DATA ANALYSIS and VISUALIZATION

Data

*What is **Data**?*
Data is hard facts. These are units of information, often numeric, that are collected through observation.

The "data gets broadly divided into 2 kinds:

1. Qualitative or Categorical data
2. Quantitative or Numerical data

Categorical data comprises categories like *overweight*, *underweight*, or *normal* on a BMI scale as categories.
Similarly, it could be the flavor of ice cream. Categories could be *mango*, *vanilla*, *chocolate*, *pineapple*, *orange*, etc

Numerical data on the other hand is your numeric data like your actual *weight*, *height*, volume of your *ice cream* etc.

Data Analysis

*What is **Data Analysis**?*
Data Analysis is the process of evaluating data by applying statistical and/or logical techniques. *Data Analysis* involves:

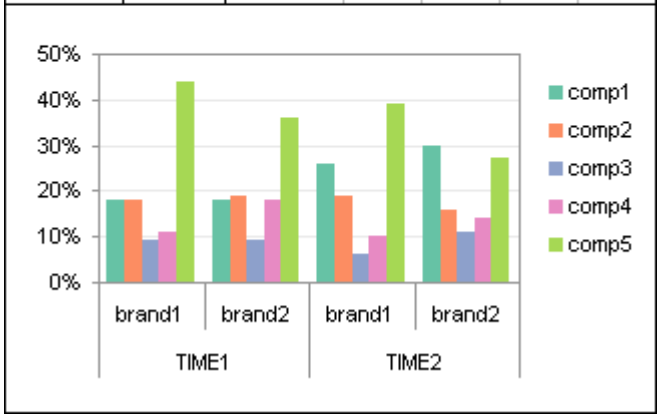
1. Performing *data cleaning/data wrangling* to improve data quality.
2. Getting data into the right format, getting rid of unnecessary data, correcting spelling mistakes, etc.
3. *Manipulating* data using tools like Excel or Python etc. This may include plotting the data out, creating pivot tables, and so on.
4. *Analyzing* and *interpreting* the data using statistical tools (i.e., finding correlations, trends, outliers, etc.).

Data Visualization

*What is **Data Visualization**?*
Data Visualization is the representation of data or information in a graph, chart, or other visual formats. Understanding of data in the raw form or tables would consume a lot of time and effort of stakeholders, readers, and users. Apart from this, it would be extremely difficult to interpret the main message in that form.

Instead, if we choose a graphical representation, we would use charts, graphs and infographics to express those messages, trends.

mkt share		METRIC				
time	PROD	comp1	comp2	comp3	comp4	comp5
TIME1	brand1	18%	18%	9%	11%	44%
	brand2	18%	19%	9%	18%	36%
TIME2	brand1	26%	19%	6%	10%	39%
	brand2	30%	16%	11%	14%	27%



https://peltiertech.com/images/img200811/pt_col_timebrand_comp.png

we are easily attracted towards graphical representation of the table. On top of that, it is easy to interpret as well.

Teacher's Activity: Understanding the lethal of Rohit Sharma

In this activity, we will understand how productive or vulnerable was Rohit Gurunath Sharma on each ball of the over from a given dataset.

Understanding the Dataset

The dataset consists of 3 columns or `features` namely `ball`, `batsman_runs` and `player_dismissed`.

1. `ball` represents the balls of the over Rohit Sharma has faced in his entire IPL career.
2. `batsman_runs` accounts for the runs attributed to batsman for that particular ball.
3. `player_dismissed` provides the name of player who was dismissed on a particular ball.

Understanding the Approach

1. Understanding the most vulnerable ball for Rohit Sharma: Here, we will take into account the number of times the player himself was dismissed on any given ball of the over. The columns we will use are:

    A. `player_dismissed` with entries specific to `RG Sharma`.
    B. `ball`.

    We will `group` the data on `ball` of the over and plot a `bar` graph to interpret the most number of dismissals for a particular ball of over

1. Understanding the most productive ball for Rohit Sharma: Here, we will consider the total runs scored by the player on any given ball of the over. The columns we will use are:

    A. `batsman_runs` .
    B. `ball` .

    We will `group` the data on `ball` of the over and plot a `bar` graph to interpret the most runs scored on a particular ball of over

Importing Packages

Packages are imported in following manner.

```
import package_name
```

In the next cell we have imported the following packages.

