesia	875	Stratovolc	Historical	D1							1		1	
27	2013	Mayon	Luzon-Philippines	Philippines	2462	Stratovolc	Historical	D1	5		1	8	1					5
28	2013	Paluweh	Lesser Sunda	Indonesia	875	Stratovolc	Historical	D1	5		1							5
29	2013	Ubinas	Peru	Peru	5672	Stratovolc	Historical	D1								1		
30	2013	Sakurajima	Kyushu-Jap	Japan	1117	Stratovolc	Historical	D1	24						1			
31	2013	Sinabung	Sumatra	Indonesia	2460	Stratovolc	Holocene	U							2			
32	2013	Okataina	New Zealand	New Zealand	1111	Lava dome	Historical	D1	1		1							1

# Box Plot – Volcano Year and Death



# Box Plot – Volcano Country and Year



# Program Code - R and Shiny

The screenshot shows the RStudio interface with the following components:

- Top Bar:** Contains the RStudio logo, menu items (File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help), and a "Project (None)" dropdown.
- Code Editor:** Displays the `dataviz.R` file containing R code for a Shiny application. The code defines a UI with a fluid page, a title, a header, and a box plot. It also defines a server function that creates a box plot from the `mtcars` dataset based on the selected input year and country. The code concludes with `shinyApp(ui, server)`.
- Environment Tab:** Shows the Global Environment tab with the message "Environment is empty".
- Console Tab:** Displays the R version information and the standard R welcome message. It ends with a prompt "`> |`".
- Bottom Taskbar:** Shows the Windows taskbar with various pinned icons (File Explorer, Edge, Mail, Photos, OneDrive, Google Chrome, File History, Task View, Taskbar settings) and system status indicators (battery level at 47%, signal strength, battery level, ENG, 09-11-2020, 14:30).

# Program Code - R and Shiny

The screenshot shows the RStudio interface with two main panes. The left pane displays R code for a shiny application. The right pane shows the R console and environment.

**RStudio Environment:**

- Project: (None)
- Files: dataviz.R, run.R, ui.R, server.R
- Run App: Run App
- Environment: Global Environment
- Global Environment: Environment is empty

**RStudio Console:**

```
1:1 (Top Level) - R Script -> |
```

R version 3.6.1 (2019-07-05) -- "Action of the Toes"  
Copyright (C) 2019 The R Foundation for Statistical Computing  
Platform: x86\_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.

> |

# Task – 3 Distance Metrics

- ▶ **Euclidean** - Direction gives the direction from each cell to the closest source.
- ▶ **Manhattan** - Manhattan distance, also known as city block distance and used calculate the distance between two data points in a grid-like path.
- ▶ **Simple Matching Co-efficient** – It gives equal information and diversity about data.
- ▶ **Jaccard** - used to calculate the intersection of  $n \times n$  clusters and divided by the union of two sets.
- ▶ **Cosine** – It is a metric used to measure the angle between two vectors and determines which vectors are in same direction.

# Time Series Plot

- ▶ Time series graphs can be used to visualize trends in counts or numerical values over time.
- ▶ Because date and time information is continuous categorical data (expressed as a range of values),
- ▶ points are plotted along the x-axis and connected by a continuous line.

