UNIVERSITY OF KARACHI

DATA ANALYSIS & VISUALIZATION ON REAL WORLD DATA SETS

PREPARED BY:
SHAYS AHMED KHAN
NOUMAN AHMED KHAN
MUSTUFA AHMED

DEGREE PROGRAM:

- BS IN FINANCIAL MATHEMATICS

NAME OF PROJECT:

- DATA ANALYSIS OF PUBG WEAPON STATS

COURCE NAME:

- PYTHON PROGRAMMING

DATED: JANUARY, 2022

SUBMITTED TO :
SYED UMAID AHMED

INDEX

1. INTRODUCTION TO DATA ANALYSIS

DATA ANALYSIS is defined as a process of cleaning, transforming, and modeling data to discover useful information for business decision-making. The purpose of Data Analysis is to extract useful information from data and taking the decision based upon the data analysis.

a. **WHY DATA ANALYSIS?** To grow your business even to grow in your life, sometimes all you need to do is Analysis!

If your business is not growing, then you have to look back and acknowledge your mistakes and make a plan again without repeating those mistakes. And even if your business is growing, then you have to look forward to making the business to grow more. All you need to do is analyze your business data and business processes.

b. DATA ANALYSIS TOOLS

R
Python
sas
Java
SQL
MATLAB

c. TYPES OF DATA ANALYSIS

There are several **types of Data Analysis** techniques that exist based on business and technology. However, the major Data Analysis methods are:

- Text Analysis
- Statistical Analysis
- Diagnostic Analysis
- Predictive Analysis
- Prescriptive Analysis

Text Analysis

Text Analysis is also referred to as Data Mining. It is one of the methods of data analysis to discover a pattern in large data sets using databases or data mining tools. It used to transform raw data into business information. Business Intelligence tools are present in the market which is used to take strategic business decisions. Overall it offers a way to extract and examine data and deriving patterns and finally interpretation of the data.

Statistical Analysis

Statistical Analysis shows "What happen?" by using past data in the form of dashboards. Statistical Analysis includes collection, Analysis, interpretation, presentation, and modeling of data. It analyses a set of data or a sample of data. There are two categories of this type of Analysis – Descriptive Analysis and Inferential Analysis.

Descriptive Analysis

Analyses complete data or a sample of summarized numerical data. It shows mean and deviation for continuous data whereas percentage and frequency for categorical data.

Inferential Analysis

Analyses sample from complete data. In this type of Analysis, you can find different conclusions from the same data by selecting different samples.

Diagnostic Analysis

Diagnostic Analysis shows "Why did it happen?" by finding the cause from the insight found in Statistical Analysis. This Analysis is useful to identify behavior patterns of data. If a new problem arrives in your business process, then you can look into this Analysis to find similar patterns of that problem. And it may have chances to use similar prescriptions for the new problems.

Predictive Analysis

Predictive Analysis shows "what is likely to happen" by using previous data. The simplest data analysis example is like if last year I bought two dresses based on my savings and if this year my salary is increasing double then I can buy four dresses. But of course it's not easy like this because you have to think about other circumstances like chances of prices of clothes is increased this year or maybe instead of dresses you want to buy a new bike, or you need to buy a house!

So here, this Analysis makes predictions about future outcomes based on current or past data. Forecasting is just an estimate. Its accuracy is based on how much detailed information you have and how much you dig in it.

Prescriptive Analysis

Prescriptive Analysis combines the insight from all previous Analysis to determine which action to take in a current problem or decision. Most data-driven companies are utilizing Prescriptive Analysis because predictive and descriptive Analysis are not enough to improve data performance. Based on current situations and problems, they analyze the data and make decisions.

d. IMPORTANCE OF DATA ANALYSIS

Data analysis is important in research because it makes studying data a lot simpler and more accurate. It helps the researchers straightforwardly interpret the data so that researchers don't leave anything out that could help them derive insights from it.

Data analysis is a way to study and analyze huge amounts of data. Research often includes going through heaps of data, which is getting more and more for the researchers to handle with every passing minute.

Hence, data analysis knowledge is a huge edge for researchers in the current era, making them very efficient and productive.

2. DATA ANALYSIS PROCESS

The **Data Analysis Process** is nothing but gathering information by using a proper application or tool which allows you to explore the data and find a pattern in it. Based on that information and data, you can make decisions, or you can get ultimate conclusions.

a. **DATA REQUIREMENT** First of all, you have to think about why do you want to do this data analysis? All you need to find out the purpose or aim of doing the Analysis of data. You have to decide which type of data analysis you wanted to do! In this phase, you have to decide what to analyze and how to measure it, you have to understand why you are investigating and what measures you have to use to do this Analysis.

b. **DATA COLLECTION** After requirement gathering, you will get a clear idea about what things you have to measure and what should be your findings. Now it's time to collect your data based on requirements. Once you collect your data, remember that the collected data must be processed or organized for Analysis. As you collected data from various sources, you must have to keep a log with a collection date and source of the data.

c. **DATA CLEANING** Now whatever data is collected may not be useful or irrelevant to your aim of Analysis, hence it should be cleaned. The data which is collected may contain duplicate records, white spaces or errors. The data should be cleaned and error free. This phase must be done before Analysis because based on data cleaning, your output of Analysis will be closer to your expected outcome.

3. DATA VISUALIZATION

Data visualization is a field in data analysis that deals with visual representation of data. It graphically plots data and is an effective way to communicate inferences from data.

Using data visualization, we can get a visual summary of our data. With pictures, maps and graphs, the human mind has an easier time processing and understanding any given data. Data visualization plays a significant role in the representation of both small and large data sets, but it is especially useful when we have large data sets, in which it is impossible to see all of our data, let alone process and understand it manually.

a. DATA VISUALIZATION IN PYTHON

Python offers several plotting libraries, namely Matplotlib, Seaborn and many other such data visualization packages with different features for creating informative, customized, and appealing plots to present data in the most simple and effective way.

4. PYTHON

a. WHAT IS PYTHON?

Python is an object-oriented, high-level programming language.

Its built-in data structures and properties, combined with dynamic typing (meaning we don't have to declare the type of variable like in C or Java) and binding, make it ideal for application development and its use as a scripting language.

Python's simple syntax emphasizes readability and reduces programming maintenance.

Python is a programming language that can be used for a wide range of tasks such as software creation, web development, script writing, unlike HTML, CSS, and JavaScript.

b. DATA ANALYSIS WITH PYTHON

Python is a multi-functional, maximally interpreted programming language with several advantages that are often used to streamline massive, and complex data sets.
Python has a number of distinguishing characteristics that make it the best option for data analysis.

5. PYTHON LIBRARIES USED IN PROJECT

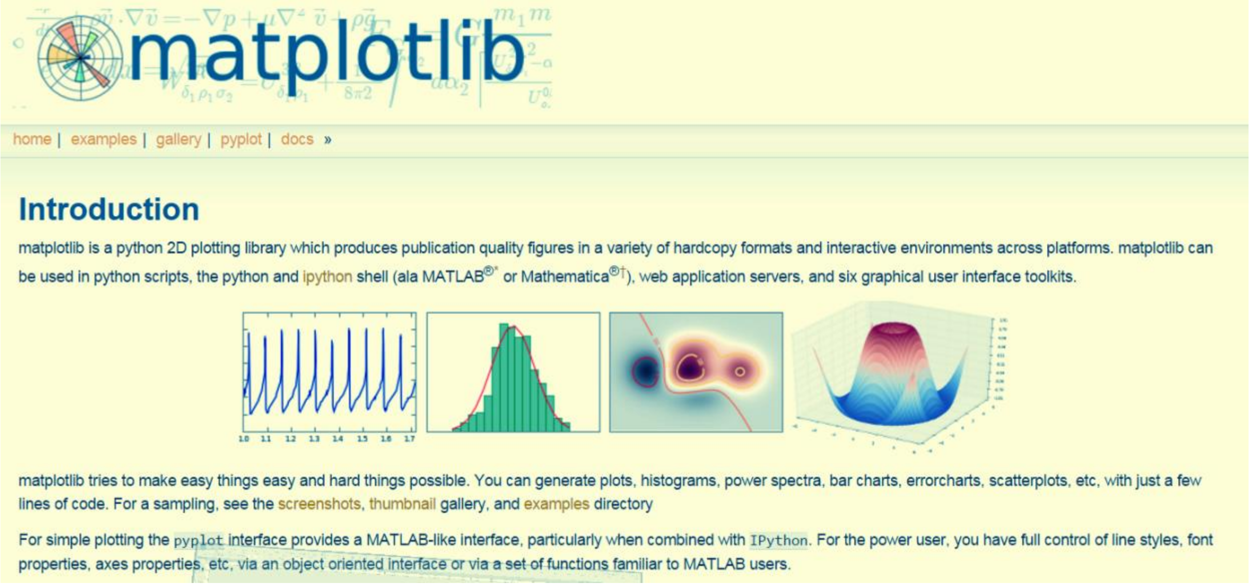
Python Libraries We know that a module is a file with some Python code, and a package is a directory for sub packages and modules. But the line between a package and a Python library is quite blurred.

A Python library is a reusable chunk of code that you may want to include in your projects.

Compared to languages like C++ or C, a Python libraries do not pertain to any specific context in Python. Here, a 'library' loosely describes a collection of core modules.

a. MATPLOTLIB

Matplotlib helps with data analyzing, and is a numerical plotting library. We talked about it in Python for Data Science.



Introduction

matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. matplotlib can be used in python scripts, the python and ipython shell (ala MATLAB®* or Mathematica®†), web application servers, and six graphical user interface toolkits.

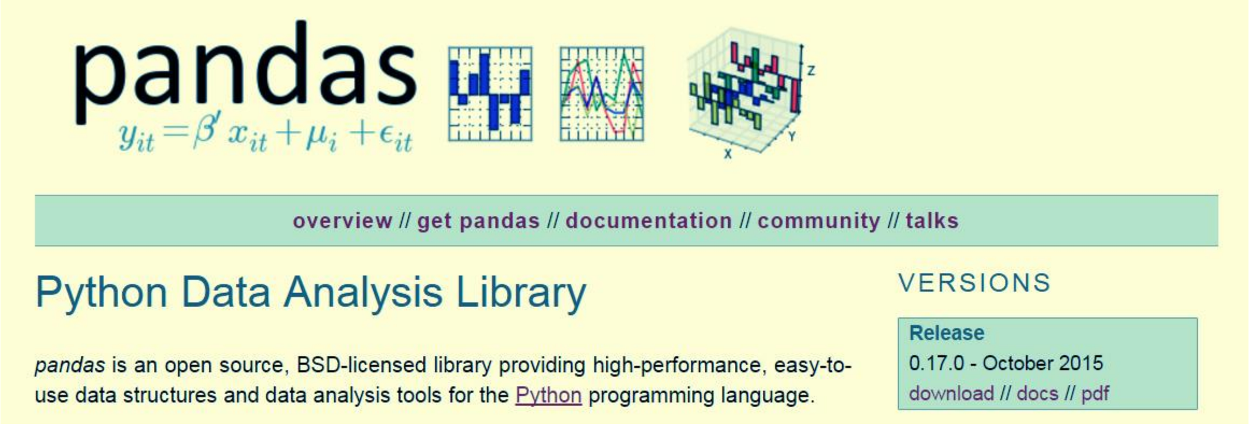
matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc, with just a few lines of code. For a sampling, see the screenshots, thumbnail gallery, and examples directory

For simple plotting the `pyplot` interface provides a MATLAB-like interface, particularly when combined with `IPython`. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

b. PANDAS

Like we’ve said before, Pandas is a must for data-science.

