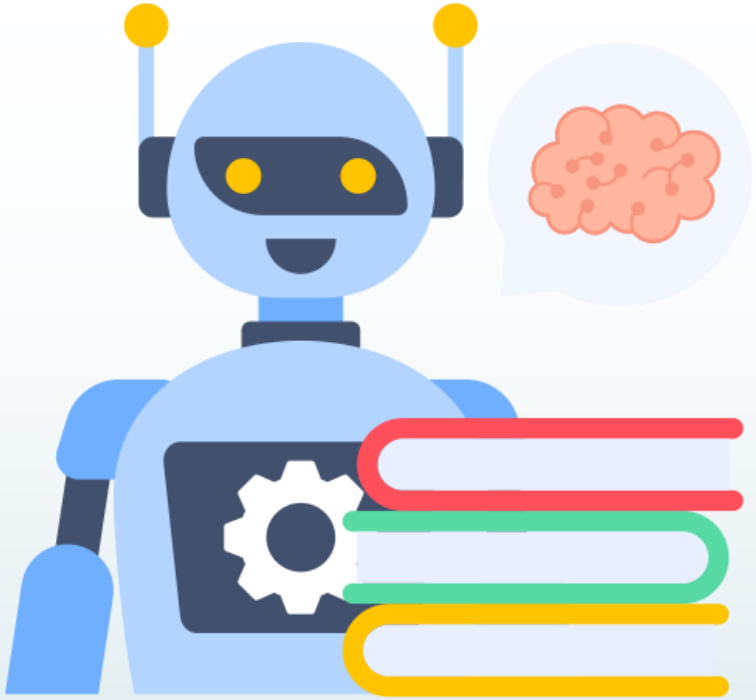


Descriptive Analytics

Ashima Tyagi
Assistant Professor
School of Computer Science & Engineering

- Descriptive analytics is a component of analytics and is the science of describing the past data; it thus capture "what happened" in a given context.
- The primary objective of descriptive analytics is comprehension of data using **data summarization, basic statistical measures** and **data visualization**.

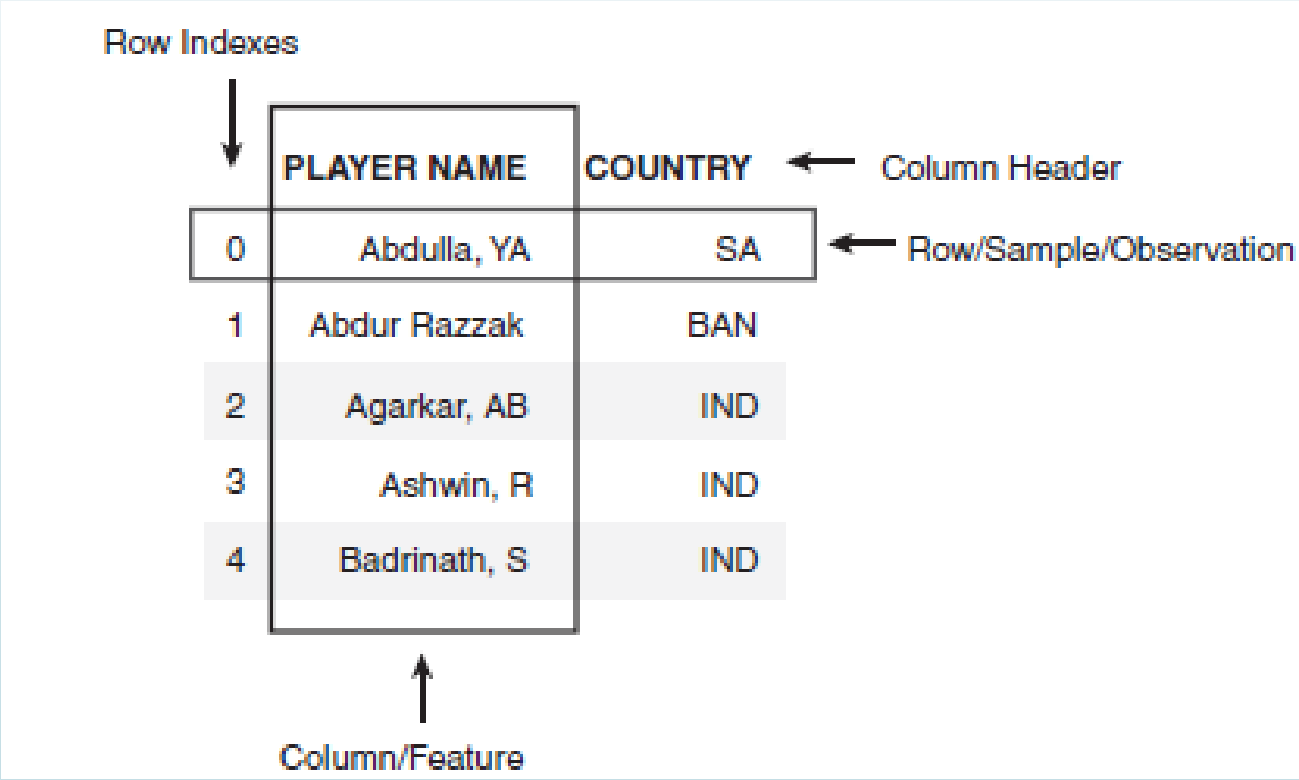
Outline

- DataFrame
- Working with Pandas Library
- Exploratory Data Visualization

DataFrame

- A DataFrame is very efficient two-dimensional data structure as shown in Figure.
- It is flat in structure and is arranged in rows and columns.
- Rows and columns

can be indexed or named.



Library to process dataframe:

Pandas library support methods to explore, analyze, and prepare data. It can be used for performing activities such as load, filter, sort, group, join datasets and also for dealing with missing data.

Working with Pandas Library

- Pandas is an open-source, Berkeley Software Distribution (BSD)-licensed library providing *high-performance, easy-to-use data structures and data analysis tools* for the Python programming language (source:https://pandas.pydata.org/).

- To use the Pandas library, we need to import pandas module into the environment using the import keyword-

```
import pandas as pd
```

- Pandas library has provided different methods for loading datasets with many different formats onto DataFrames. For example:

1. read_csv to read comma separated values.
2. read_json to read data with json format.
3. read_fwf to read data with fixed width format.
4. read_excel to read excel files.
5. read_table to read database tables.

- Information of read_csv method:

```
pd.read_csv? (Press SHIFT + ENTER)
```

- IPL dataset is stored in comma separated values (csv) format, so we will use pd.read_csv method to read and load it onto a DataFrame. The dataset contains header information in the first line.

```
df = pd.read_csv('IPL IMB381IPL2013.csv')
```

- To find out the type of variable ipl_auction_df, we can pass the variable to type() function of python.

```
type(df)
```

Pandas Features:

The following operations are performed on the dataset using pandas library

- Selecting
- Filtering
- Aggregating
- Joining
- Slicing/dicing

- Displaying First Few Record: **df.head(5)**
- Finding Summary of the DataFrame: **list(df.columns)**
- Transpose the dataframe: **df.head(5).transpose()**
- Dimension or size of the DataFrame: **df.shape**
- Detailed summary: **df.info()**

Slicing and Indexing of Dataframe

- To select few rows and columns, the DataFrame can be accessed or sliced by indexes or names.
- The row and column indexes always start with value 0.

Example:

- Start with row with index 0 and end with row with index 5, but not including 5. [0:5] is same as [:5]. By default, the indexing always starts with 0: **df[0:5]**
- Negative indexing: to select the last five records: **df[-5:]**
- Specific columns of a DataFrame can also be selected or sliced by column names.: **df['COL NAME'][0:5]**
- To select two columns,: **df[['COL1 NAME', 'COL2']][0:5]**
- Specific rows and columns can also be selected using row and column indexes. Use iloc(index location) method: **df.iloc[4:9, 1:4]**

- The occurrences of each unique value in a column:

  df.Col.value_counts()

- Passing parameter normalize=True to the value_counts() will calculate the percentage of occurrences of each unique value.:

  df.COl.value_counts(normalize=True)*100

- Sorting DataFrame by Column Values(ascending by default):

  df[['Col1', 'Col2']].sort_values('Col2')[0:5]

- Sorting in Descending order;

  df [['Col1', 'Col2']]. sort_values('Col2', ascending = False)[0:5]

- Creating New Columns:
  - **df['Col3'] = df['Col1'] – df['Col2']**

Grouping and Aggregating

Sometimes, it may be required to group records based on column values and then apply aggregated operations such as mean, maximum, minimum, etc.