1. `pandas` . It is the most common library used by data scientists for data manipulation and cleaning
2. `numpy` . It adds support for arrays, along with a collection of mathematical functions to operate on these arrays.
3. `matplotlib` . It is a plotting library for python. `.pyplot` is a sub-package or set of functions available in matplotlib which we'll be using

`pd` , `np` , `plt` are all `aliases` for their corresponding packages. `Alias` are second name assigned to values or variables.

`%matplotlib inline` is a "magic function" renders plots

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Loading the Dataset

In the cell below, we have created a new pandas DataFrame by the name `df` and imported the mentioned file.
We have used `.head()` function to see the first 5 values of the dataset we created.

`.head()` can show up any number of values based on the parameter given.
If we want to see more, we can pass value in the function like `df.head(10)` will show first 10 values of the dataset

In [2]:
```
df = pd.read_csv("https://raw.githubusercontent.com/jainharshit27/datasets/main/Rohi
df.head()
```

Out[2]:

	ball	batsman_runs	player_dismissed
0	2	0	NaN
1	3	0	NaN
2	4	0	RG Sharma
3	3	1	NaN

	ball	batsman_runs	player_dismissed
4	5	1	NaN

1. Understanding the most vulnerable ball for Rohit Sharma

Reducing the dataset to our need

In the cell below, we have created a new pandas DataFrame by the name `df_Rohit` and assigned it a filtered version of dataframe `df` such that only those observations are accepted which have `player_dismissed` value as `RG Sharma`. This can be done like:

```
df[df["player_dismissed"] == "RG Sharma"]
```

Here, `df["player_dismissed"] == "RG Sharma"`, this value will mark observation True wherever it is. Passing that value through `df[]` will filter out the `False` values.

Then, we have grouped data using `.grouby()` function using various values of `ball` feature/column. The `groupby()` function is then followed by `.count()` to summarize values for other numerical columns in the dataframe. The resulting dataframe is then assigned to dataframe `df_Rohit_dismissed`.

In [3]:
```
df_Rohit = df[df["player_dismissed"] == "RG Sharma"]
df_Rohit_dismissed = df_Rohit.groupby("ball").count()
df_Rohit_dismissed
```

Out[3]:

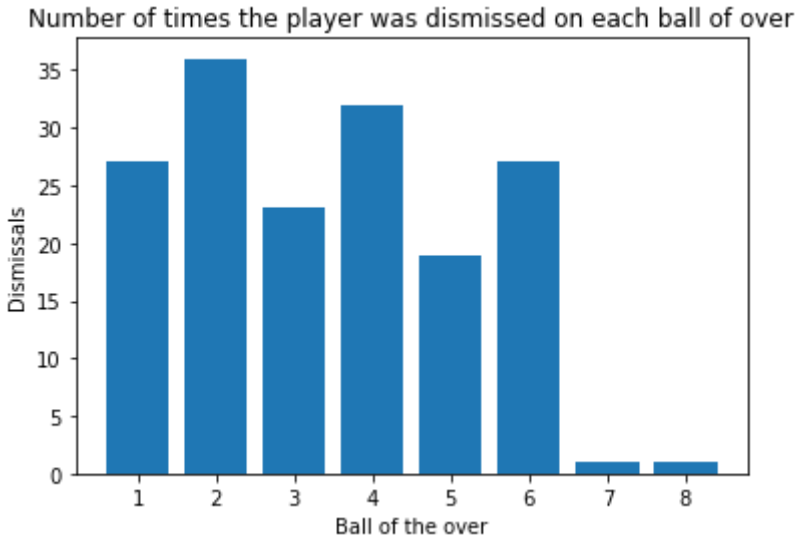
ball	batsman_runs	player_dismissed
1	27	27
2	36	36
3	23	23
4	32	32
5	19	19
6	27	27
7	1	1
8	1	1

Plotting of information

`plt.title()` provides the graph or chart with a title. The title we have used is "Number of times player was dismissed on each ball of over".
`plt.bar()` function is used to plot bar chart. We have plotted bar chart for `df_Rohit_dismissed` dataframe's index value which are, infact, each ball of the over as categories or x-axis of the chart and the dismissals of Rohit Sharma as y-axis.
`plt.show()` function combines all the elements of charts and shows them in harmony.