It provides fast, expressive, and flexible data structures to easily (and intuitively) work with structured (tabular, multidimensional, potentially heterogeneous) and time-series data.


pandas
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$
Z
Y
X
overview // get pandas // documentation // community // talks
VERSIONS
download // docs // pdf

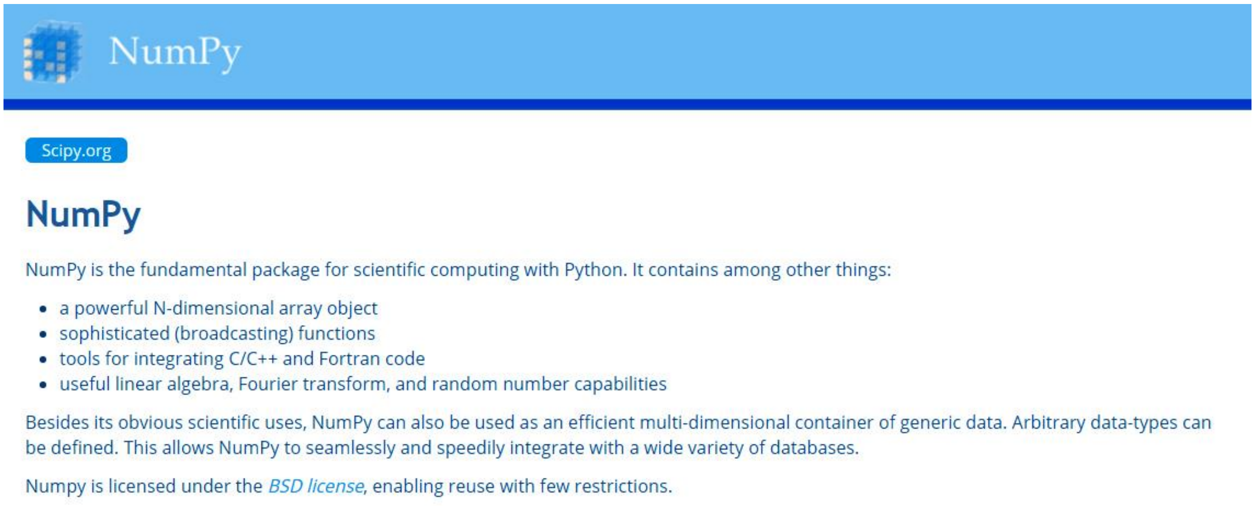
Python Data Analysis Library

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

Release
0.17.0 - October 2015

c. NUMPY

It has advanced math functions and a rudimentary scientific computing package.



NumPy

Scipy.org

NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

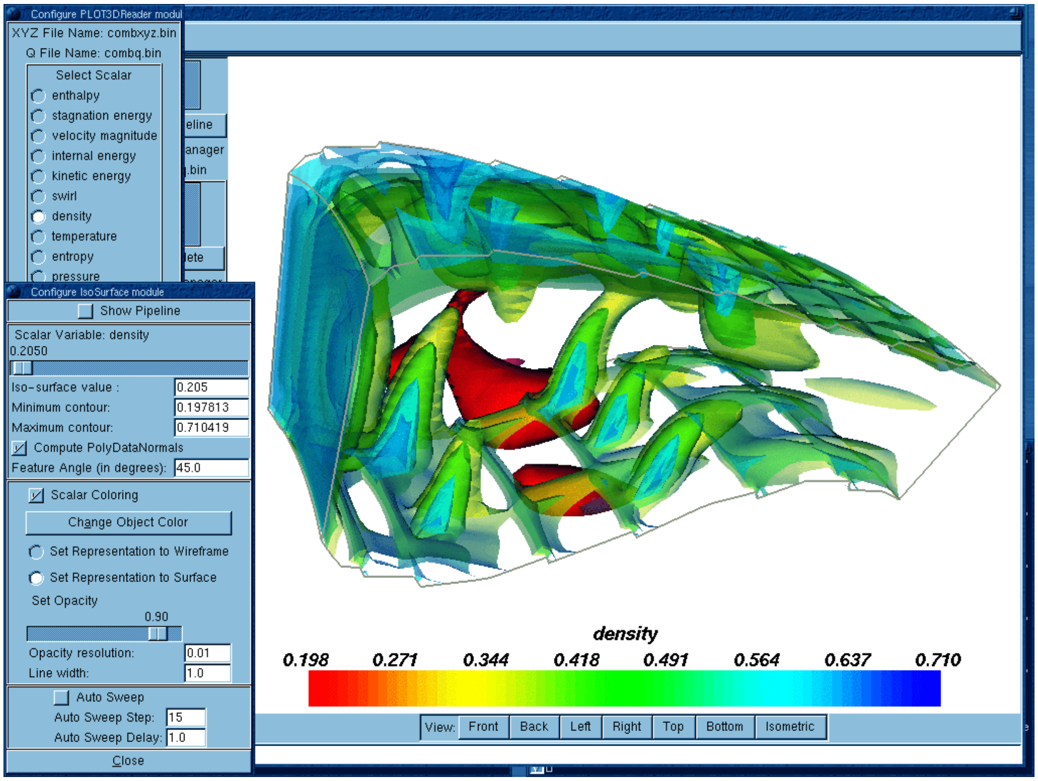
Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Numpy is licensed under the *BSD license*, enabling reuse with few restrictions.

d. SCIPY

Next up is SciPy, one of the libraries we have been talking so much about. It has a number of user-friendly and efficient numerical routines.

These include routines for optimization and numerical integration.



Configure PLOT3DReader modul

XYZ File Name: combxyz.bin

Q File Name: combq.bin

Select Scalar
○ enthalpy
○ stagnation energy
○ velocity magnitude
○ internal energy
○ kinetic energy
○ swirl
● density
○ temperature
○ entropy
○ pressure

Configure IsoSurface module

□ Show Pipeline

Scalar Variable: density
0.2050

Iso-surface value : 0.205
Minimum contour: 0.197813
Maximum contour: 0.710419
☑ Compute PolyDataNormals
Feature Angle (in degrees): 45.0

☑ Scalar Coloring

○ Set Representation to Wireframe
● Set Representation to Surface

Set Opacity 0.90
Opacity resolution: 0.01
Line width: 1.0

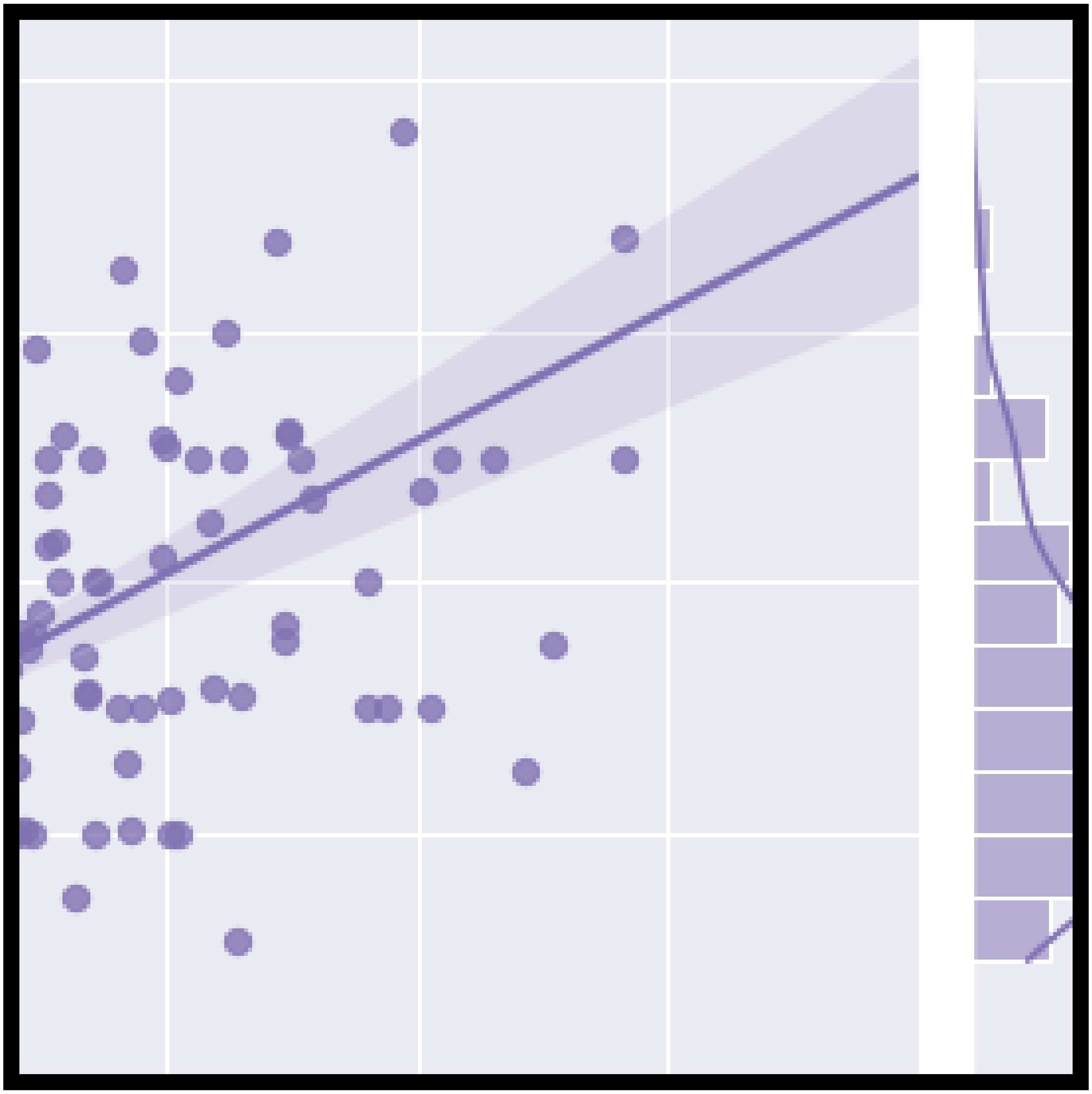
□ Auto Sweep
Auto Sweep Step: 15
Auto Sweep Delay: 1.0

e. SEABORN

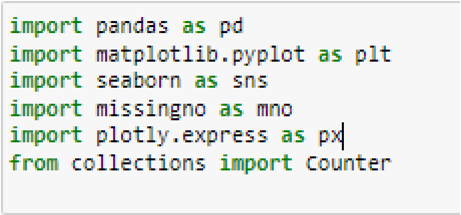
Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

For a brief introduction to the ideas behind the library, you can read the introductory notes or the paper. Visit the installation page to see how you can download the package and get started with it. You can browse the example gallery to see some of the things that you can do with seaborn, and then check out the tutorial or API reference to find out how.

To see the code or report a bug, please visit the GitHub repository. General support questions are most at home on stackoverflow or discourse, which have dedicated channels for seaborn.

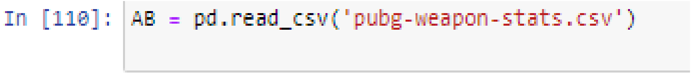


Step 1 : Importing Necessary Libraries



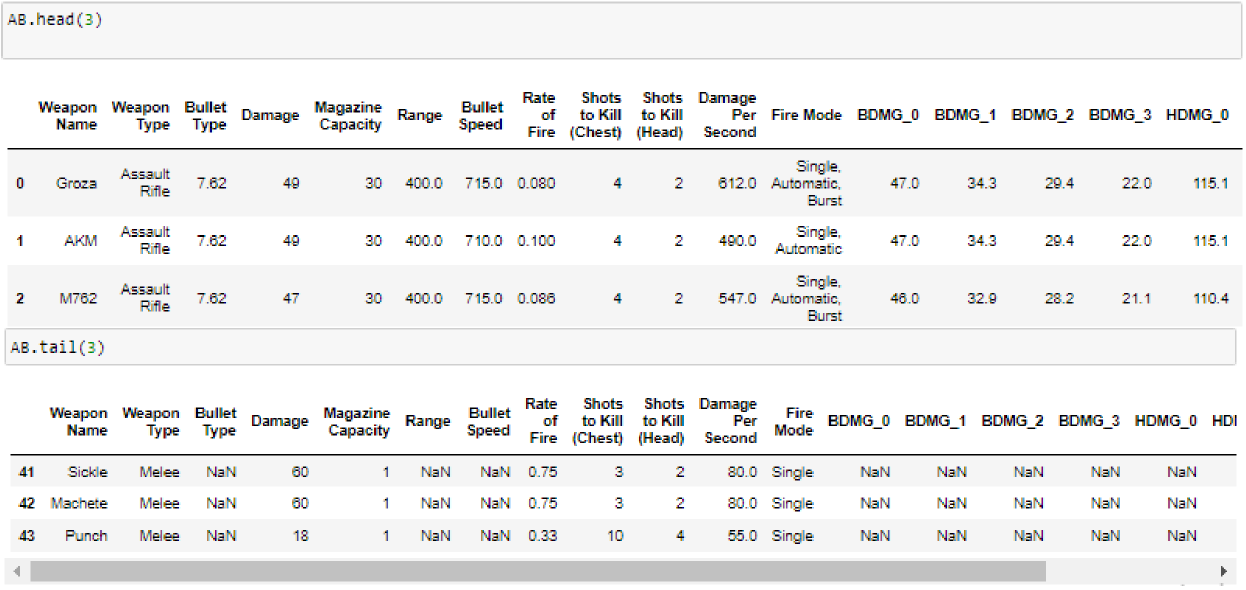
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as mno
import plotly.express as px
from collections import Counter
```

Step 2 : PD read csv read a comma-separated values (csv) **file into DataFrame.** Also supports optionally iterating or breaking of the file into chunks