Example:

- To find average SOLD PRICE for each age category, group all records by AGE and then apply mean() on SOLD PRICE column.:
  df.groupby('AGE')['SOLD PRICE'].mean()
- Put the result in a different dataframe

```
df_mean_age = df.groupby('AGE')['SOLD
PRICE'].mean().reset_index()
```

- Joining DataFrames (axis=0 means row-wise):

```
New_df = pd.concat( [df1, df2], axis=0)
```

- Re-naming columns:

```
df.rename(columns = {'col1':'col_1'}, inplace = True)
```

- Filtering Records Based on Conditions

```
df [ df['Col1'] > 80 ] [['Col2', 'Col1']]
```

- Removing a Column or a Row from a Dataset:

- 1. To drop a column, pass the column name and axis as 1.
- 2. To drop a row, pass the row index and axis as 0.

```
df.drop('col1', inplace = True, axis = 1)
```

HANDLING MISSING VALUES

To check if there are null values in a dataframe:

df.isnull()

We can verify if some of the rows contain null values in column.

df['col1'].isnull()

- We can drop the rows with null values:

df = df.dropna()

df = df['COLl'].dropna()

EXPLORATION OF DATA USING VISUALIZATION

- Data visualization is useful to gain insights and understand what happened in the past in a given context.
- It is also helpful for feature engineering.

We can use the following charts for data visualization.

Drawing Plots

1. Bar Chart
2. Histogram
3. Density Plot
4. Box Plot
5. Scatter Plot

Drawing Plots

- Matplotlib is a Python 2D plotting library for creating 2D plots of arrays in Python. Matplotlib is written in Python and makes use of NumPy arrays. It is well integrated with pandas to read columns and create plots.
- It provides extensive set of plotting APIs to create various plots such as scattered, bar, box, and distribution plots with custom styling and annotation. Detailed documentation for matplotlib can be found at https://matplotlib.org/.
- To display the plots on the Jupyter Notebook, we need to provide a directive %matplotlib inline. Only if the directive is provided, the plots will be displayed on the notebook.
- Seaborn is also a Python data visualization library based on matplotlib.
- It provides a high-level interface for drawing innovative and informative statistical charts (source: https://seaborn.pydata.org/).
- Seaborn, which is built on top of matplotlib, is a library for making elegant charts in Python and is well integrated with pandas DataFrame.

```
import matplotlib.pyplot as plt
import seaborn as sn
%matplotlib inline
```

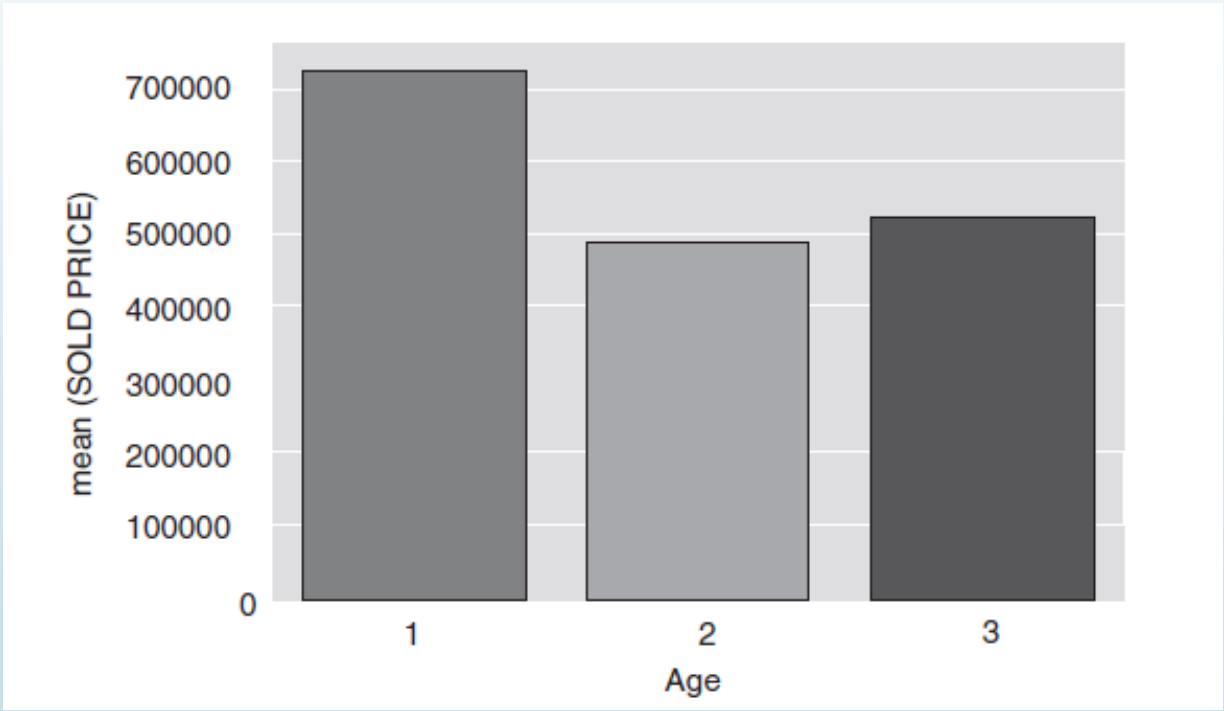
1. Bar Chart

- Bar chart is a frequency chart for qualitative variable (or categorical variable). Bar chart can be used to assess the most-occurring and least-occurring categories within a dataset.
- To draw a bar chart, call barplot() of seaborn library.
- Bar chart can be used to represent quantitative data.

To display the average sold price by each age category, pass SOLD PRICE as y parameter and AGE as x

parameter. Figure 2.3 shows a bar plot created to show the average SOLD PRICE for each age category

sn.barplot(x = 'AGE', y = 'SOLD PRICE', data = df);



In Figure, it can be noted that the average sold price is higher for age category 1.

Bar chart

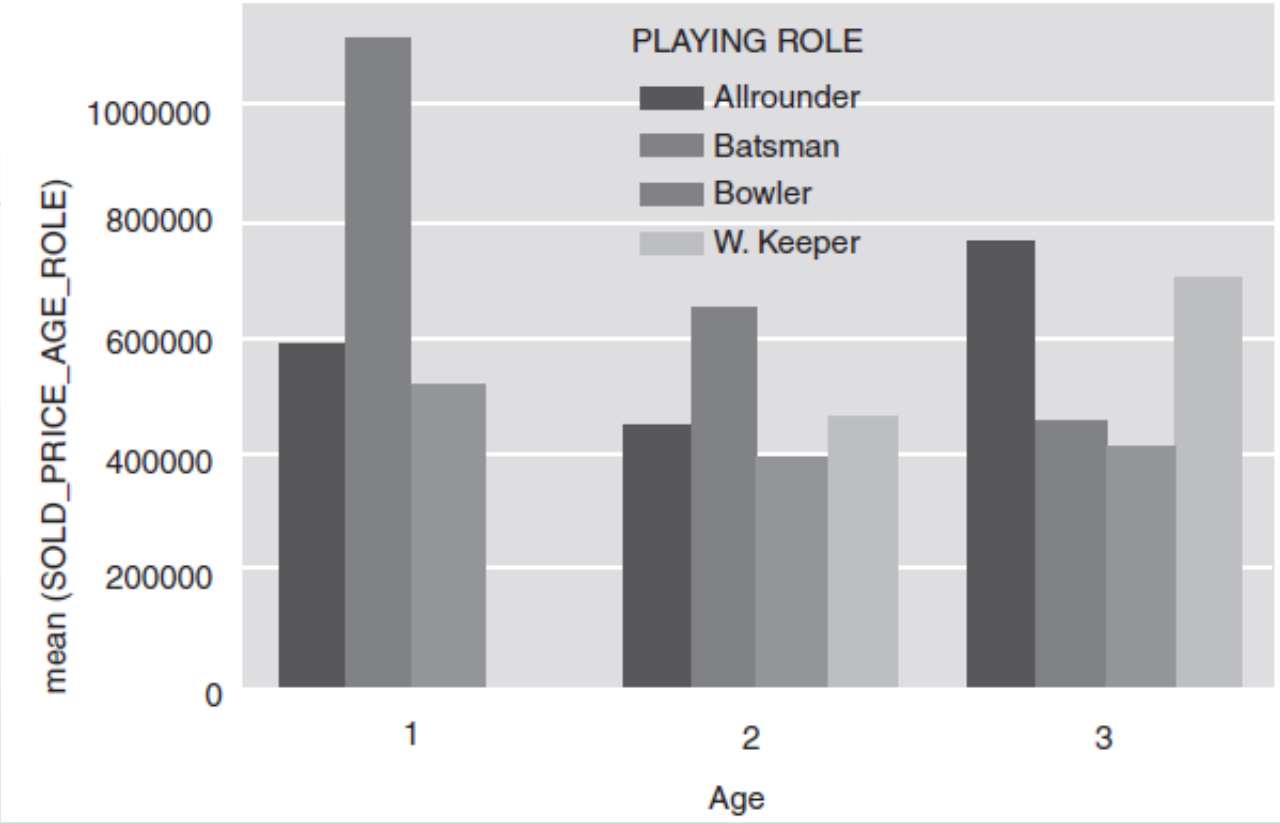
We can also create bar charts, which are grouped by a third variable.