# Ploty - Geomap program code

```
task3.py - F:/Applied Informatics/Semester-III/Data Visualization/Laboratory Works/Homework 3/task3.py (3.7.2)
File Edit Format Run Options Window Help
from urllib.request import urlopen
import json
with urlopen('https://raw.githubusercontent.com/plotly/datasets/master/geojson-counties-fips.json') as response: counties = json.load(response)
import pandas as pd
df = pd.read_csv('F:/Applied Informatics/Semester-III/Data Visualization/Laboratory Works/Homeowrk_1/volcano_data_2010.csv', dtype={"fips": str})
df.reset_index(inplace=True)

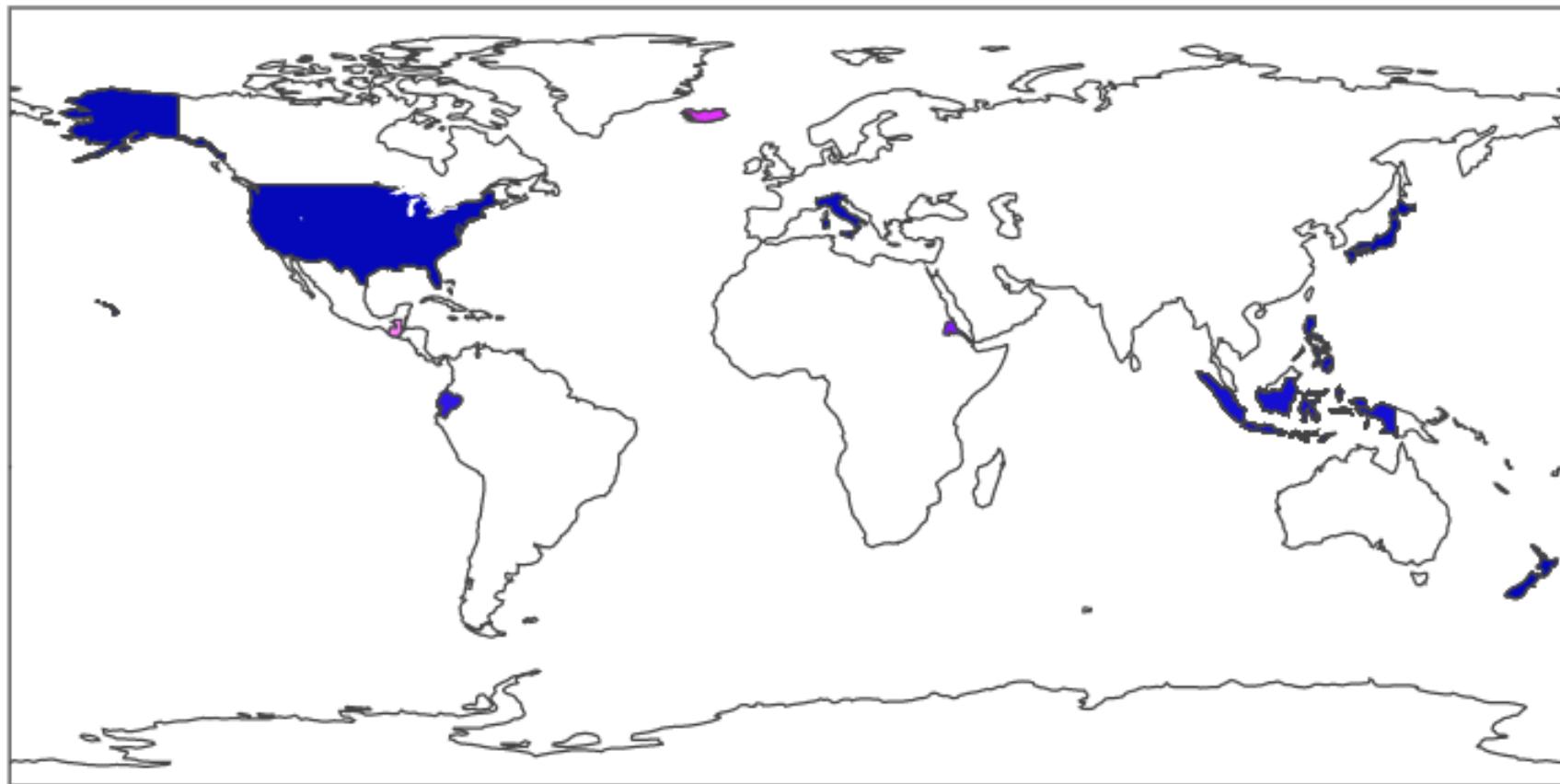
df['year'] = [d.year for d in df.elevation]
df['elevation'] = [d.strftime('%b') for d in df.elevation]
years = df['year'].unique()

np.random.seed(100)
mycolors = np.random.choice(list(mpl.colors.XKCD_COLORS.keys()), len(years), replace=False)

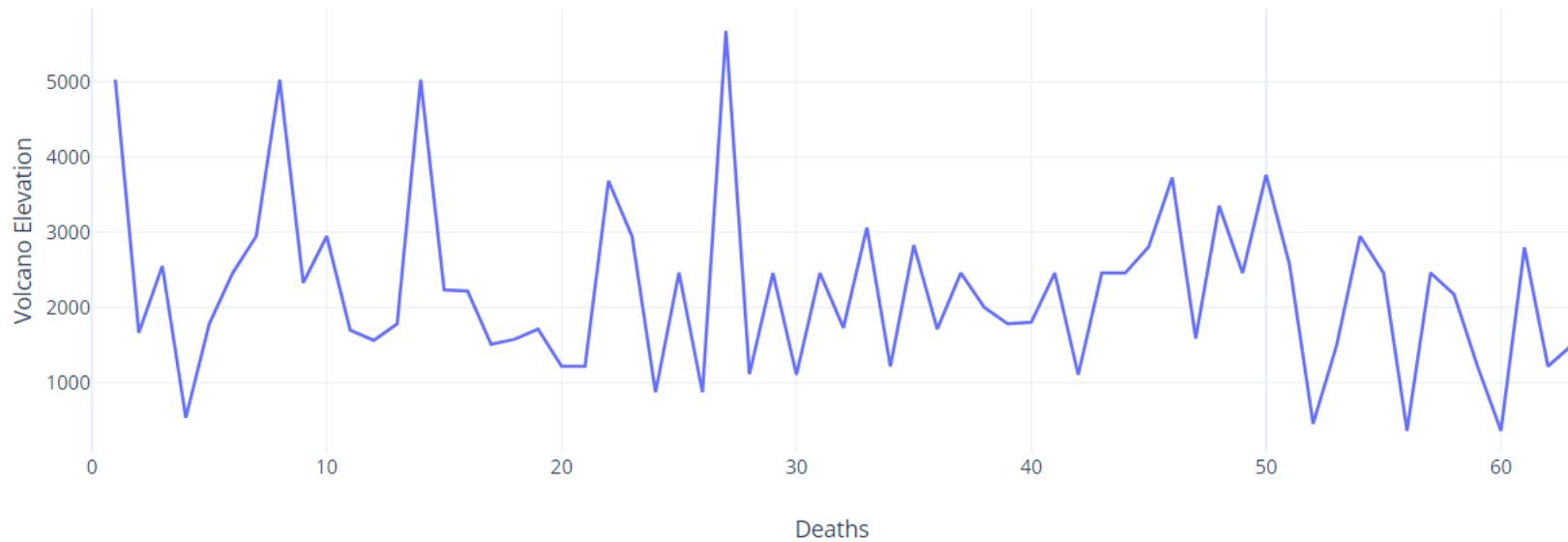
plt.figure(figsize=(16,12), dpi= 80)
for i, y in enumerate(years):
    if i > 0:
        plt.plot('month', 'value', data=df.loc[df.year==y, :], color=mycolors[i], label=y)
        plt.text(df.loc[df.year==y, :].shape[0]-.9, df.loc[df.year==y, 'value'][-1:], y, fontsize=12, color=mycolors[i])

# Decoration
plt.gca().set(xlim=(-0.3, 11), ylim=(2, 30), ylabel="$Elevation", xlabel="$Year$")
plt.yticks(fontsize=12, alpha=.7)
plt.title("Seasonal Plot of Volcanic eruption elevation Time Series", fontsize=20)
plt.show()
import plotly.express as px