```
In [110]: AB = pd.read_csv('pubg-weapon-stats.csv')
```

Step 3: Read First & Last Three Rows of Files



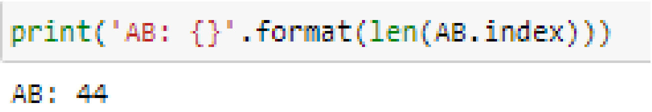
```
AB.head(3)
```

	Weapon Name	Weapon Type	Bullet Type	Damage	Magazine Capacity	Range	Bullet Speed	Rate of Fire	Shots to Kill (Chest)	Shots to Kill (Head)	Damage Per Second	Fire Mode	BDMG_0	BDMG_1	BDMG_2	BDMG_3	HDMG_0
0	Groza	Assault Rifle	7.62	49	30	400.0	715.0	0.080	4	2	612.0	Single, Automatic, Burst	47.0	34.3	29.4	22.0	115.1
1	AKM	Assault Rifle	7.62	49	30	400.0	710.0	0.100	4	2	490.0	Single, Automatic	47.0	34.3	29.4	22.0	115.1
2	M762	Assault Rifle	7.62	47	30	400.0	715.0	0.086	4	2	547.0	Single, Automatic, Burst	46.0	32.9	28.2	21.1	110.4

```
AB.tail(3)
```

	Weapon Name	Weapon Type	Bullet Type	Damage	Magazine Capacity	Range	Bullet Speed	Rate of Fire	Shots to Kill (Chest)	Shots to Kill (Head)	Damage Per Second	Fire Mode	BDMG_0	BDMG_1	BDMG_2	BDMG_3	HDMG_0	HDI
41	Sickle	Melee	NaN	60	1	NaN	NaN	0.75	3	2	80.0	Single	NaN	NaN	NaN	NaN	NaN	
42	Machete	Melee	NaN	60	1	NaN	NaN	0.75	3	2	80.0	Single	NaN	NaN	NaN	NaN	NaN	
43	Punch	Melee	NaN	18	1	NaN	NaN	0.33	10	4	55.0	Single	NaN	NaN	NaN	NaN	NaN	

Step 4 : Find The No. of Rows

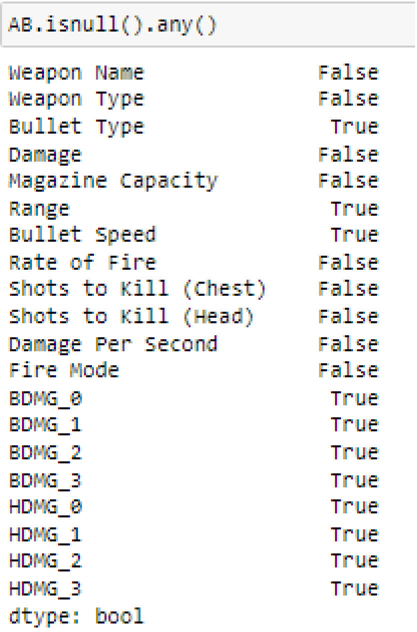


```
print('AB: {}'.format(len(AB.index)))
```

```
AB: 44
```

Step 5 : Find The Null Values or Missing No.

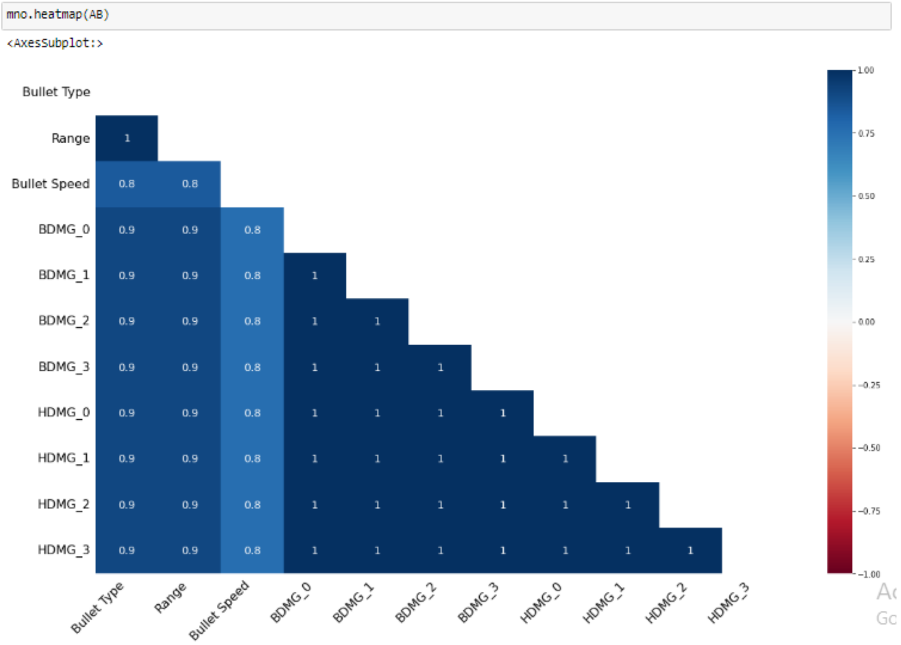
Simple Way



```
AB.isnull().any()
```

```
Weapon Name                 False
Weapon Type                 False
Bullet Type                  True
Damage                      False
Magazine Capacity           False
Range                        True
Bullet Speed                 True
Rate of Fire                False
Shots to Kill (Chest)       False
Shots to Kill (Head)        False
Damage Per Second           False
Fire Mode                   False
BDMG_0                       True
BDMG_1                       True
BDMG_2                       True
BDMG_3                       True
HDMG_0                       True
HDMG_1                       True
HDMG_2                       True
HDMG_3                       True
dtype: bool
```

By using missingno Library

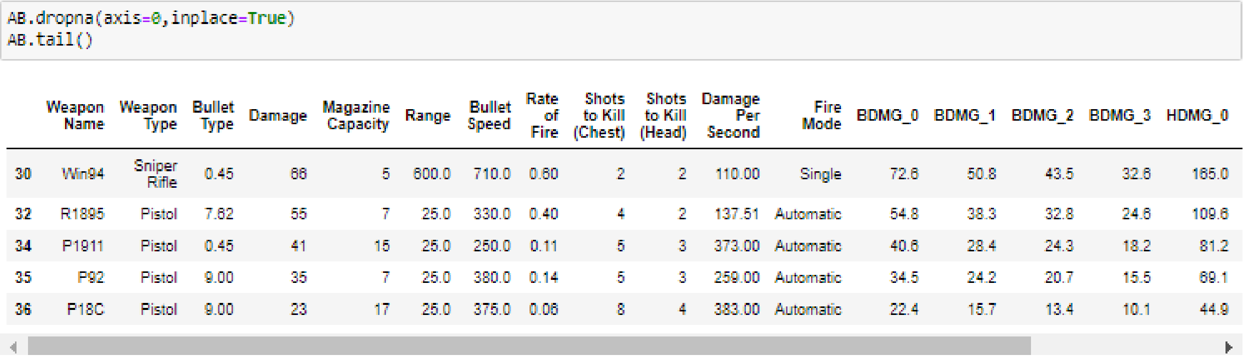


```
mno.heatmap(AB)
```

```
<AxesSubplot:>
```

	Bullet Type	Range	Bullet Speed	BDMG_0	BDMG_1	BDMG_2	BDMG_3	HDMG_0	HDMG_1	HDMG_2	HDMG_3
Bullet Type											
Range	1										
Bullet Speed	0.8	0.8									
BDMG_0	0.9	0.9	0.8								
BDMG_1	0.9	0.9	0.8	1							
BDMG_2	0.9	0.9	0.8	1	1						
BDMG_3	0.9	0.9	0.8	1	1	1					
HDMG_0	0.9	0.9	0.8	1	1	1	1				
HDMG_1	0.9	0.9	0.8	1	1	1	1	1			
HDMG_2	0.9	0.9	0.8	1	1	1	1	1	1		
HDMG_3	0.9	0.9	0.8	1	1	1	1	1	1	1	

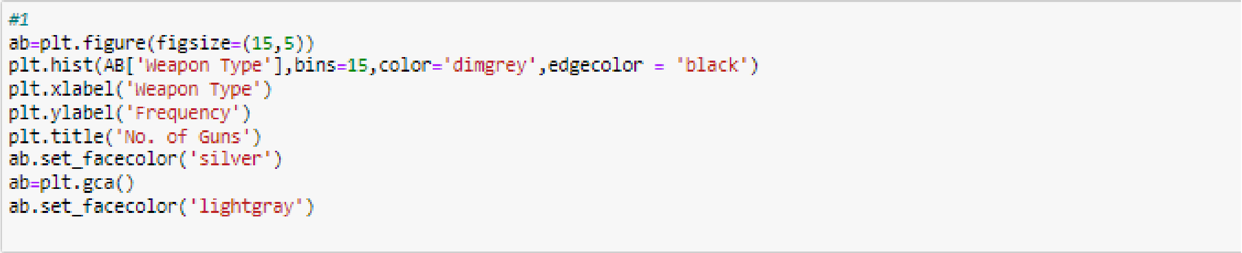
Step 6 : Dropped all the rows containing null values because it effects Data cleaning process due to more Null values present in small DataSet.
After this rows are dropped and No. of rows now is thirty six.



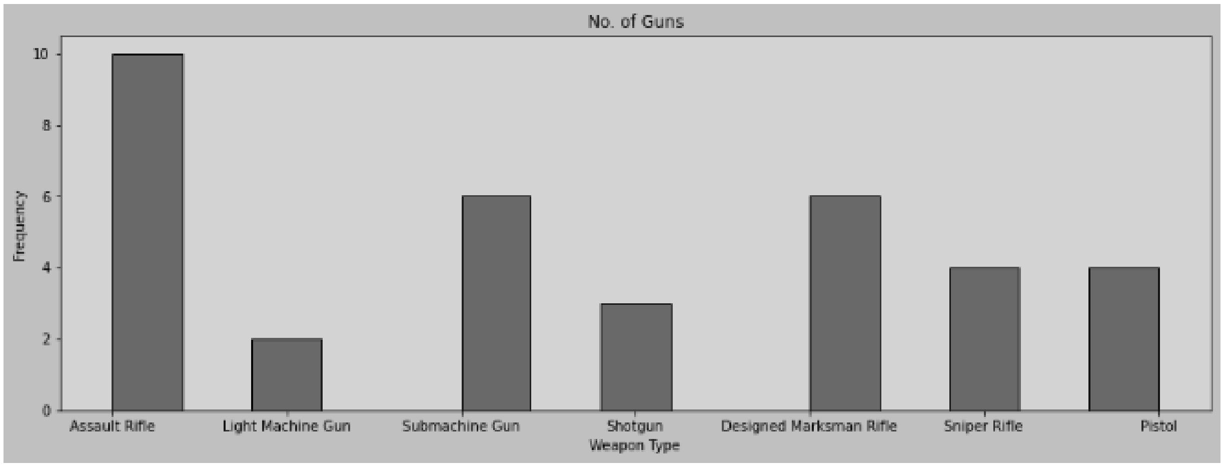
```
AB.dropna(axis=0,inplace=True)
AB.tail()
```

	Weapon Name	Weapon Type	Bullet Type	Damage	Magazine Capacity	Range	Bullet Speed	Rate of Fire	Shots to Kill (Chest)	Shots to Kill (Head)	Damage Per Second	Fire Mode	BDMG_0	BDMG_1	BDMG_2	BDMG_3	HDMG_0
30	Win94	Sniper Rifle	0.45	66	5	600.0	710.0	0.60	2	2	110.00	Single	72.6	50.8	43.5	32.6	165.0
32	R1895	Pistol	7.62	55	7	25.0	330.0	0.40	4	2	137.51	Automatic	54.8	38.3	32.8	24.6	109.6
34	P1911	Pistol	0.45	41	15	25.0	250.0	0.11	5	3	373.00	Automatic	40.6	28.4	24.3	18.2	81.2
35	P92	Pistol	9.00	35	7	25.0	380.0	0.14	5	3	259.00	Automatic	34.5	24.2	20.7	15.5	69.1
36	P18C	Pistol	9.00	23	17	25.0	375.0	0.06	8	4	383.00	Automatic	22.4	15.7	13.4	10.1	44.9

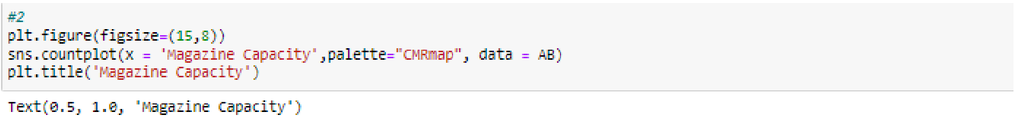
Step 7 : The following Plot shows different types of Weapons And Frequency of their availability : From the Plot we can say Assualt Riffles are more Frequently available.