In the following example, average sold price is shown for each age category but grouped by a third variable, that is, playing roles. The parameter hue takes the third variable as parameter.

In this case, we pass PLAYING ROLE as hue parameter.

```
sn.barplot(x = 'AGE', y = 'SOLD_PRICE_AGE_ROLE', hue = 'PLAYING ROLE', data = df);
```

Bar chart



In Figure, it can be noted that in the age categories 1 and 2, batsmen are paid maximum, whereas allrounders are paid maximum in the age category 3.

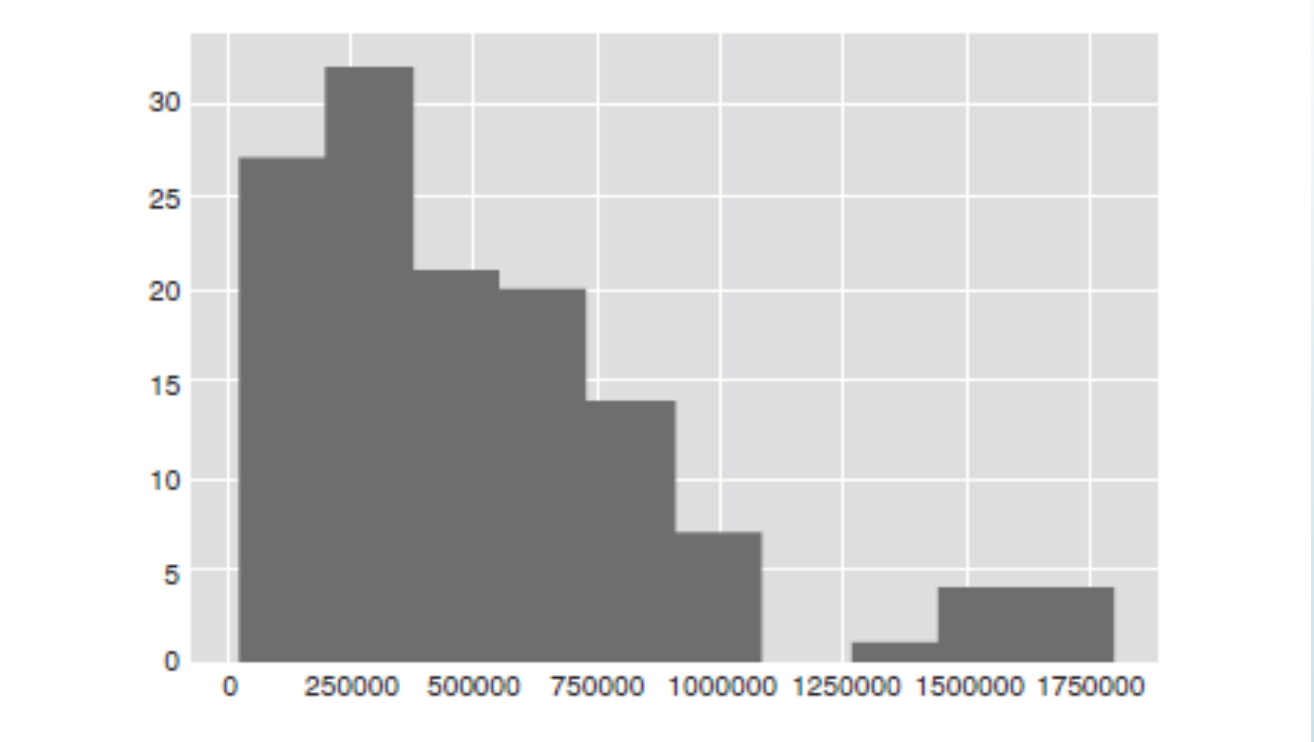
This could be because allrounders establish their credentials as good allrounders over a period.

2. Histogram

- A histogram is a plot that shows the frequency distribution of a set of continuous variable. Histogram gives an insight into the underlying distribution (e.g., normal distribution) of the variable, outliers, skewness, etc.
- To draw a histogram, invoke hist() method of matplotlib library.

The following is an example of how to draw a histogram for SOLD PRICE and understand its distribution

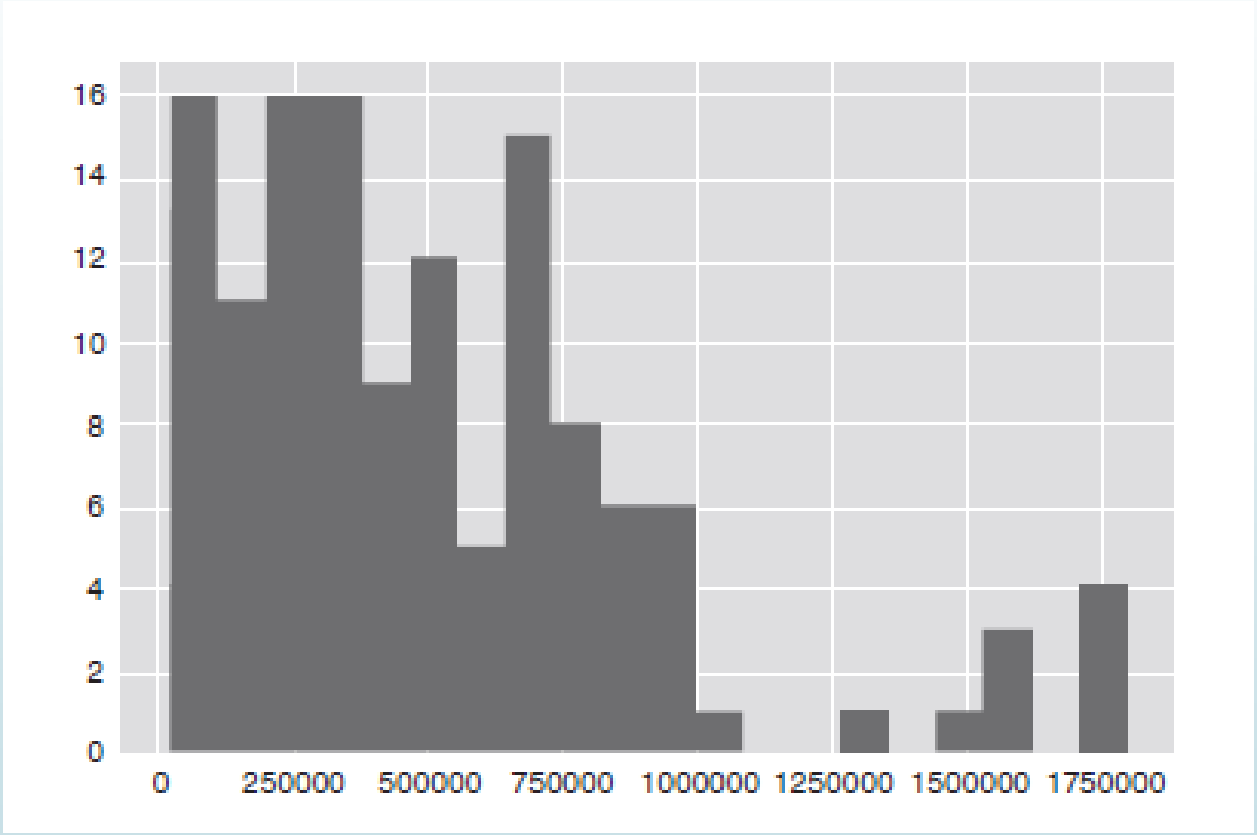
```
plt.hist(df['SOLD PRICE'] );
```



The histogram shows that SOLD PRICE is right skewed. Most players are auctioned at low price range of 250000 and 500000, whereas few players are paid very highly, more than 1 million dollars.