fig = px.choropleth(df, geojson=counties, locations='fips', color='elevation', color_continuous_scale="Viridis", range_color=(0, 12), scope='world', labels={'elevation':'Elevation'})
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```

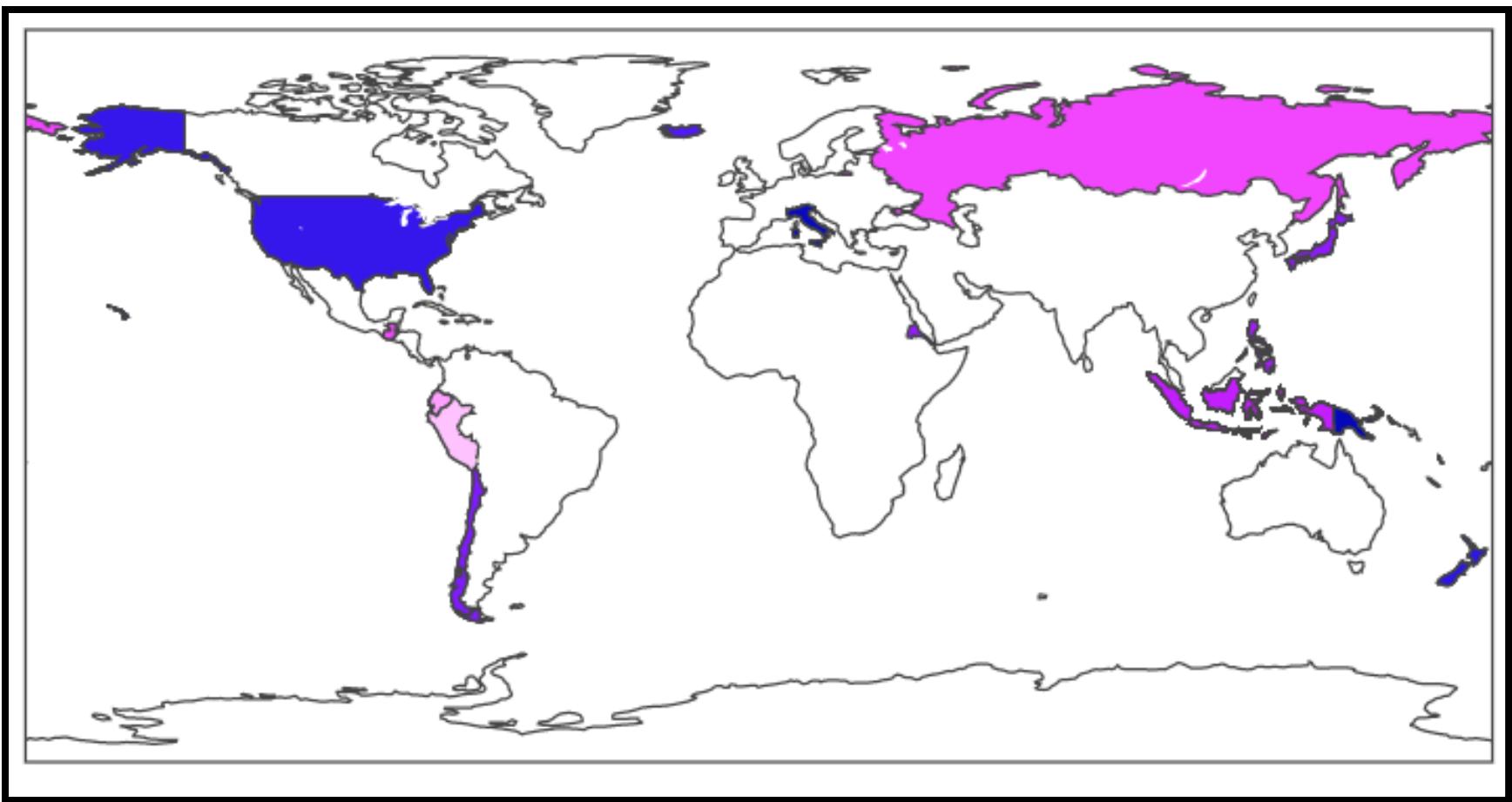
# Volcano eruption - Countries



# Time series – Volcano Elevation and Deaths



# Countries and Elevation



# Task No.4 – Multi-dimensional data visualization

- ▶ It represents one dimension as point and 2D, 3D as a object or graph.
- ▶ It is used for descriptive data analysis.
- ▶ The dimensions is a cluster of related objects also known as attributes.
- ▶ Multidimensional data is used to provide information about fact (Numerical) data in or more perspective.

# Common Methods to Visualize

- ▶ The **facet** approach partitions a **plot** into a matrix of panels.
- ▶ Each panel shows a different subset of the data.
- ▶ Colors – To show different groups / classes.
- ▶ Shapes – To show variations.
- ▶ Sizes – To show the analytic measurements.

# Correlation Analysis

- ▶ Pearson Correlation coefficient also known as bivariate correlation
- ▶ It is a statistic that measures linear correlation between two variables  $X$  and  $Y$ .
- ▶ It has a value between +1 and -1.
- ▶ A value of +1 is total positive linear correlation, 0 is no linear correlation, and -1 is total negative linear correlation
- ▶ **Pearson Correlation**  $(x,y) = \text{cov}(x,y) / \text{std}(x) * \text{std}(y)$