```
#1
ab=plt.figure(figsize=(15,5))
plt.hist(AB['Weapon Type'],bins=15,color='dimgrey',edgecolor = 'black')
plt.xlabel('Weapon Type')
plt.ylabel('Frequency')
plt.title('No. of Guns')
ab.set_facecolor('silver')
ab=plt.gca()
ab.set_facecolor('lightgray')
```

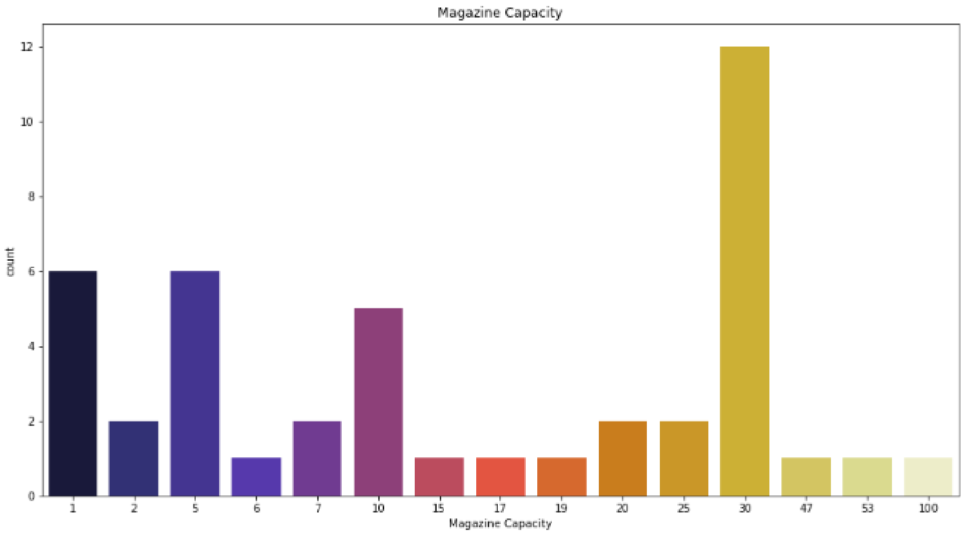


Step 8 : The following Plot shows the Magazine Capacity of each Weapon Type : From the Plot we can analyze 30 Magazine Guns are more in number

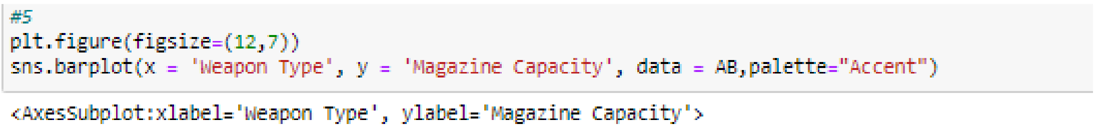


```
#2
plt.figure(figsize=(15,8))
sns.countplot(x = 'Magazine Capacity',palette="CMRmap", data = AB)
plt.title('Magazine Capacity')
```

```
Text(0.5, 1.0, 'Magazine Capacity')
```

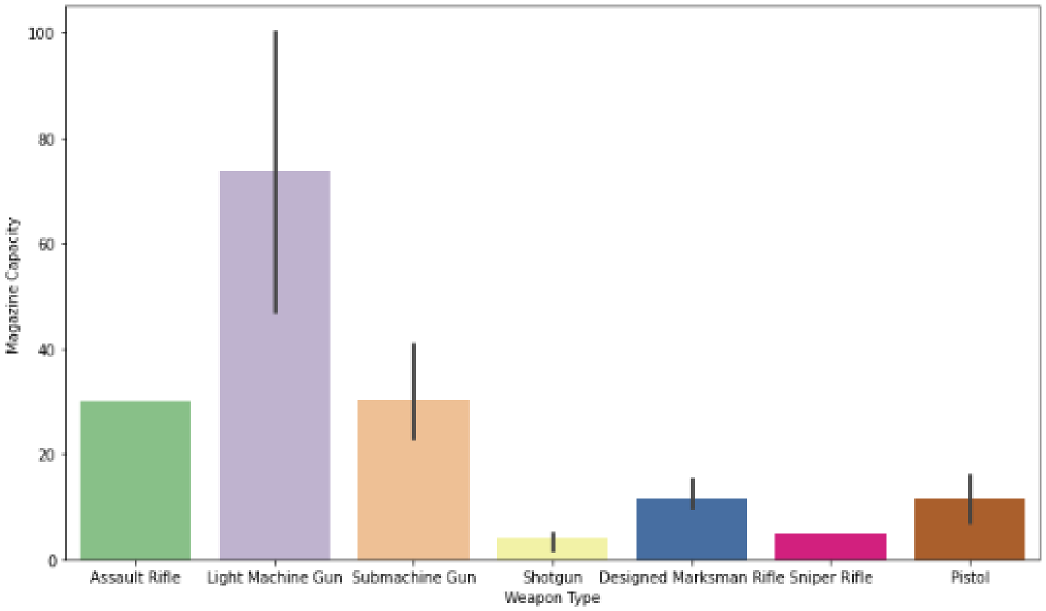


Step 9 : Weapon Type Vs Magazine Capacity



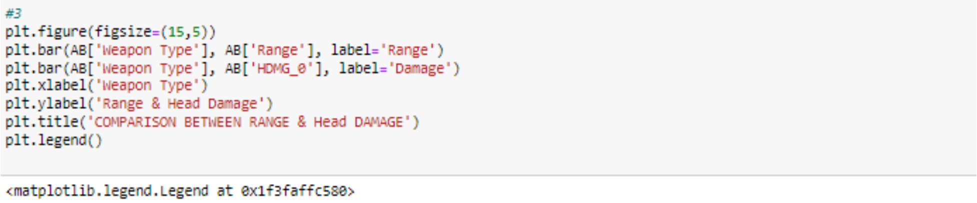
```
#5
plt.figure(figsize=(12,7))
sns.barplot(x = 'Weapon Type', y = 'Magazine Capacity', data = AB,palette="Accent")
```

```
<AxesSubplot:xlabel='Weapon Type', ylabel='Magazine Capacity'>
```



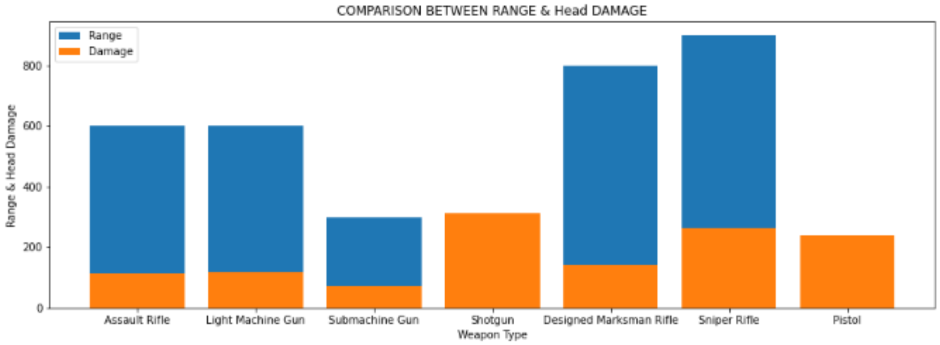
Step 10 : The following Plot shows the comparision between Range and Damage of different Weapon Type (Body Damage without Helmet) .
Long Range and High Damage Guns are more Efficient and are used more Frequently : From this we can visualize Assault Riffle is more Consistent and used more Frequently.

HEAD DAMAGE VS RANGE

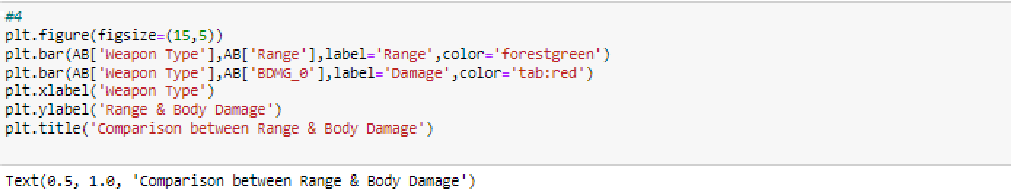


```
#3
plt.figure(figsize=(15,5))
plt.bar(AB['Weapon Type'], AB['Range'], label='Range')
plt.bar(AB['Weapon Type'], AB['HDMG_0'], label='Damage')
plt.xlabel('Weapon Type')
plt.ylabel('Range & Head Damage')
plt.title('COMPARISON BETWEEN RANGE & Head DAMAGE')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x1f3faffc580>
```

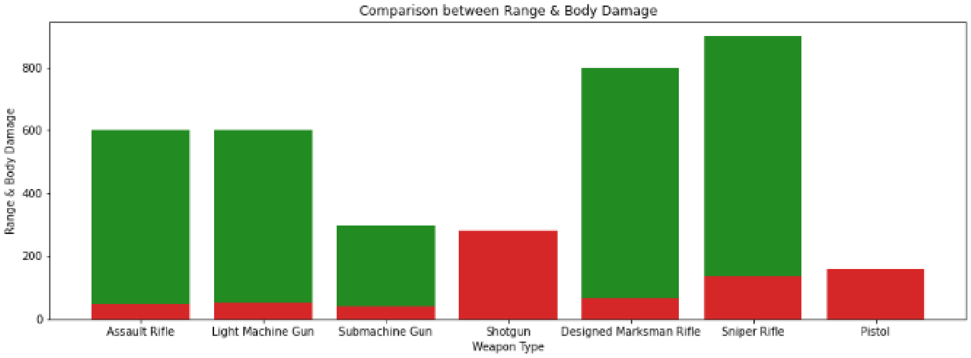


BODY DAMAGE VS RANGE

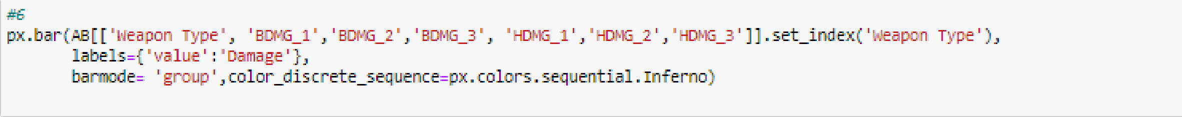