By default, it creates 10 bins in the histogram. To create more bins, the bins parameter can be set in the hist() method as follows:

```
plt.hist(df['SOLD PRICE'], bins= 20 );
```



3. Distribution or Density Plot

A distribution or density plot depicts the distribution of data over a continuous interval. Density plot is like smoothed histogram and visualizes distribution of data over a continuous interval.

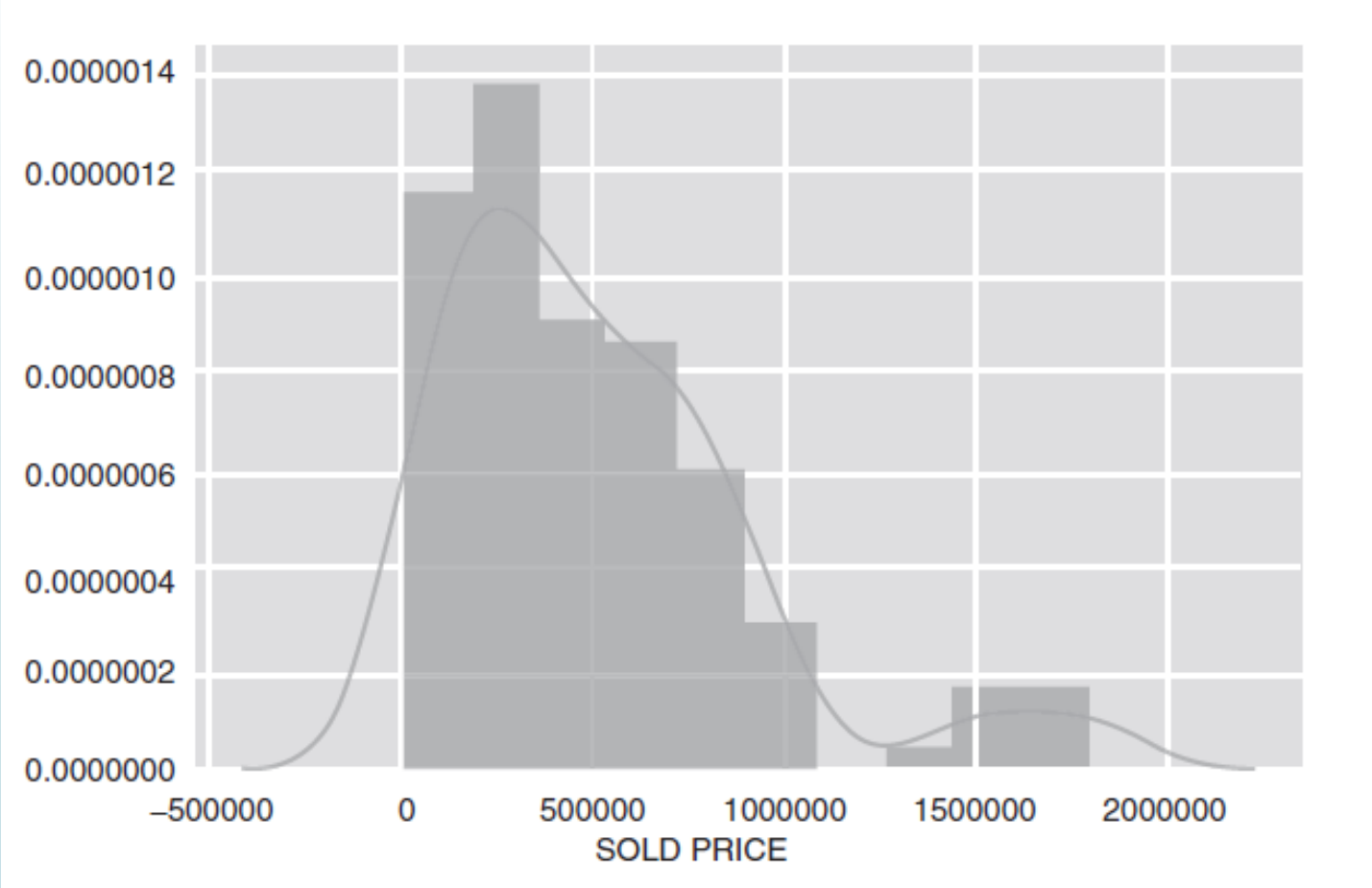
So, a density plot also gives insight into what might be the distribution of the population

Difference between Density plot and histogram:

In simple terms, a histogram shows the counts of values in each range, while a density plot shows the proportion of values in each range. While a histogram is made up of bars that touch each other, a density plot is a smooth curve that shows the distribution of the data in a more continuous way.

To draw the distribution plot, we can use distplot() of seaborn library. Density plot for the outcome variable "SOLD PRICE" is shown

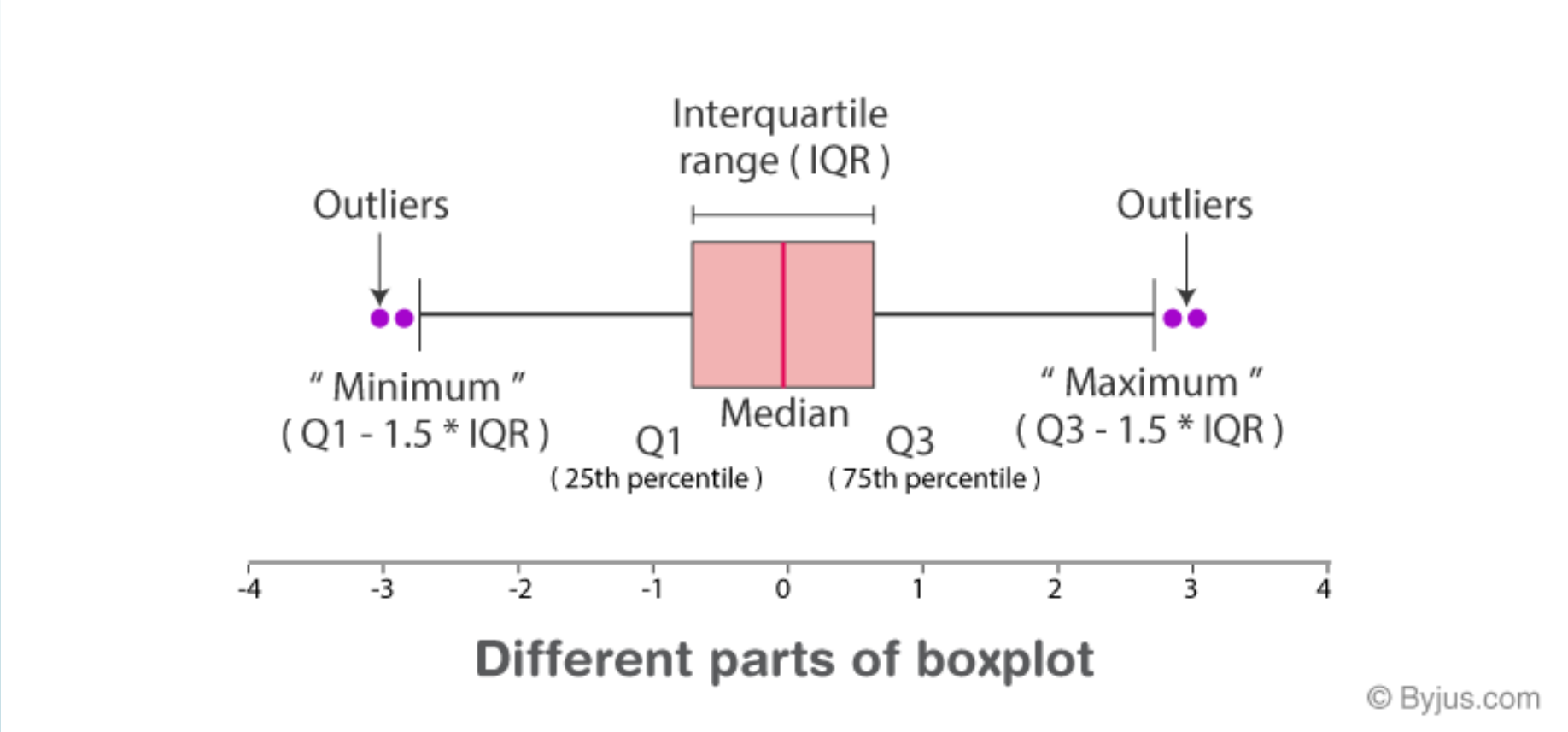
sn.distplot(df['SOLD PRICE']);