- ▶  $\text{Cov}(x,y) = \text{Sum}[(x - \text{mean}(x))(Y - \text{mean}(y))]$
- ▶ **Correlation**  $= [xy] - [x] * [y] / \text{Sqrt}([x^2] - [x]^2) * \text{Sqrt}([y^2] - [y]^2)$

# Year 2010 and 2011 Correlation Analysis

	Year	Elevation	DEATHS Year 2010	INJURIES	TOTAL_MISSING
TOTAL_MISSING	-0.051	-0.875	0.543	-0.203	1
INJURIES	0.023	-0.437	0.781	1	-0.203
DEATHS	0.678	0.103	1	0.781	0.543
Elevation	-0.543	1	0.103	-0.437	-0.875
Year	1	-0.543	0.678	0.023	-0.051

# Program Code

- ▶ Import pandas as pd , numpy as np
- ▶ Import seaborn as sns
- ▶ Import math
- ▶ Import matplotlib.pyplot as plt
- ▶ Import scipy.stats.pearsonr
  
- ▶ Volcano\_filepath = “E:/Applied informatics/semester-III/Data Visualization/Homework4/volcano\_data\_2010.csv”
- ▶ Volcano\_data = pd.read\_csv(volcano\_filepath, index\_col="Month")
  
- ▶ Vc = Volcano\_data
- ▶ X = Volcano\_filepath  
plt.sns.barplot(x=volcano\_data.index, y=Volcano\_data['NK'])
- ▶ Cov\_volcano = cor(Vc, Volcano\_data)
  
- ▶ Data\_Volcano = **scipy.stats.pearsonr(x, y)**
- ▶ plt.figure(figsize=(14,7))
- ▶ plt.title(“Volcano Correlation analysis 2010 and 2011”)
- ▶ sns.heatmap(data=Volcano\_data, annot=True)

Task - 6/7

# Principal Component Analysis

- ▶ **Principal component analysis (PCA)** is a technique used to emphasize variation and bring out strong patterns in a dataset.
- ▶ It's often used to make **data** easy to explore and **visualize**.