```
#4
plt.figure(figsize=(15,5))
plt.bar(AB['Weapon Type'],AB['Range'],label='Range',color='forestgreen')
plt.bar(AB['Weapon Type'],AB['BDMG_0'],label='Damage',color='tab:red')
plt.xlabel('Weapon Type')
plt.ylabel('Range & Body Damage')
plt.title('Comparison between Range & Body Damage')
```

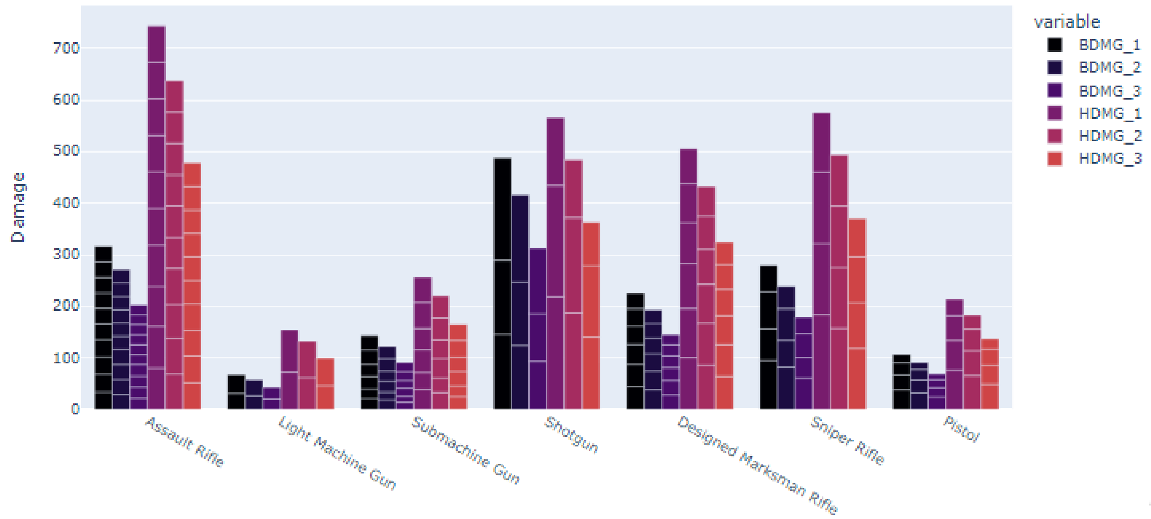
```
Text(0.5, 1.0, 'Comparison between Range & Body Damage')
```



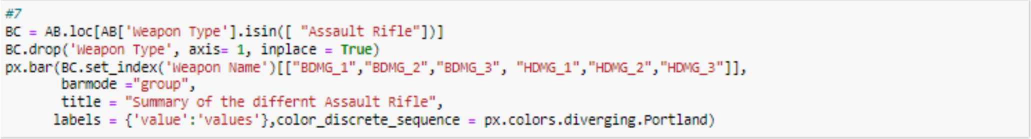
Step 11 : Weapon Type & their Damages (1,2&3 BODYSHOT and 1,2&3 HEADSHOT) but in this graph we didn't recognize it correctly.



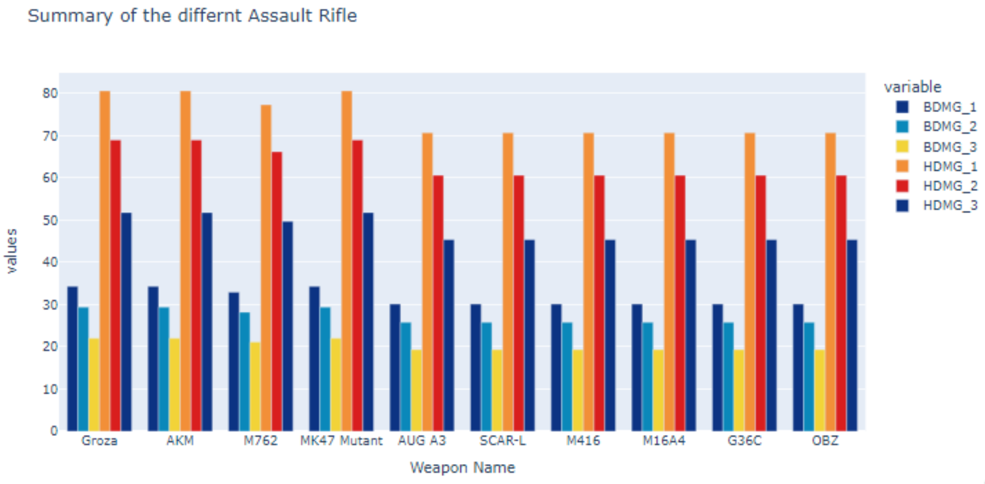
```
#6
px.bar(AB[['Weapon Type', 'BDMG_1','BDMG_2','BDMG_3', 'HDMG_1','HDMG_2','HDMG_3']].set_index('Weapon Type'),
       labels={'value':'Damage'},
       barmode= 'group',color_discrete_sequence=px.colors.sequential.Inferno)
```



Assualt Riflle & their Damages (1,2&3 BODYSHOT and 1,2&3 HEADSHOT)



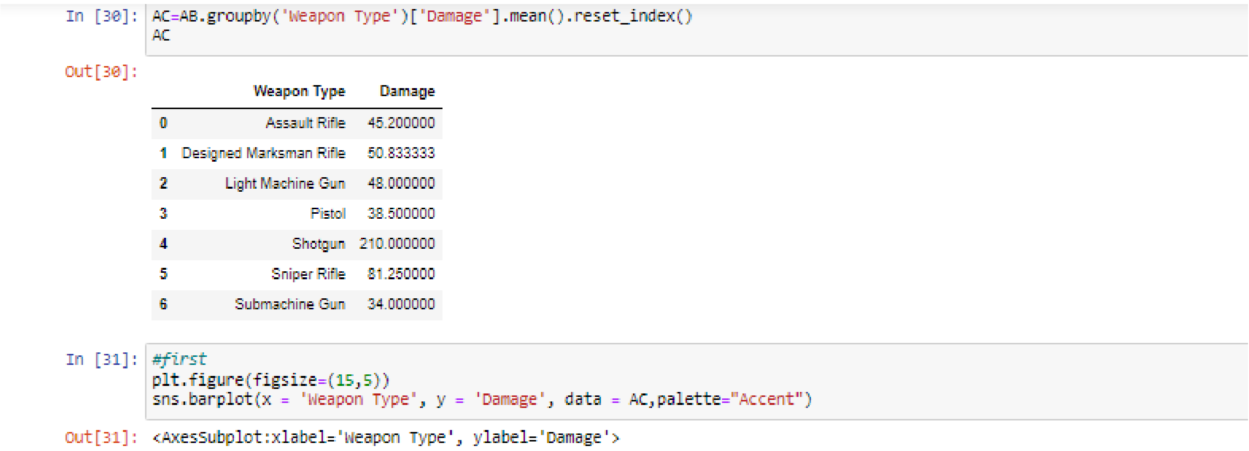
```
#7
BC = AB.loc[AB['Weapon Type'].isin([ "Assault Rifle"])]
BC.drop('Weapon Type', axis= 1, inplace = True)
px.bar(BC.set_index('Weapon Name')[["BDMG_1","BDMG_2","BDMG_3", "HDMG_1","HDMG_2","HDMG_3"]],
       barmode ="group",
       title = "Summary of the differnt Assault Rifle",
      labels = {'value':'values'},color_discrete_sequence = px.colors.diverging.Portland)
```



In this way we can take out all Weapon Type & their Damages in Accurate Way.

First for Weapon Type we Visualize each of the following individually :
1) Damage
2) Range
3) Bullet Speed
4) Rate of Fire
Then we Visualize any Two Categories of Weapon Types which are more Efficient.

Step 12 : Weapon Type VS Damage

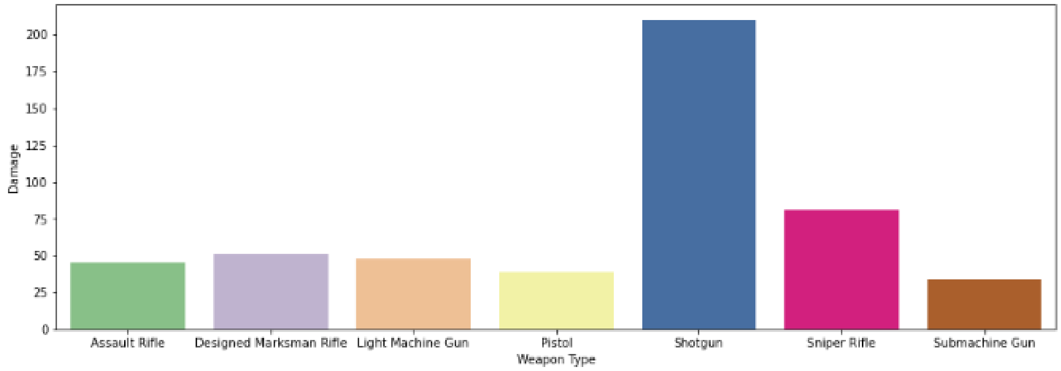


```
In [30]: AC=AB.groupby('Weapon Type')['Damage'].mean().reset_index()
         AC
```

Out[30]:

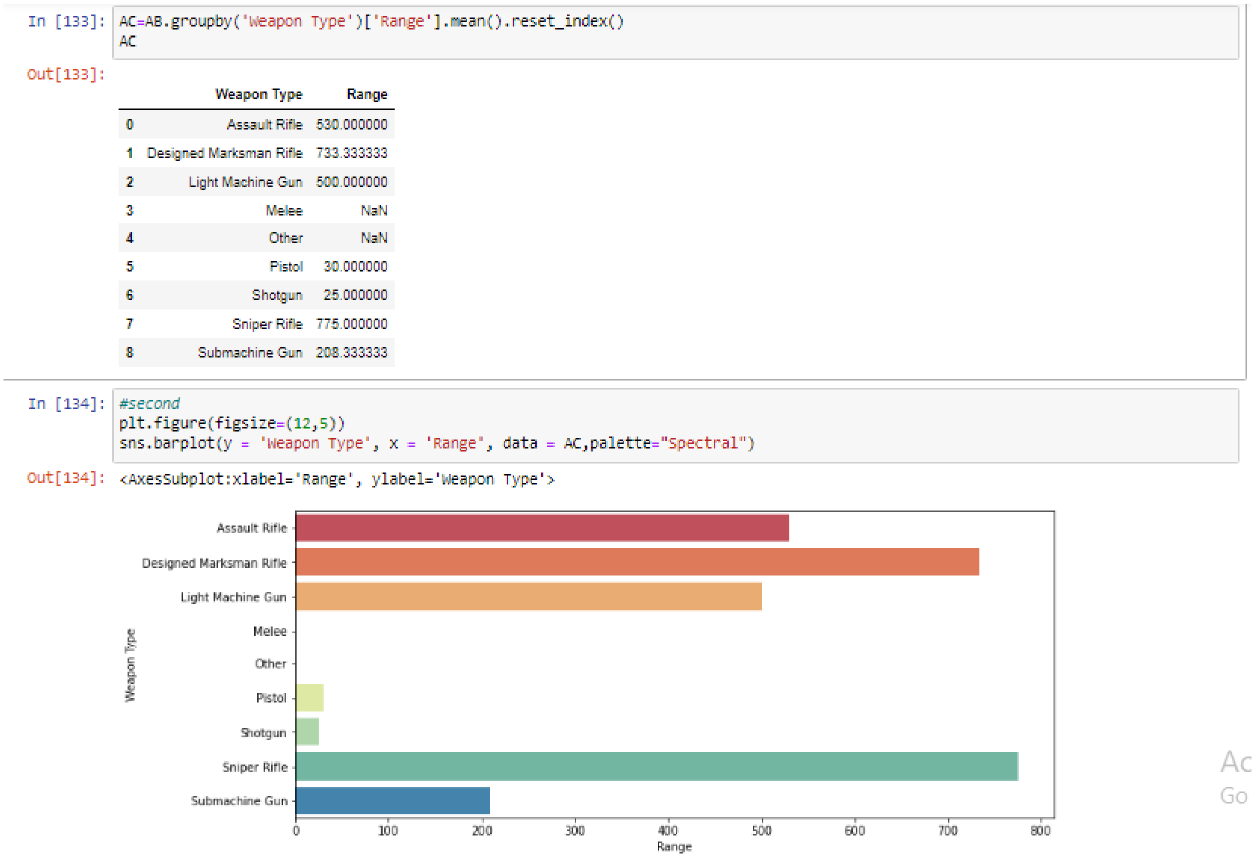
	Weapon Type	Damage
0	Assault Rifle	45.200000
1	Designed Marksman Rifle	50.833333
2	Light Machine Gun	48.000000
3	Pistol	38.500000
4	Shotgun	210.000000
5	Sniper Rifle	81.250000
6	Submachine Gun	34.000000

```
In [31]: #first
         plt.figure(figsize=(15,5))
         sns.barplot(x = 'Weapon Type', y = 'Damage', data = AC,palette="Accent")

Out[31]: <AxesSubplot:xlabel='Weapon Type', ylabel='Damage'>
```



The following Plot shows that SHOTGUN has More Damage then other Weapons.

Step 13 : Weapon Type VS Range



```
In [133]: AC=AB.groupby('Weapon Type')['Range'].mean().reset_index()
          AC
Out[133]:
```

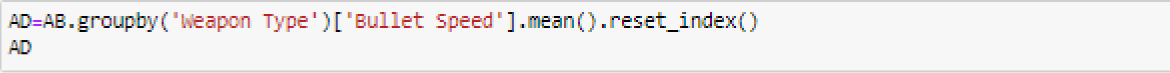
	Weapon Type	Range
0	Assault Rifle	530.000000
1	Designed Marksman Rifle	733.333333
2	Light Machine Gun	500.000000
3	Melee	NaN
4	Other	NaN
5	Pistol	30.000000
6	Shotgun	25.000000
7	Sniper Rifle	775.000000
8	Submachine Gun	208.333333

```
In [134]: #second
          plt.figure(figsize=(12,5))
          sns.barplot(y = 'Weapon Type', x = 'Range', data = AC,palette="Spectral")
Out[134]: <AxesSubplot:xlabel='Range', ylabel='Weapon Type'>
```

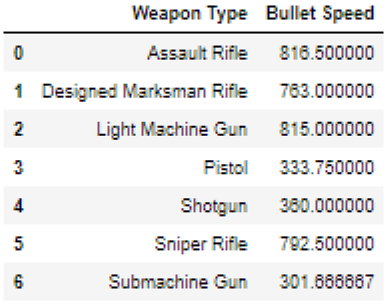
Ac
Go

The following Plot shows that SNIPER RIFLE has more Range the other Weapons .

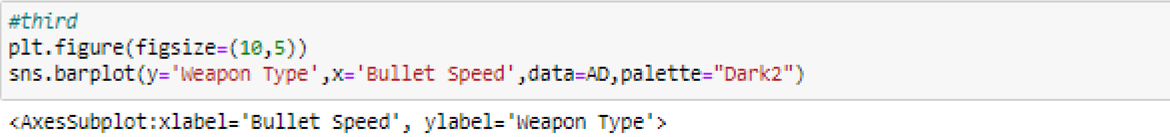
Step 14 : Weapon Type VS Bullet Speed