4. Box Plot

- When we display the data distribution in a standardized way using 5 summary of one column– minimum, Q1 (First Quartile), median, Q3(third Quartile), and maximum, it is called a Box plot.
- It is also termed as box and whisker plot.
- The method to summarize a set of data that is measured using an interval scale is called a box and whisker plot. These are maximum used for data analysis

image below which shows the minimum, maximum, first quartile, third quartile, median and outliers.



Box Plot

- **Minimum:** The minimum value in the given dataset
- **First Quartile (Q1):** The first quartile is the median of the lower half of the data set.
- **Median:** The median is the middle value of the dataset, which divides the given dataset into two equal parts. The median is considered as the second quartile.
- **Third Quartile (Q3):** The third quartile is the median of the upper half of the data.
- **Maximum**: The maximum value in the given dataset.

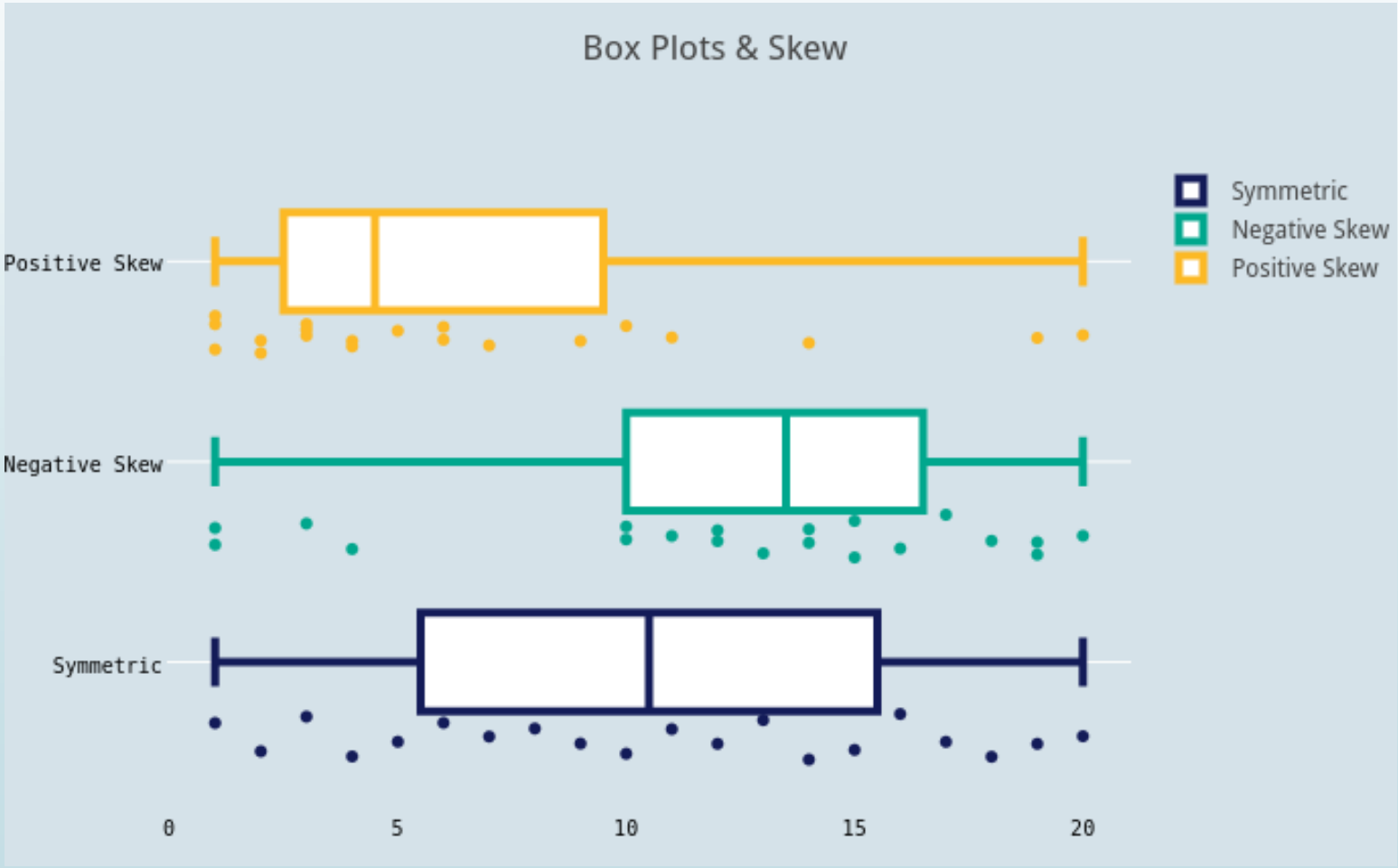
Apart from these five terms, the other terms used in the box plot are:

- **Interquartile Range (IQR):** The difference between the third quartile and first quartile is known as the interquartile range. (i.e.) IQR = Q3-Q1
- **Outlier:** The data that falls on the far left or right side of the ordered data is tested to be the outliers. Generally, the outliers fall more than the specified distance from the first and third quartile.

(i.e.) Outliers are greater than Q3+(1.5 . IQR) or less than Q1-(1.5 . IQR).

Box plot distribution

The box plot distribution will explain how tightly the data is grouped, how the data is skewed, and also about the symmetry of data.