In [4]:
```
plt.title("Number of times the player was dismissed on each ball of over")
plt.bar(df_Rohit_dismissed.index, df_Rohit_dismissed["player_dismissed"])
plt.xlabel("Ball of the over")
```

```
plt.ylabel("Dismissals")
plt.show()
```



We have plotted the graph with 0-8 being the categories representing each ball of the over where 7 & 8 are the balls that occurred when the over had wide or no balls. The height of the categories is based upon the count of the player_dismissed feature. The graph is an output of the code.

Conclusion: RG Sharma is most vulnerable to 2nd ball of the over.

2. Understanding the most productive ball for Rohit Sharma

Grouping of data

we have grouped data using `.grouby()` function using various values of `ball` feature/column. The `groupby()` function is then followed by `.sum()` to summarize values for other numerical columns in the dataframe. The resulting dataframe is then assigned to dataframe `df_runs_per_ball` .

In [5]:
```
df_runs_per_ball = df.groupby("ball").sum()
df_runs_per_ball
```

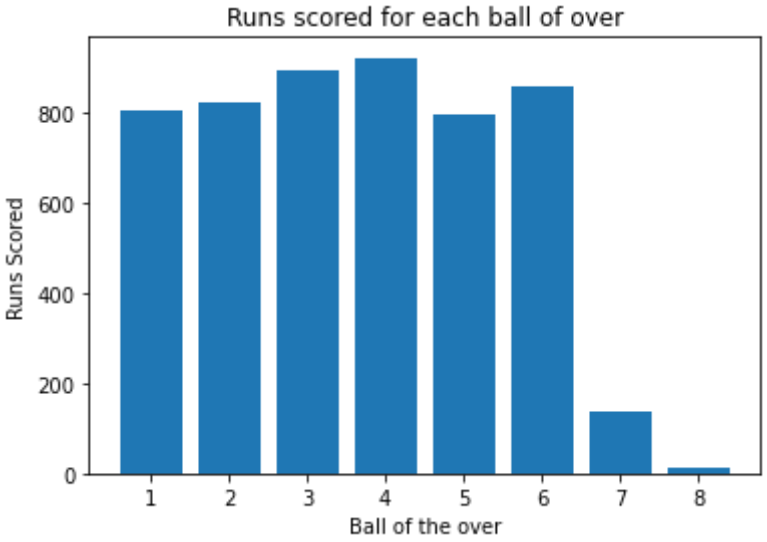
Out[5]:

ball	batsman_runs
1	804
2	819
3	890
4	920
5	795
6	855
7	136
8	11

Plotting of information

`plt.title()` provides the graph or chart with a title.
`plt.bar()` function is used to plot bar chart. We have plotted bar chart for `df_runs_per_ball` dataframe's index value which are, infact, each ball of the over as categories or x-axis of the chart and the runs scored by Rohit Sharma as y-axis.
`plt.show()` function combines all the elements of charts and shows them in harmony.

In [6]:
```
plt.title("Runs scored for each ball of over")
plt.bar(df_runs_per_ball.index, df_runs_per_ball["batsman_runs"])
plt.xlabel("Ball of the over")
plt.ylabel("Runs Scored")
plt.show()
```



We have plotted the graph with 0-8 being the categories representing each ball of the over where 7 & 8 are the balls that occurred when the over had wide or no balls. The height of the categories is based upon the sum of the batsman_runs feature. The graph is an output of the code.

Conclusion: RG Sharma has scored most in the 4th ball of the over.