```
AD=AB.groupby('Weapon Type')['Bullet Speed'].mean().reset_index()
AD
```

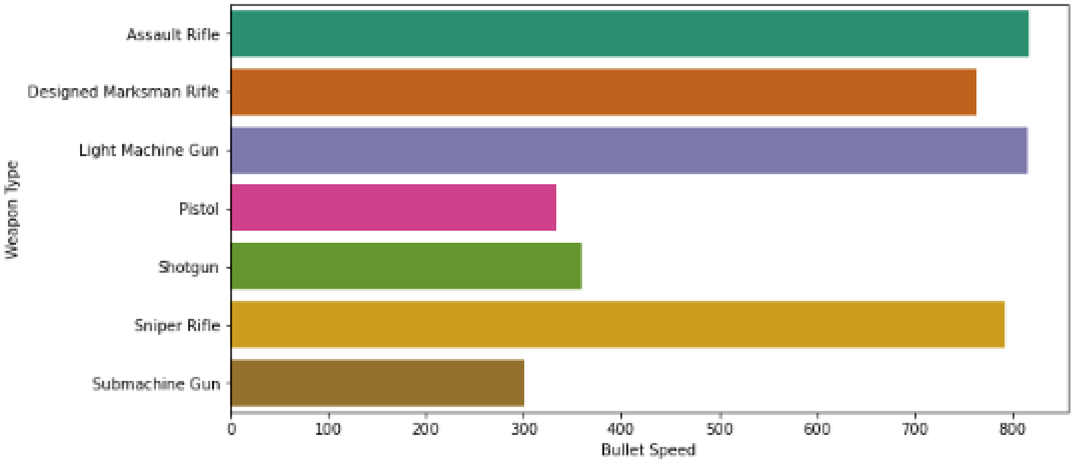


	Weapon Type	Bullet Speed
0	Assault Rifle	816.500000
1	Designed Marksman Rifle	763.000000
2	Light Machine Gun	815.000000
3	Pistol	333.750000
4	Shotgun	360.000000
5	Sniper Rifle	792.500000
6	Submachine Gun	301.666667



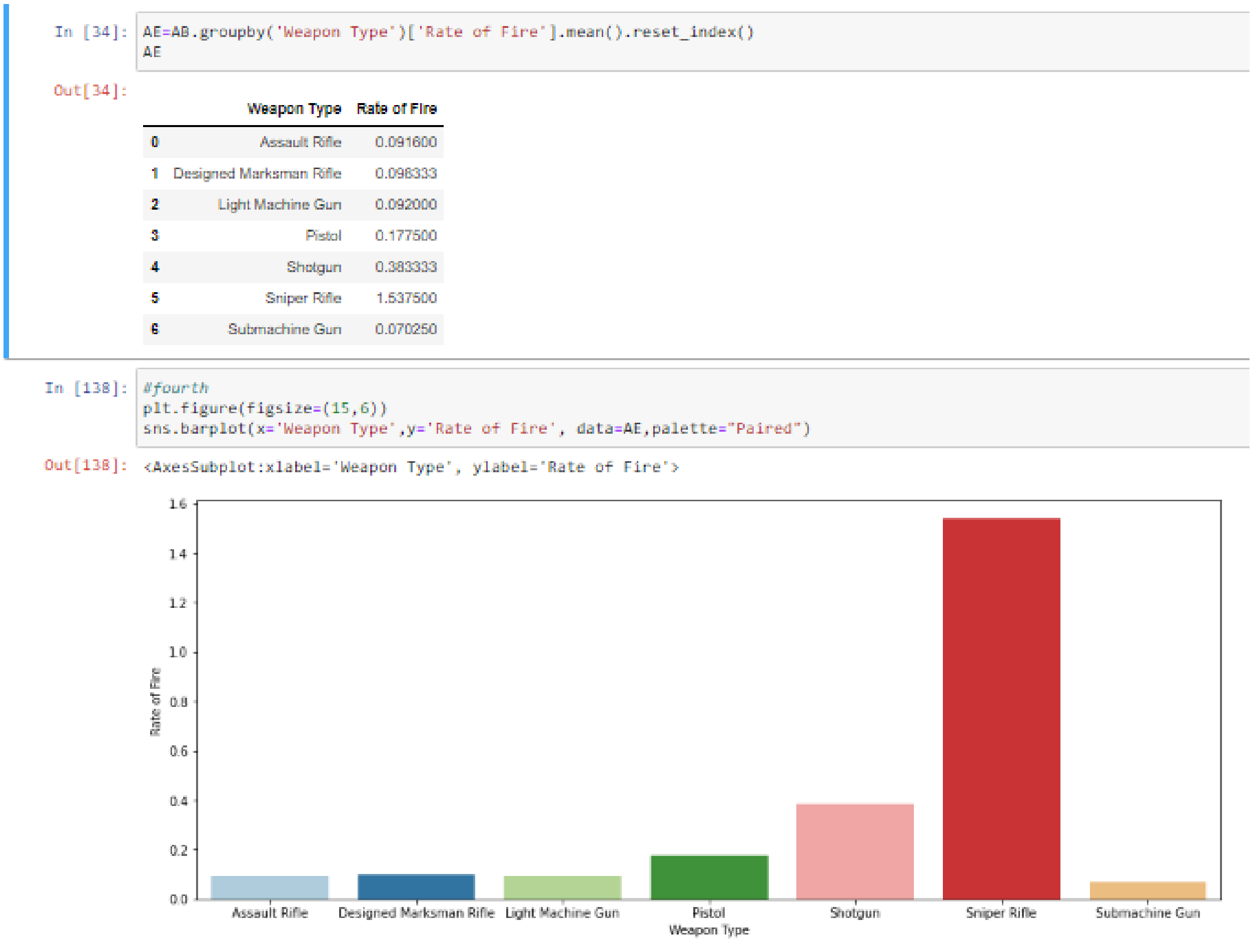
```
#third
plt.figure(figsize=(10,5))
sns.barplot(y='Weapon Type',x='Bullet Speed',data=AD,palette="Dark2")
```

```
<AxesSubplot:xlabel='Bullet Speed', ylabel='Weapon Type'>
```



The following Plot shows that Assault Rifle & Light Machine Gun has almost same Bullet Speed.

Step 15 : Weapon Type VS Rate of Fire



```
In [34]: AE=AB.groupby('Weapon Type')['Rate of Fire'].mean().reset_index()
         AE
```

Out[34]:

	Weapon Type	Rate of Fire
0	Assault Rifle	0.091600
1	Designed Marksman Rifle	0.098333
2	Light Machine Gun	0.092000
3	Pistol	0.177500
4	Shotgun	0.383333
5	Sniper Rifle	1.537500
6	Submachine Gun	0.070250

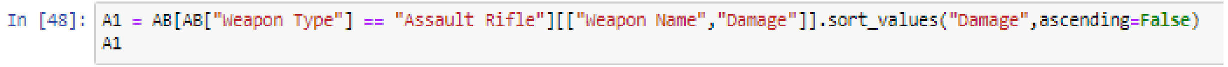
```
In [138]: #fourth
          plt.figure(figsize=(15,6))
          sns.barplot(x='Weapon Type',y='Rate of Fire', data=AE,palette="Paired")

Out[138]: <AxesSubplot:xlabel='Weapon Type', ylabel='Rate of Fire'>
```

The Following Plot shows that SNIPER RIFLE has more Rate of Fire

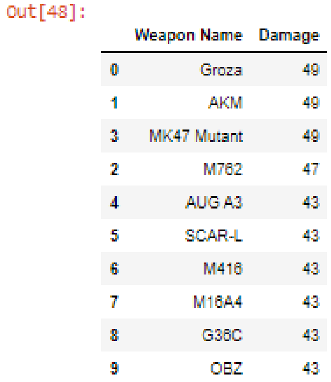
By Comparing all four Categories together, from all the above Analysis and Visualization we can say that :
Assualt Rifle + Sniper Rifle is the Best Combo of Guns and is more Efficient. So lets Visualize them Individually for the same 4 different Categories.

Step 16 :

1) For DAMAGE
   a) Assault Rifle Visualization

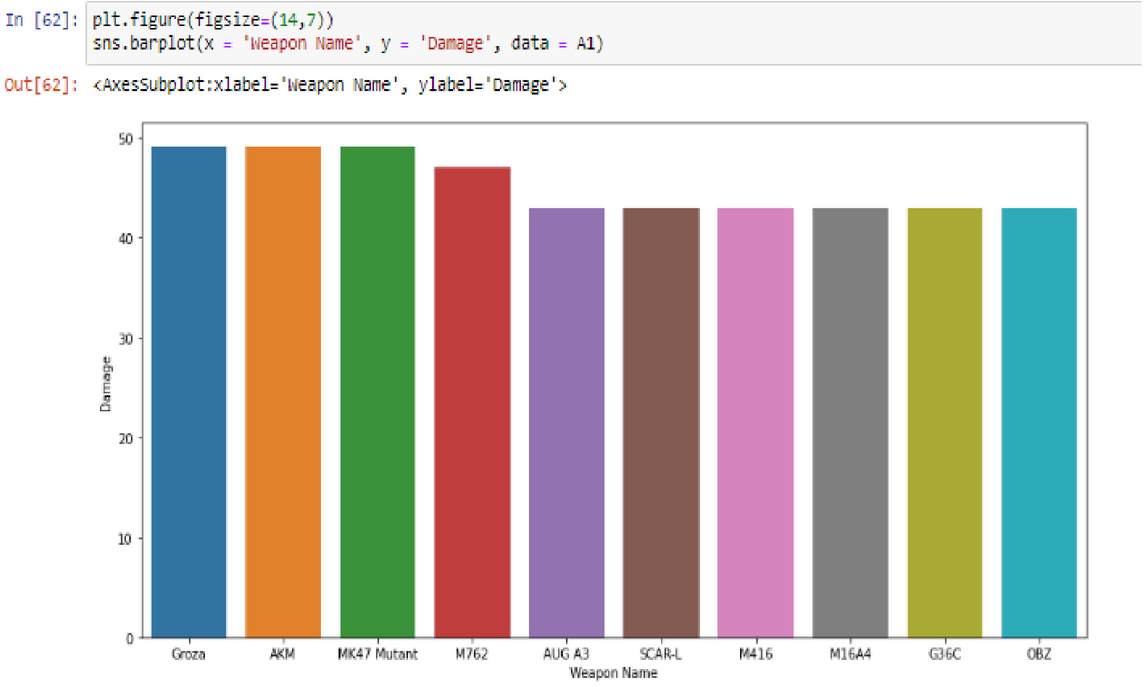


```
In [48]: A1 = AB[AB["Weapon Type"] == "Assault Rifle"][["Weapon Name","Damage"]].sort_values("Damage",ascending=False)
         A1
```



Out[48]:

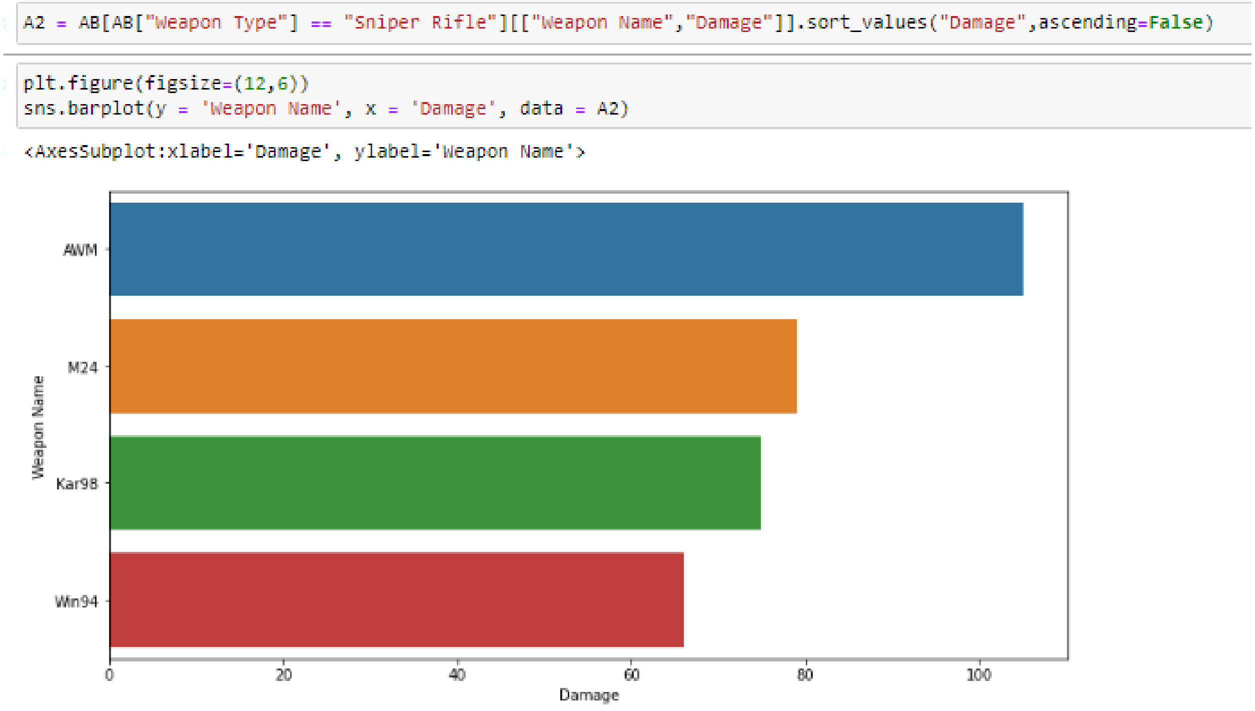
	Weapon Name	Damage
0	Groza	49
1	AKM	49
3	MK47 Mutant	49
2	M762	47
4	AUG A3	43
5	SCAR-L	43
6	M416	43
7	M16A4	43
8	G36C	43
9	OBZ	43