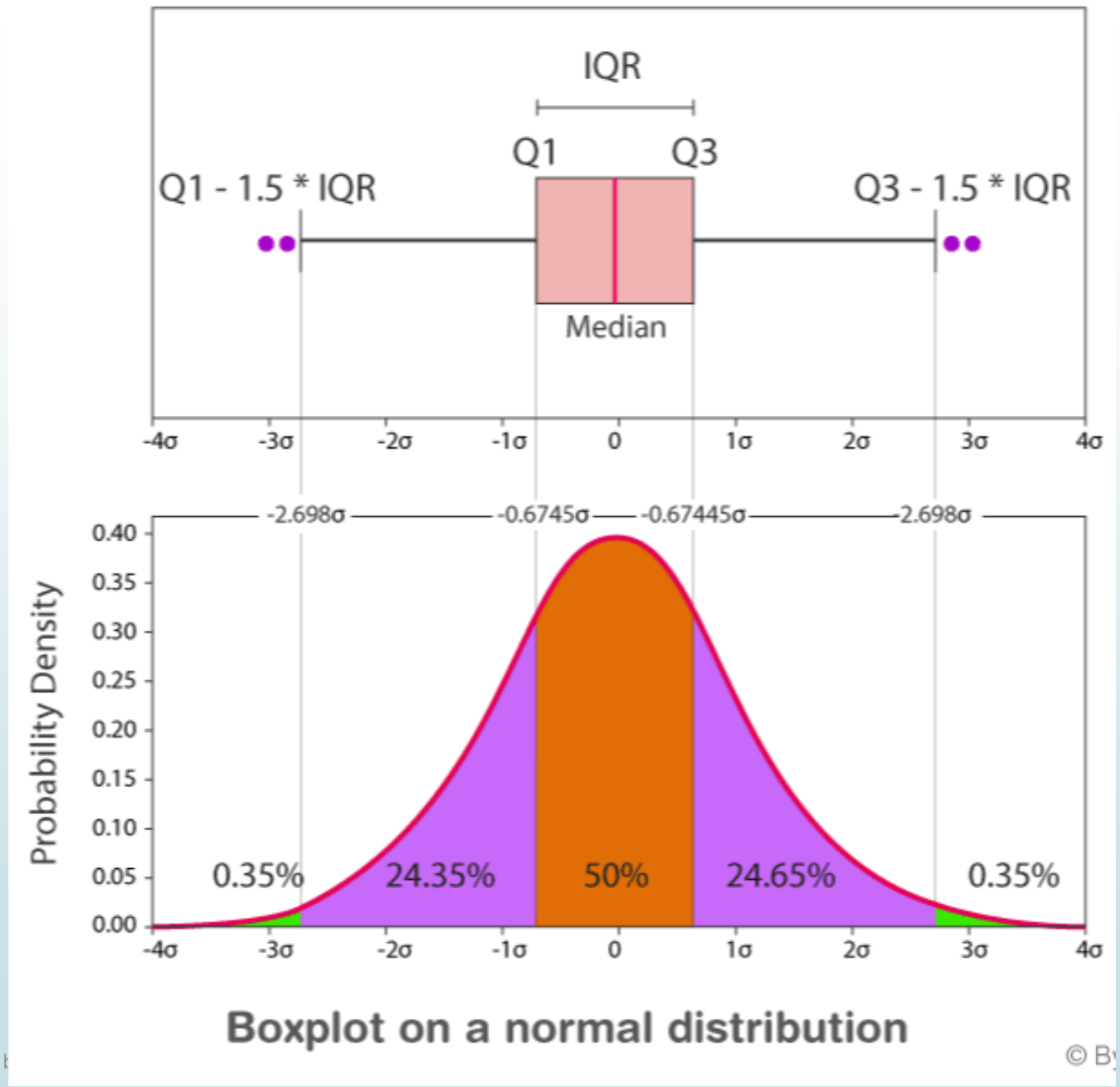


Box Plot

- **Positively Skewed:** If the distance from the median to the maximum is greater than the distance from the median to the minimum, then the box plot is positively skewed.

- **Negatively Skewed:** If the distance from the median to minimum is greater than the distance from the median to the maximum, then the box plot is negatively skewed.

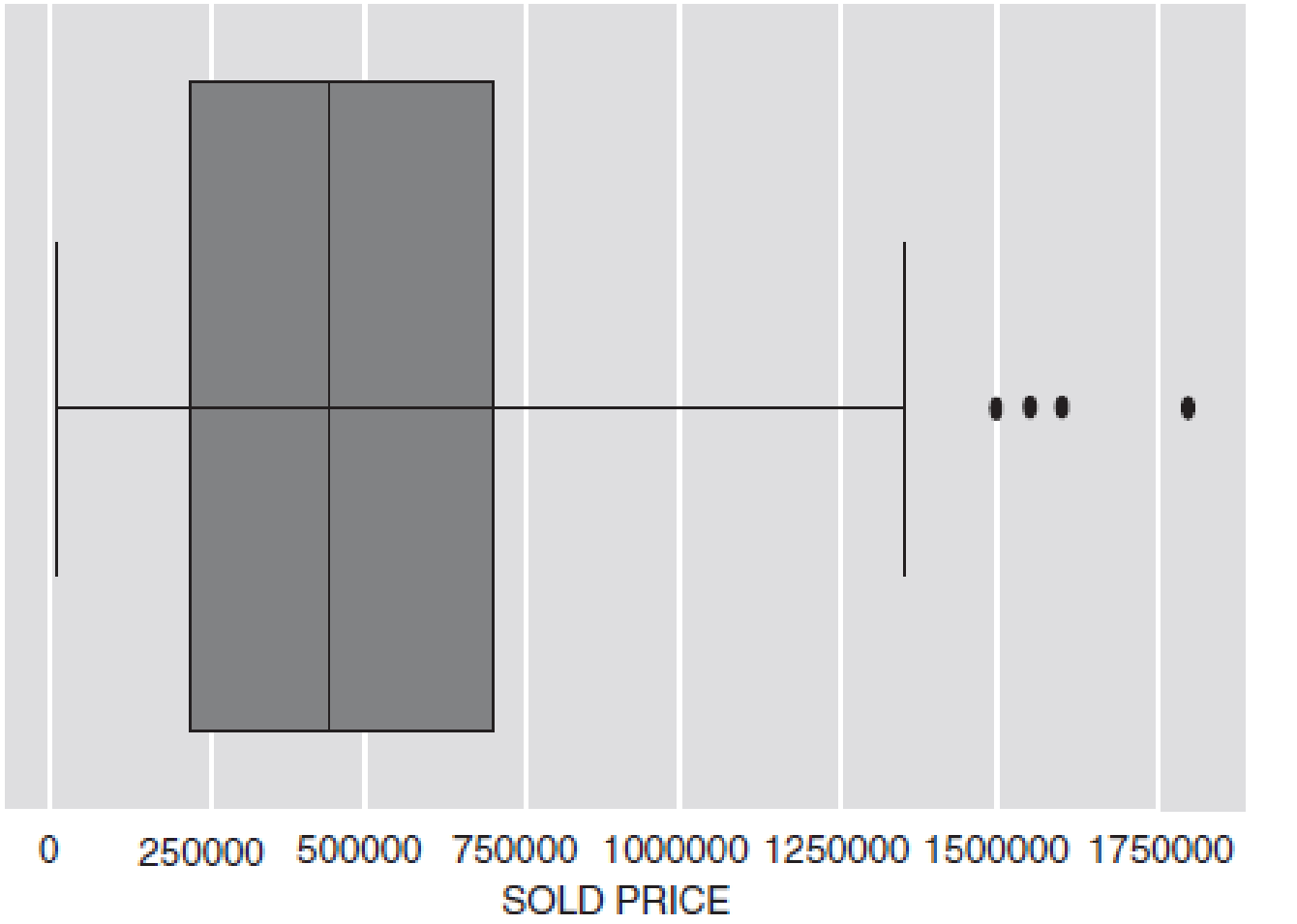
- **Symmetric:** The box plot is said to be symmetric if the median is equidistant from the maximum and minimum values.


IQR
Q1
Q3
Q1 - 1.5 * IQR
Q3 - 1.5 * IQR
Median
-4σ
-3σ
-2σ
-1σ
0
1σ
2σ
3σ
4σ
-2.698σ
-0.6745σ
-0.67445σ
2.698σ
Probability Density
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0.35%
24.35%
50%
24.65%
0.35%
Boxplot on a normal distribution

© B

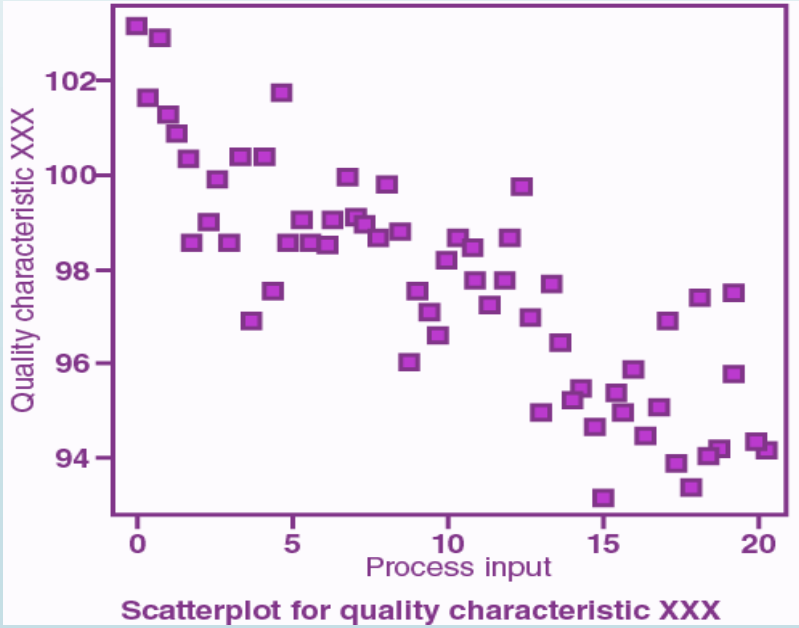
Box Plot

```
box = sn.boxplot(df['SOLD PRICE']);
```