- ▶ Projects higher dimensional data into lower dimensional data
- ▶ It combines both the dimesion correlated features into new features.
- ▶ Provides valuable insights beyond descriptive statistics and helps to discover underlying patterns

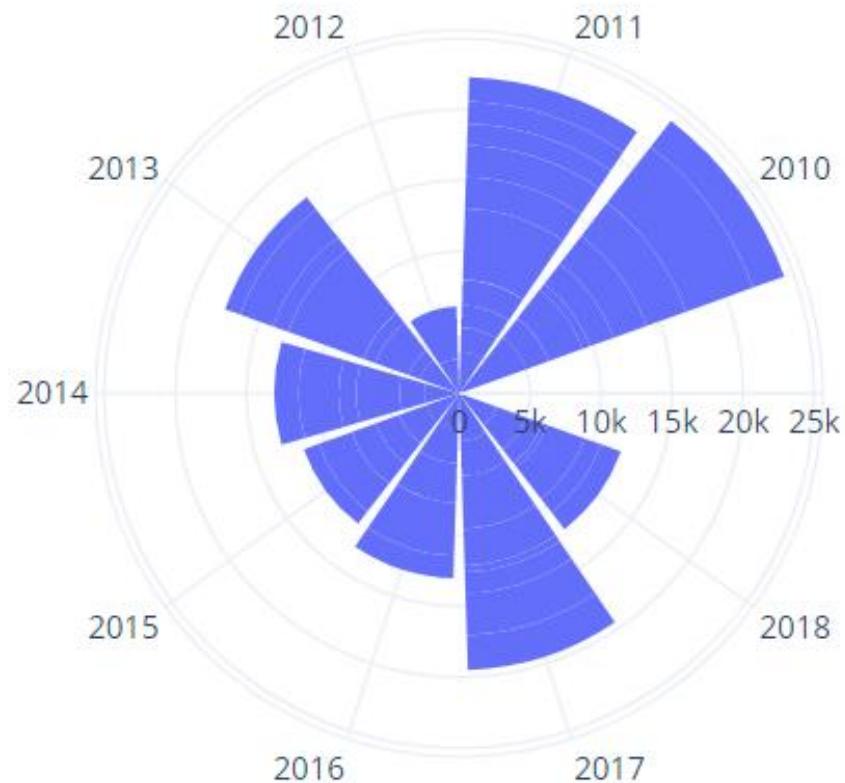
# PCA – Metrics for Data Exploration

- ▶ Explained Variance - measures how much a model can reflect the variance of the whole data. Principle components try to capture as much of the variance as possible and this measure shows to what extent they can do that.
- ▶ Factor loading - indicates how much a variable correlates with a component. Each component is made of a linear combination of variables, where some might have more weight than others. Factor loadings indicate this as correlation coefficients, ranging from -1 to 1, and make components interpretable.

# Program Code – Plotly

```
import numpy as np
import matplotlib.pyplot as plt
data = F://Applied Informatics/Semester-III/volcano_data_2010.csv
Compute pie slices N = 20
theta = np.linspace(0.0, 2 * np.pi, N, endpoint=False)
radii = 10 * np.random.rand(N)
width = np.pi / 4 * np.random.rand(N)
colors = plt.cm.viridis(radii / 10.)
ax = plt.subplot(111, projection='polar')
ax.bar(theta, radii, width=width, bottom=0.0, color=colors, alpha=0.5)
plt.show()
```

# Elevation and Year – Polar Bar



Longest axes – 2010  
Smalles Axes – 2012

# Country and Elevation



# Deaths and Country



# Thank You for Listening

Any Queries???