```
In [62]: plt.figure(figsize=(14,7))
         sns.barplot(x = 'Weapon Name', y = 'Damage', data = A1)

Out[62]: <AxesSubplot:xlabel='Weapon Name', ylabel='Damage'>
```

The Following Plot shows tough completion between GROZA & AKM , and MK47 Mutant is Burst Weapon not Auto .

b) Sniper Rifle Visualization



```
A2 = AB[AB["Weapon Type"] == "Sniper Rifle"][["Weapon Name","Damage"]].sort_values("Damage",ascending=False)
```

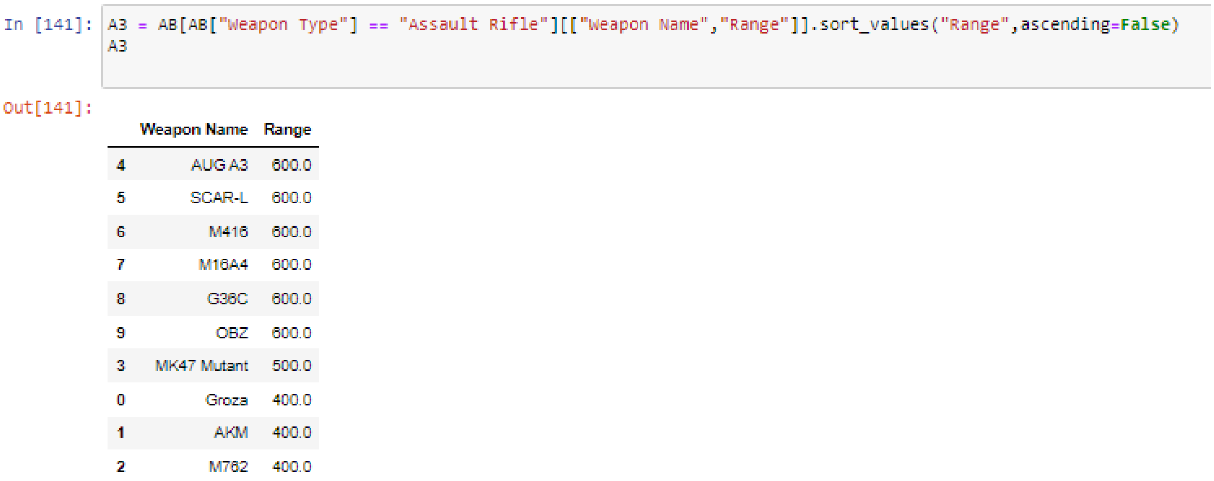
```
plt.figure(figsize=(12,6))
sns.barplot(y = 'Weapon Name', x = 'Damage', data = A2)
```

```
<AxesSubplot:xlabel='Damage', ylabel='Weapon Name'>
```

The following Plot shows that AWM has more then other three Sniper.

Step 17 :

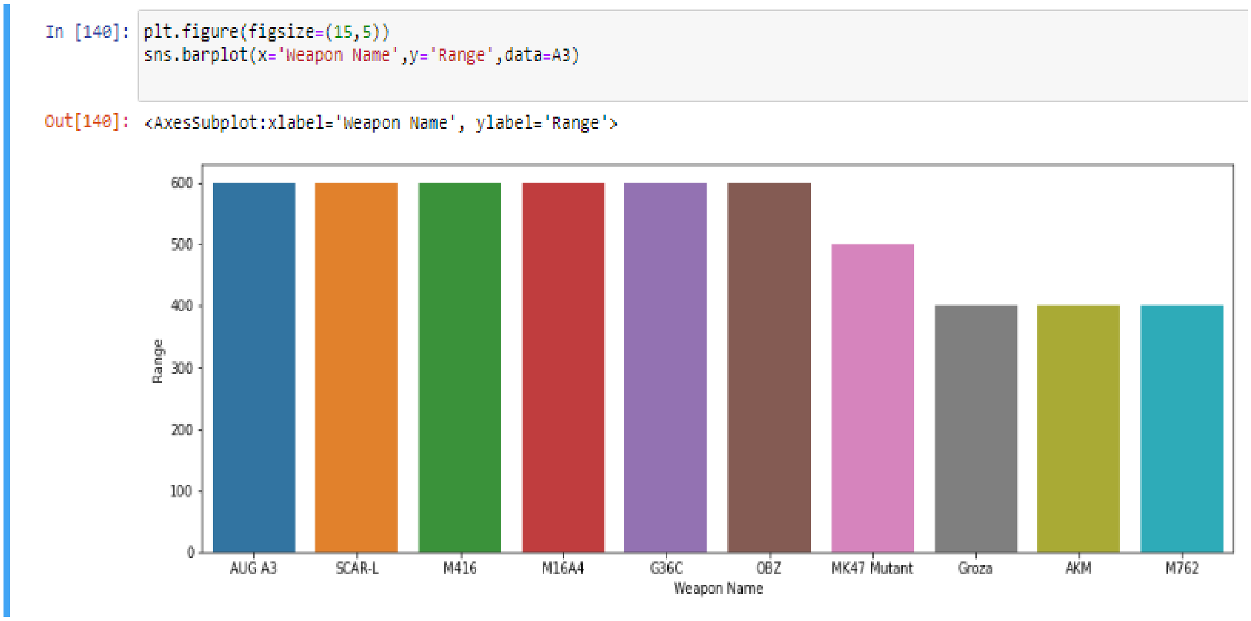
3) Range
   a) Assault Rifle Visualization



```
In [141]: A3 = AB[AB["Weapon Type"] == "Assault Rifle"][["Weapon Name","Range"]].sort_values("Range",ascending=False)
          A3
```

Out[141]:

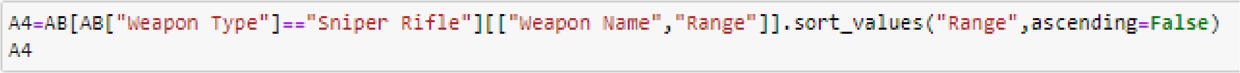
	Weapon Name	Range
4	AUG A3	600.0
5	SCAR-L	600.0
6	M416	600.0
7	M16A4	600.0
8	G36C	600.0
9	OBZ	600.0
3	MK47 Mutant	500.0
0	Groza	400.0
1	AKM	400.0
2	M762	400.0



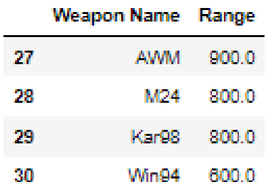
```
In [140]: plt.figure(figsize=(15,5))
          sns.barplot(x='Weapon Name',y='Range',data=A3)

Out[140]: <AxesSubplot:xlabel='Weapon Name', ylabel='Range'>
```

The Following Plot shows most of the guns have high & equal Range .

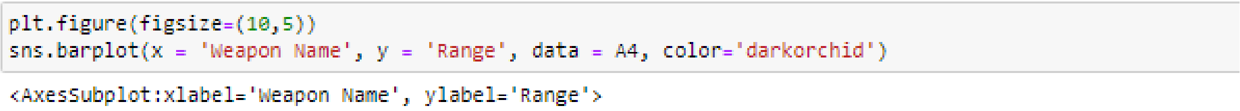
b) Sniper Rifle Visualization



```
A4=AB[AB["Weapon Type"]=="Sniper Rifle"][["Weapon Name","Range"]].sort_values("Range",ascending=False)
A4
```

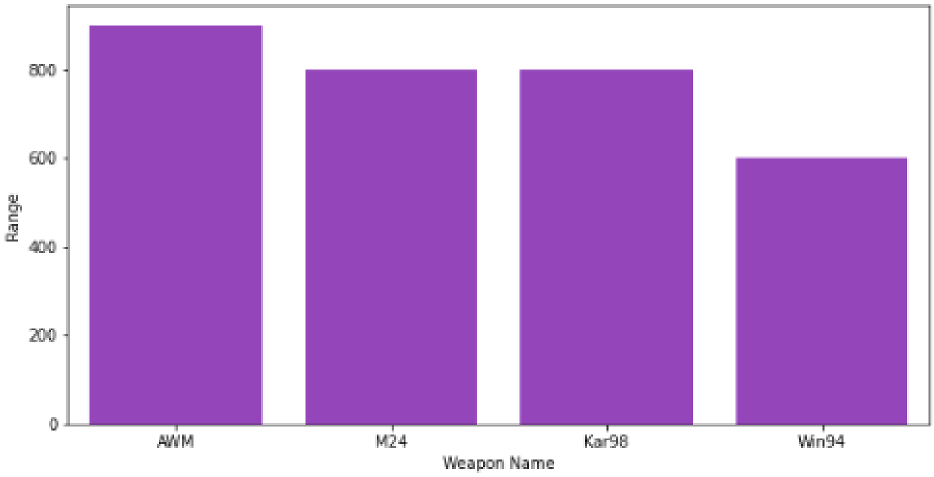


	Weapon Name	Range
27	AWM	900.0
28	M24	800.0
29	Kar98	800.0
30	Win94	600.0



```
plt.figure(figsize=(10,5))
sns.barplot(x = 'Weapon Name', y = 'Range', data = A4, color='darkorchid')
```

```
<AxesSubplot:xlabel='Weapon Name', ylabel='Range'>
```

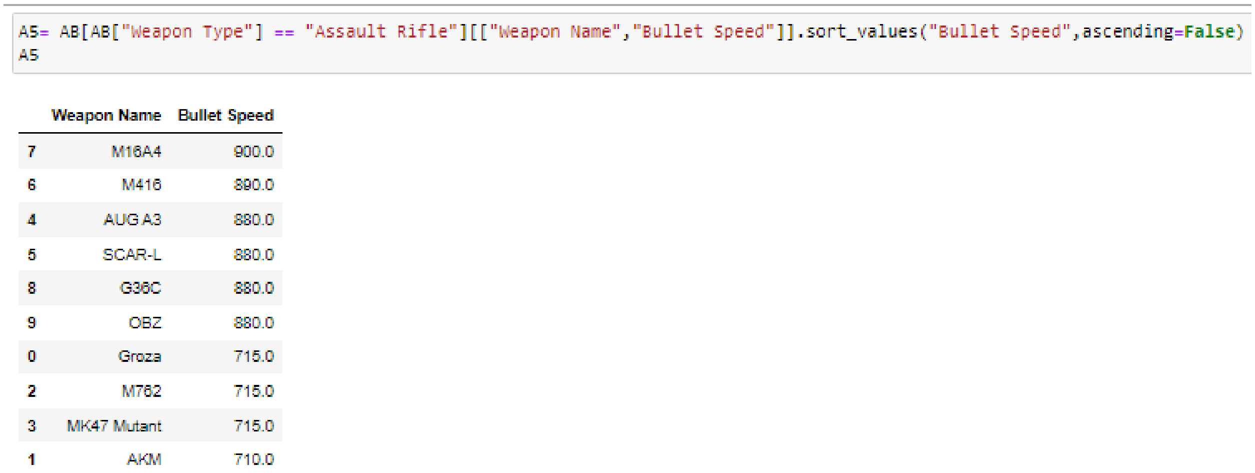


The following Plot shows that AWM have a Long Range.

Step 18 :

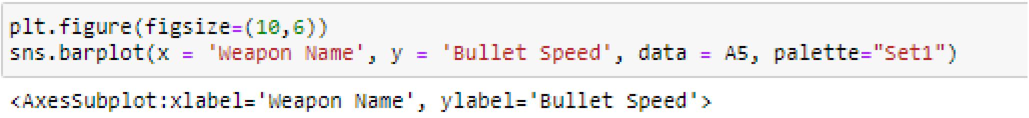
3) Bullet Speed

   a) Assault Rifle Visualization



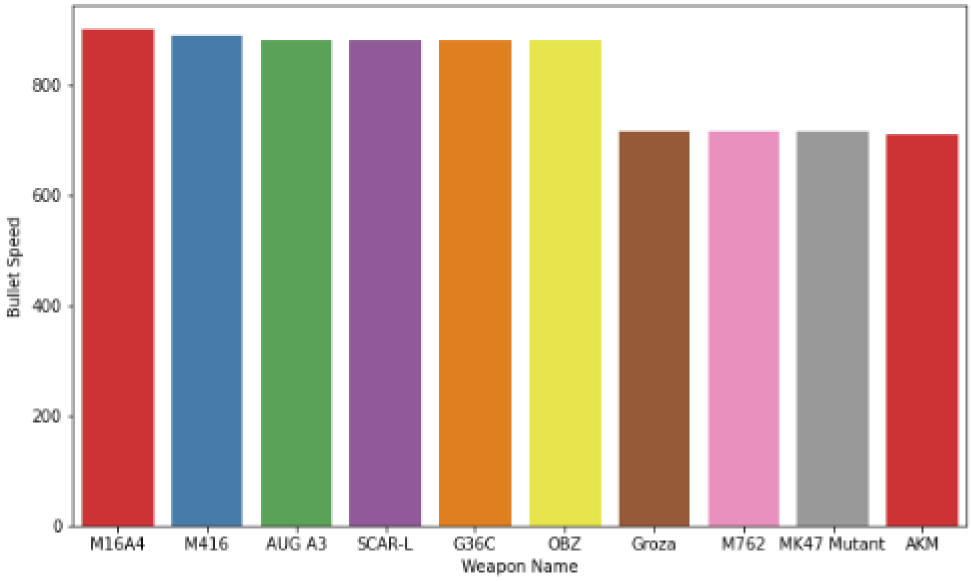
```
A5= AB[AB["Weapon Type"] == "Assault Rifle"][["Weapon Name","Bullet Speed"]].sort_values("Bullet Speed",ascending=False)
A5
```

	Weapon Name	Bullet Speed
7	M16A4	900.0
6	M416	890.0
4	AUG A3	880.0
5	SCAR-L	880.0
8	G36C	880.0
9	OBZ	880.0
0	Groza	715.0
2	M762	715.0
3	MK47 Mutant	715.0
1	AKM	710.0



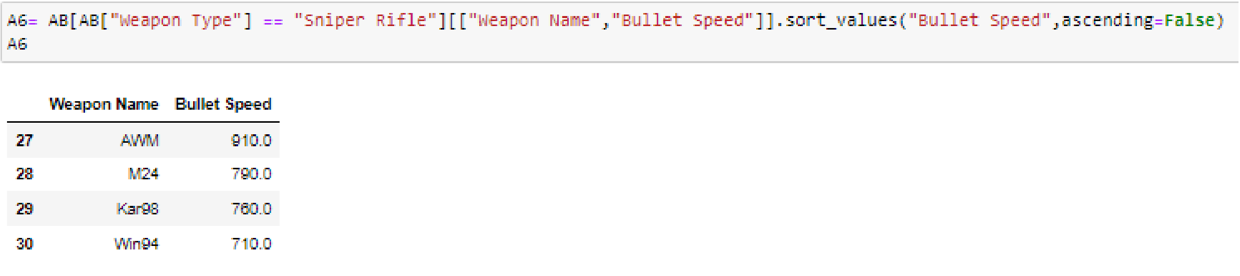
```
plt.figure(figsize=(10,6))
sns.barplot(x = 'Weapon Name', y = 'Bullet Speed', data = A5, palette="Set1")
```