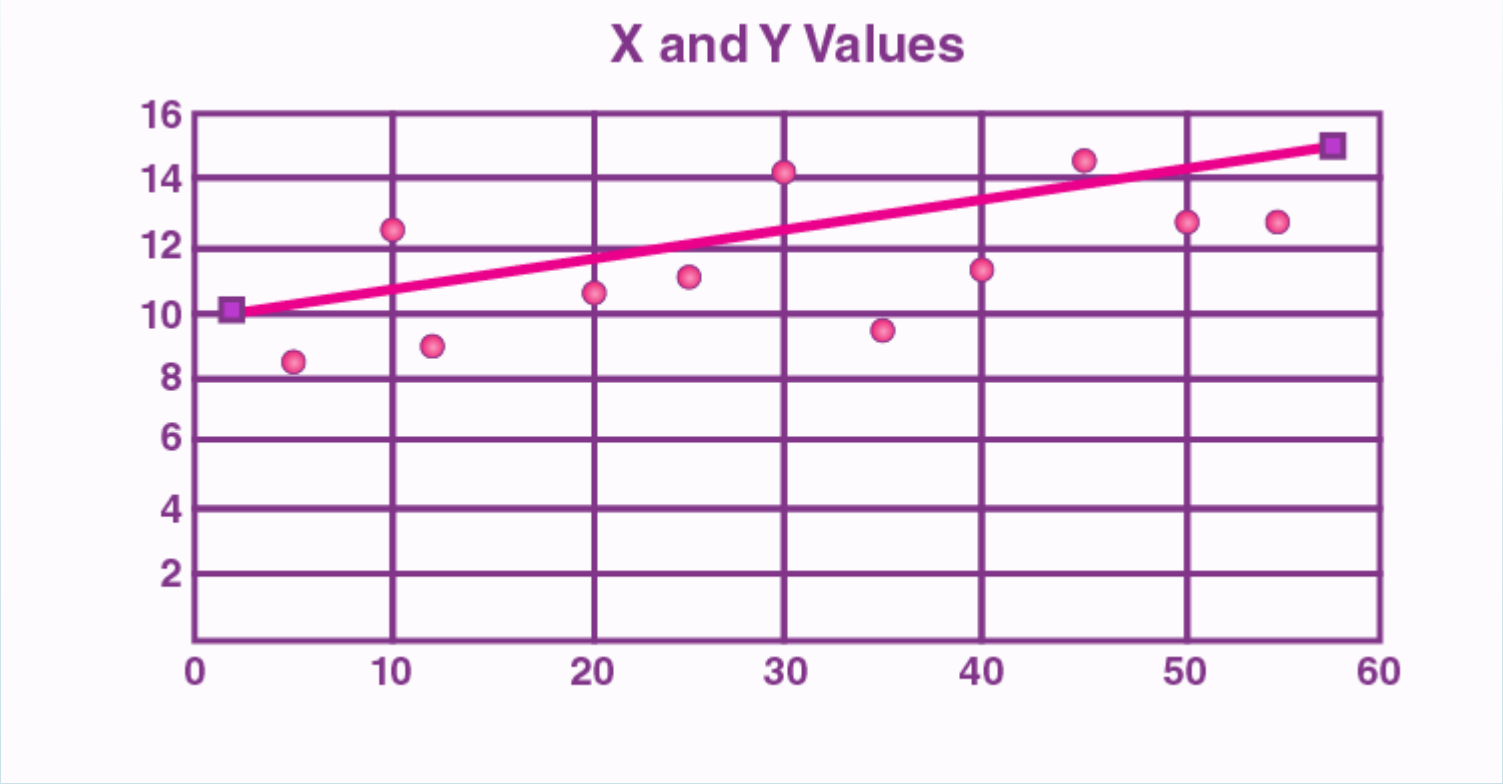
5. Scatter Plot

- Scatter plots are the graphs that present the relationship between two variables in a data-set.
- It represents data points on a two-dimensional plane or on a Cartesian system.
- The independent variable or attribute is plotted on the X-axis, while the dependent variable is plotted on the Y-axis.
- These plots are often called scatter graphs or scatter diagrams.



Scatterplot for quality characteristic XXX

The line drawn in a scatter plot, which is near to almost all the points in the plot is known as “line of best fit” or “trend line“. See the graph below for an example.



Scatter plot Correlation:

- We know that the correlation is a statistical measure of the relationship between the two variables' relative movements.
- The better the correlation, the closer the points will touch the line.
- This cause examination tool is considered as one of the seven essential quality tools.

Types of correlation

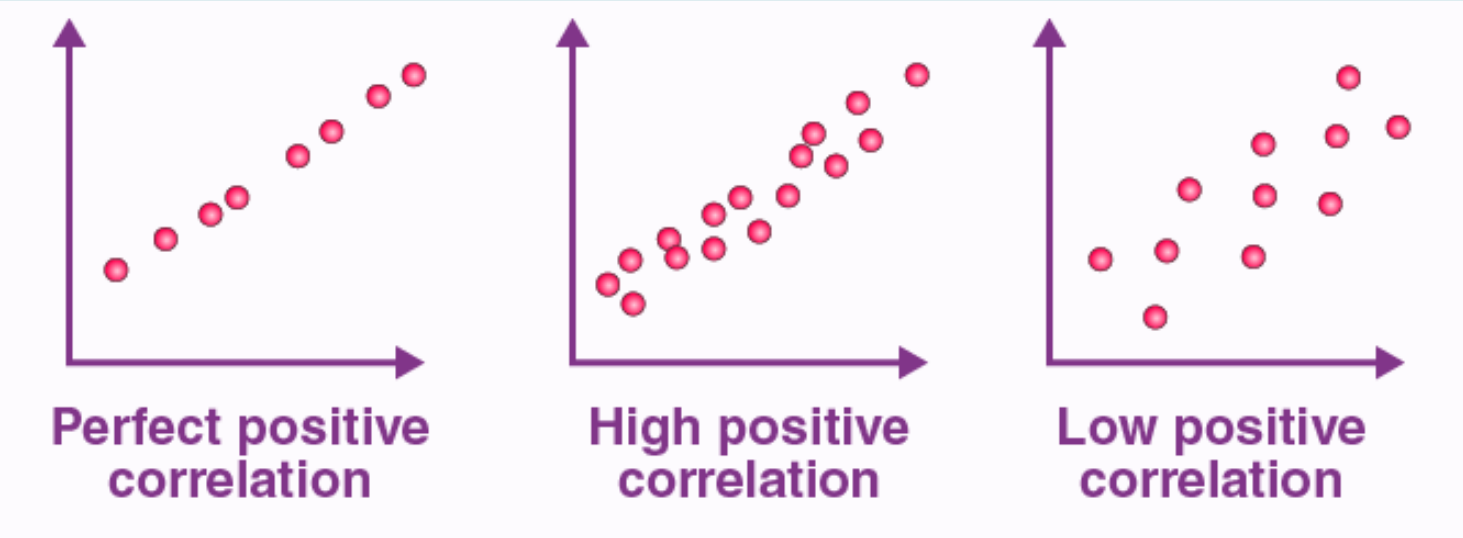
The scatter plot explains the correlation between two attributes or variables. It represents how closely the two variables are connected. There can be three such situations to see the relation between the two variables –

- Positive Correlation
- Negative Correlation
- No Correlation

Positive Correlation:

When the points in the graph are rising, moving from left to right, then the scatter plot shows a positive correlation. It means the values of one variable are increasing with respect to another. Now positive correlation can further be classified into three categories:

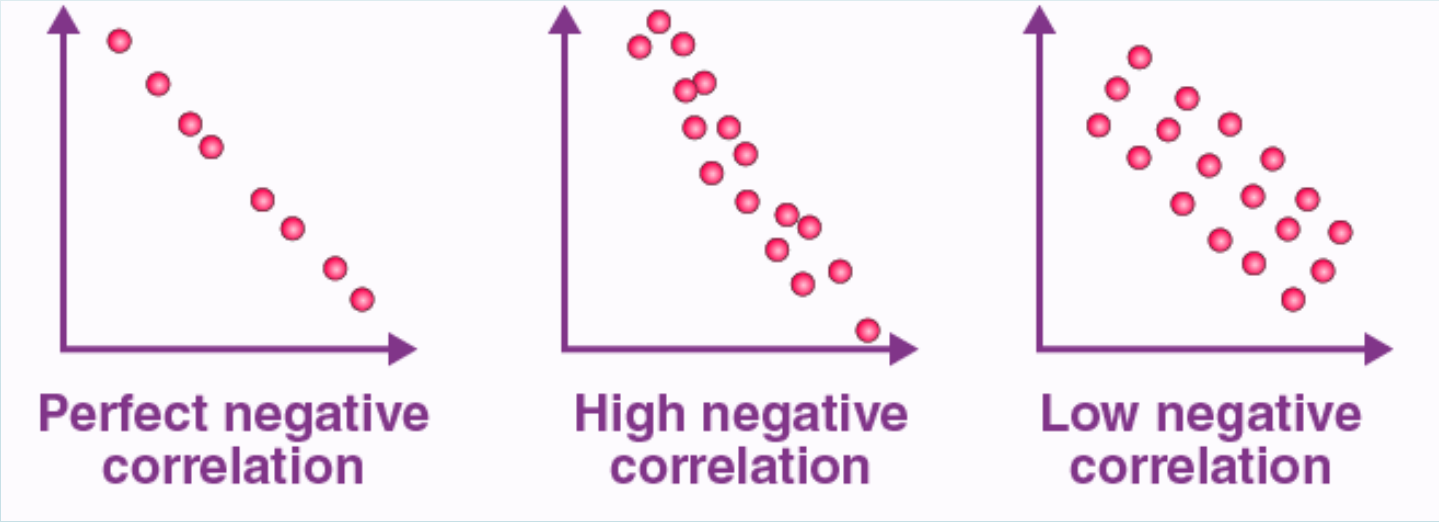
- Perfect Positive – Which represents a perfectly straight line
- High Positive – All points are nearby
- Low Positive – When all the points are scattered



Negative Correlation

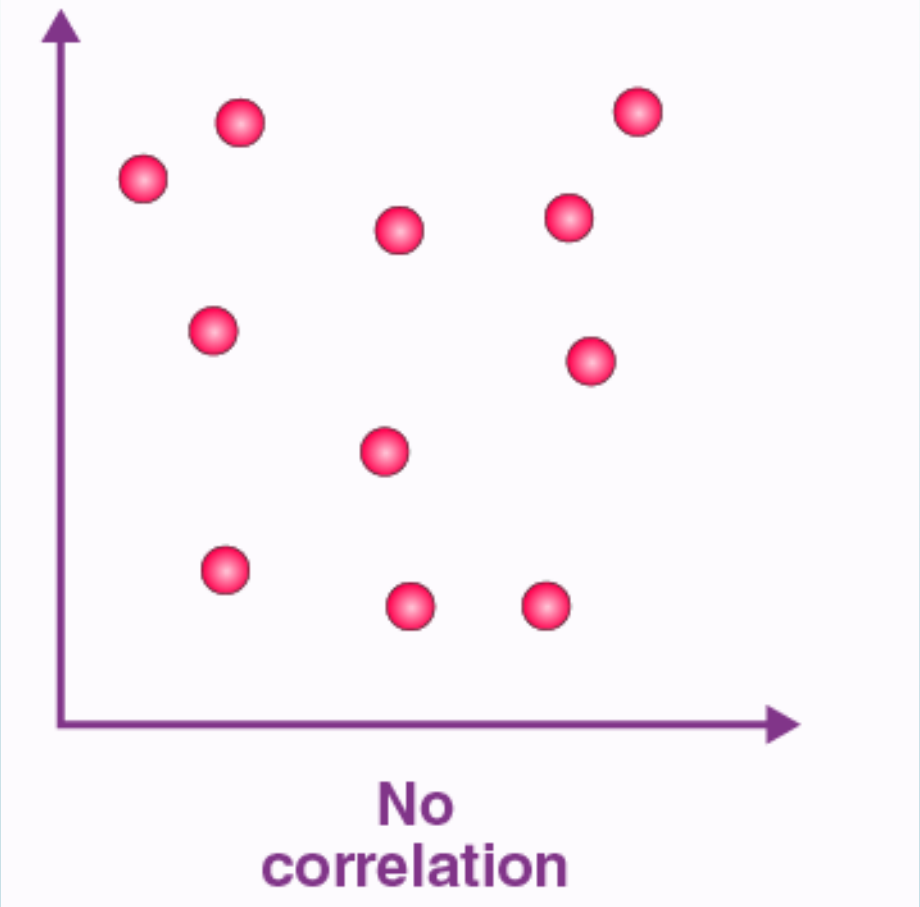
When the points in the scatter graph fall while moving left to right, then it is called a negative correlation. It means the values of one variable are decreasing with respect to another. These are also of three types:

- Perfect Negative – Which form almost a straight line
- High Negative – When points are near to one another
- Low Negative – When points are in scattered form



NO Correlation:

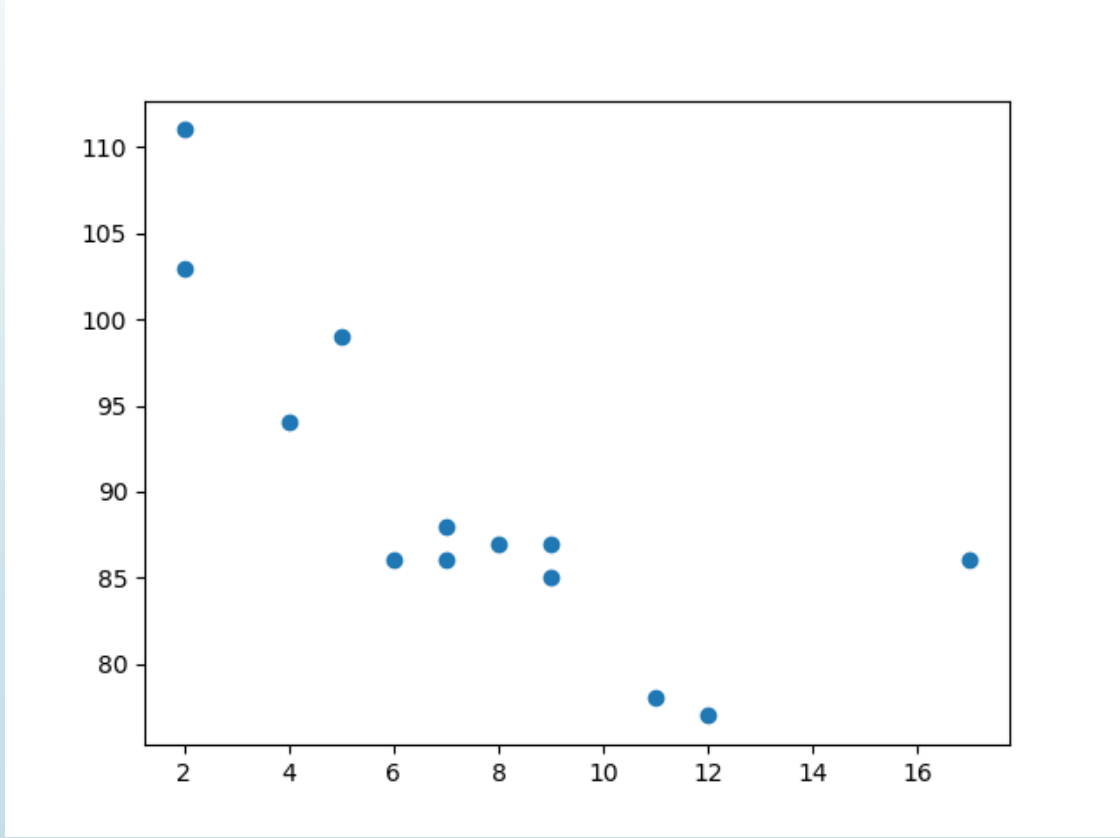
When the points are scattered all over the graph and it is difficult to conclude whether the values are increasing or decreasing, then there is no correlation between the variables.



```
import matplotlib.pyplot as plt

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

plt.scatter(x, y)
plt.show()
```



Thank You