```
<AxesSubplot:xlabel='Weapon Name', ylabel='Bullet Speed'>
```



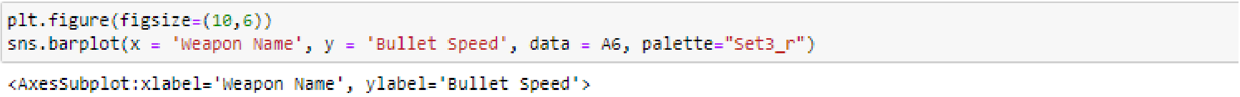
The following Plot shows that M16A4 has Fastest Bullet Speed but the problem is M16A4 is Burst & Single type of Weapon but M416 is Auto Weapon , So M416 is useable.

b) Sniper Rifle Visualization



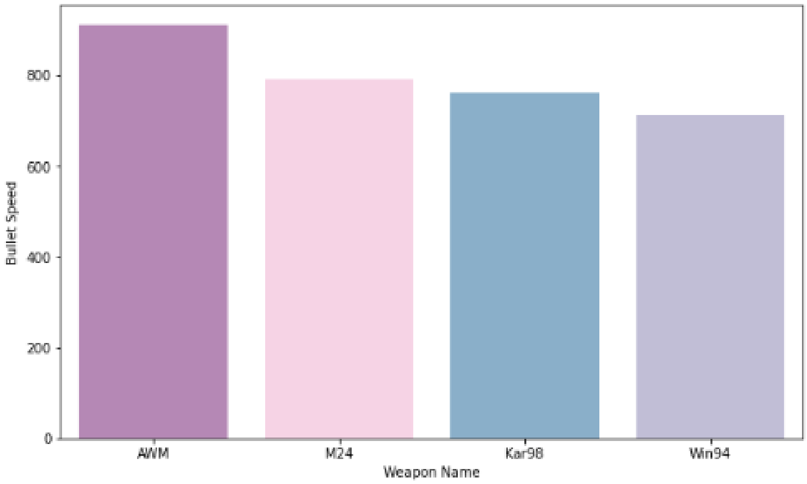
```
A6= AB[AB["Weapon Type"] == "Sniper Rifle"][["Weapon Name","Bullet Speed"]].sort_values("Bullet Speed",ascending=False)
A6
```

	Weapon Name	Bullet Speed
27	AWM	910.0
28	M24	790.0
29	Kar98	760.0
30	Win94	710.0



```
plt.figure(figsize=(10,6))
sns.barplot(x = 'Weapon Name', y = 'Bullet Speed', data = A6, palette="Set3_r")
```

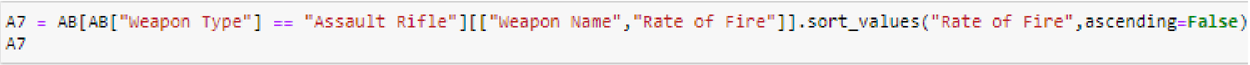
```
<AxesSubplot:xlabel='Weapon Name', ylabel='Bullet Speed'>
```



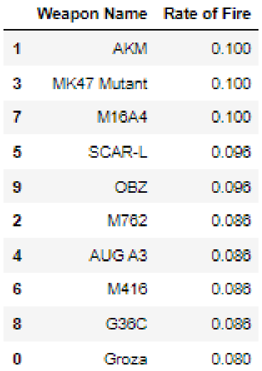
The Following Plot shows that AWM has Fastest Bullet Speed.

Step 19 :

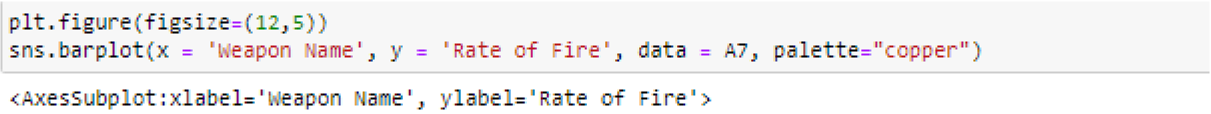
4) Rate of Fire
   a) Assualt Rifle Visualization



```
A7 = AB[AB["Weapon Type"] == "Assault Rifle"][["Weapon Name","Rate of Fire"]].sort_values("Rate of Fire",ascending=False)
A7
```

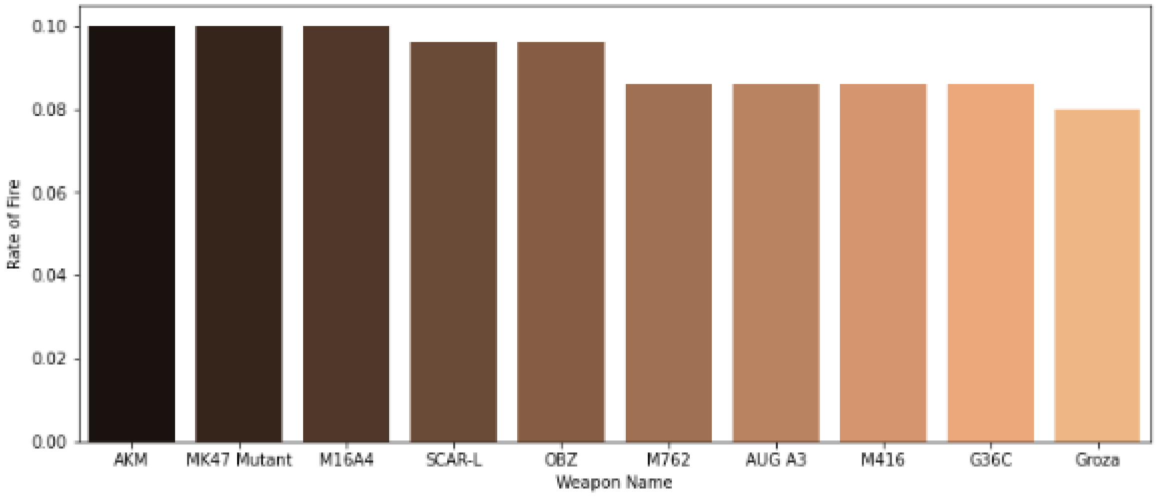


	Weapon Name	Rate of Fire
1	AKM	0.100
3	MK47 Mutant	0.100
7	M16A4	0.100
5	SCAR-L	0.096
9	QBZ	0.096
2	M762	0.086
4	AUG A3	0.086
6	M416	0.086
8	G36C	0.086
0	Groza	0.080



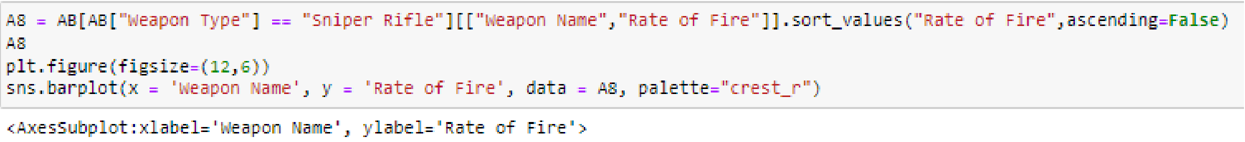
```
plt.figure(figsize=(12,5))
sns.barplot(x = 'Weapon Name', y = 'Rate of Fire', data = A7, palette="copper")
```

```
<AxesSubplot:xlabel='Weapon Name', ylabel='Rate of Fire'>
```



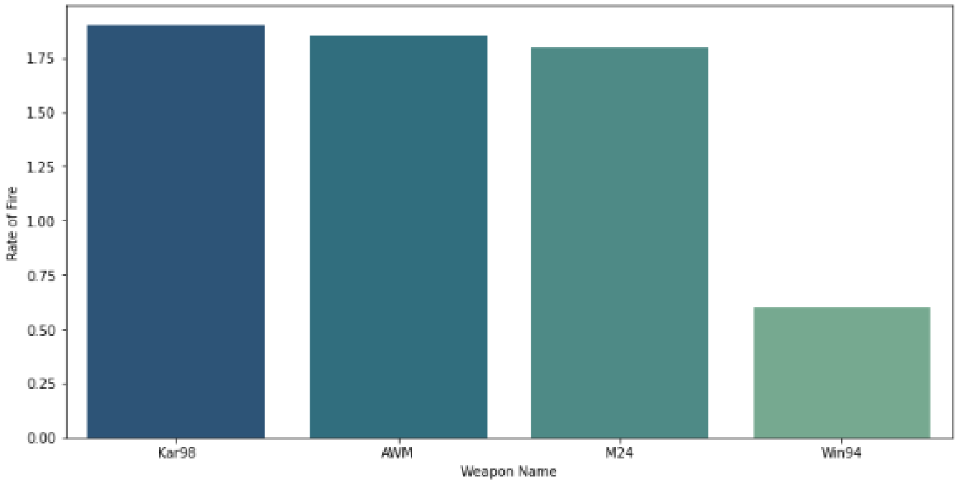
In this plot the lowest bar Weapon is the Best Weapon bcs it takes 0.08 second & fire next bullet , So Groza is less Rate of Fire so it is Best Weapon to use.

b) Sniper Rifle Visualization



```
A8 = AB[AB["Weapon Type"] == "Sniper Rifle"][["Weapon Name","Rate of Fire"]].sort_values("Rate of Fire",ascending=False)
A8
plt.figure(figsize=(12,6))
sns.barplot(x = 'Weapon Name', y = 'Rate of Fire', data = A8, palette="crest_r")
```

```
<AxesSubplot:xlabel='Weapon Name', ylabel='Rate of Fire'>
```



Step 20:

CONCLUSION :

These are the Analysis and Visualization on different Categories with respect to Weapon Type. From Above Individual Analysis of Every Weapon Name of **Assualt and Sniper Rifles**, we Conclude that **Groza + AWM** is the Beast Combination. M249 gives tough competition with Groza, either of them can be used, it's upto player wish.

Hope you liked it.