Descriptive Statistics

Measures of Central Tendency

> In statistics, a central tendency (or measure of central tendency) is a central or typical value for a probability distribution. It may also be called a center or location of the distribution. The most common measures of central tendency are the arithmetic mean, the median, and the mode. - Wikipedia

Import required libraries

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from scipy.stats import kurtosis
from scipy.stats import skew
```

Load the data

In [2]:
```
import os

data_frame = pd.read_csv("data_sets/stats.csv")
data_frame
```

Out[2]:

	Name	Salary	Country
0	Dan	40000	USA
1	Elizabeth	32000	Brazil
2	Jon	45000	Italy
3	Maria	54000	USA
4	Mark	72000	USA
5	Bill	62000	Brazil
6	Jess	92000	Italy
7	Julia	55000	USA
8	Jeff	35000	Italy
9	Ben	48000	Brazil

Basic Statistics

No. of salaries present

In [3]:
```
data_frame["Salary"].count()
```

Out[3]:
```
10
```

Cumulative salaries

In [4]:
```
data_frame["Salary"].sum()
```

Out[4]:
```
535000
```

Cumulative salaries of various countries

In [5]:
```
data_frame.groupby(["Country"])["Salary"].sum()
```

Out[5]:
```
Country
Brazil     142000
Italy      172000
USA        221000
Name: Salary, dtype: int64
```

Entry count of various countries

In [6]:
```
data_frame.groupby(["Country"]).count()
```

Out[6]:

	Name	Salary
Country		
Brazil	3	3
Italy	3	3
USA	4	4

Mean

> In mathematics and statistics, the arithmetic mean, or simply the mean or the average (when the context is clear), is the sum of a collection of numbers divided by the count of numbers in the collection. - Wikipedia

In [7]:
```
data_frame["Salary"].mean()
```

Out[7]:
```
53500.0
```

Median

> In statistics and probability theory, the median is the value separating the higher half from the lower half of a data sample, a population, or a probability distribution. For a data set, it may be thought of as "the middle" value. The basic feature of the median in describing data compared to the mean (often simply described as the "average") is

> that it is not skewed by a small proportion of extremely large or small values, and therefore provides a better representation of a "typical" value. - Wikipedia

In [8]:
```
data_frame["Salary"].median()
```
Out[8]:
```
51000.0
```

Mode

> The mode is the value that appears most often in a set of data values. In other words, it is the value that is most likely to be sampled. - Wikipedia)

In [9]:
```
data_frame["Salary"].mode()
```
Out[9]:
```
0    32000
1    35000
2    40000
3    45000
4    48000
5    54000
6    55000
7    62000
8    72000
9    92000
dtype: int64
```

Dispersion

> In statistics, dispersion (also called variability, scatter, or spread) is the extent to which a distribution is stretched or squeezed. Common examples of measures of statistical dispersion are the variance, standard deviation, and interquartile range. - Wikipedia

Variance

> In probability theory and statistics, variance is the expectation of the squared deviation of a random variable from its population mean or sample mean. Variance is a measure of dispersion, meaning it is a measure of how far a set of numbers is spread out from their average value. - Wikipedia

In [10]:
```
data_frame["Salary"].var()
```
Out[10]:
```
332055555.5555556
```

Standard Deviation

> In statistics, the standard deviation is a measure of the amount of variation or dispersion of a set of values. A low standard deviation indicates that the values tend

> to be close to the mean (also called the expected value) of the set, while a high standard deviation indicates that the values are spread out over a wider range - Wikipedia

In [11]:

```
data_frame["Salary"].std()
```

Out[11]:

```
18222.391598128816
```

Skewness

> In probability theory and statistics, skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. The skewness value can be positive, zero, negative, or undefined. - Wikipedia

In [12]:

```
skewness = data_frame["Salary"].skew()

print(skewness)

if skewness < 0:
    print("The distribution is negatively skewed")
elif skewness > 0:
    print("The distribution is positively skewed")
else:
    print("The distribution is not skewed")
```

```
1.021551304801318
The distribution is positively skewed
```

Load the BirthWeight dataset

In [13]:

```
data_frame_2=pd.read_csv('data_sets/BirthWeight.csv')
data_frame_2.head()
```

Out[13]:

	Infant ID	Gestational Age (Weeks)	Birth Weight (Grams)
0	1	34.7	1895
1	2	36.0	2030
2	3	29.3	1440
3	4	40.1	2835
4	5	35.7	3090

Set Infant ID as the index

In [14]:

```
data_frame_2.set_index("Infant ID", inplace = True)
data_frame_2.head()
```

Out[14]:

Infant ID	Gestational Age (Weeks)	Birth Weight (Grams)
1	34.7	1895
2	36.0	2030
3	29.3	1440
4	40.1	2835
5	35.7	3090

Covariance

> In probability theory and statistics, covariance is a measure of the joint variability of two random variables. If the greater values of one variable mainly correspond with the greater values of the other variable, and the same holds for the lesser values (that is, the variables tend to show similar behavior), the covariance is positive. In the opposite case, when the greater values of one variable mainly correspond to the lesser values of the other, (that is, the variables tend to show opposite behavior), the covariance is negative. The sign of the covariance therefore shows the tendency in the linear relationship between the variables. - Wikipedia

In [15]:

```
data_frame_2.cov()
```

Out[15]:

	Gestational Age (Weeks)	Birth Weight (Grams)
Gestational Age (Weeks)	9.963824	1798.025
Birth Weight (Grams)	1798.025000	485478.750

Correlation

> In statistics, correlation or dependence is any statistical relationship, whether causal or not, between two random variables or bivariate data. In the broadest sense correlation is any statistical association, though it actually refers to the degree to which a pair of variables are linearly related. Familiar examples of dependent phenomena include the correlation between the height of parents and their offspring, and the correlation between the price of a good and the quantity the consumers are willing to purchase, as it is depicted in the so-called demand curve. - Wikipedia

In [16]:

```
data_frame_2.corr(method = "pearson")
```

Out[16]:

	Gestational Age (Weeks)	Birth Weight (Grams)
Gestational Age (Weeks)	1.000000	0.817519

	Gestational Age (Weeks)	Birth Weight (Grams)
Birth Weight (Grams)	0.817519	1.000000

Load the diamonds dataset

In [17]:

```
pd.set_option("display.max_columns",None)  # to display all the columns
pd.options.display.float_format = "{:,.2f}".format

data_frame_3 = pd.read_csv("data_sets/diamonds.csv")
data_frame_3.head()
```

Out[17]:

	id	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.50	55.00	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.80	61.00	326	3.89	3.84	2.31
2	3	0.23	Good	E	VS1	56.90	65.00	327	4.05	4.07	2.31
3	4	0.29	Premium	I	VS2	62.40	58.00	334	4.20	4.23	2.63
4	5	0.31	Good	J	SI2	63.30	58.00	335	4.34	4.35	2.75

In [18]:

```
data_frame_4 = data_frame_3.drop(["id"], axis = 1)
for column in data_frame_4:
    if data_frame_4[column].dtype == "object":
        data_frame_4.drop([column], axis = 1, inplace = True)

stats_of_data_frame_4 = data_frame_4.describe()
stats_of_data_frame_4.rename(index = {"50%":"Median/50%"}, inplace = True)
stats_of_data_frame_4
```

Out[18]:

	carat	depth	table	price	x	y	z
count	53,940.00	53,940.00	53,940.00	53,940.00	53,940.00	53,940.00	53,940.00
mean	0.80	61.75	57.46	3,932.80	5.73	5.73	3.54
std	0.47	1.43	2.23	3,989.44	1.12	1.14	0.71
min	0.20	43.00	43.00	326.00	0.00	0.00	0.00
25%	0.40	61.00	56.00	950.00	4.71	4.72	2.91
Median/50%	0.70	61.80	57.00	2,401.00	5.70	5.71	3.53
75%	1.04	62.50	59.00	5,324.25	6.54	6.54	4.04
max	5.01	79.00	95.00	18,823.00	10.74	58.90	31.80

In [19]:

```
var = data_frame_4.var()

var_list = []
for col in data_frame_4.columns:
    if data_frame_4[col].dtype == "object":
        continue
```

```
    var_list.append(round(data_frame_4[col], 5))


data_frame_5 = pd.DataFrame([var_list], columns = stats_of_data_frame_4.columns, index
stats_of_data_frame_5 = stats_of_data_frame_4.append(data_frame_5)
stats_of_data_frame_5
```

Out[19]:

	carat	depth	table	price	x	y	z
count	53,940.00	53,940.00	53,940.00	53,940.00	53,940.00	53,940.00	53,940.00
mean	0.80	61.75	57.46	3,932.80	5.73	5.73	3.54
std	0.47	1.43	2.23	3,989.44	1.12	1.14	0.71
min	0.20	43.00	43.00	326.00	0.00	0.00	0.00
25%	0.40	61.00	56.00	950.00	4.71	4.72	2.91
Median/50%	0.70	61.80	57.00	2,401.00	5.70	5.71	3.53
75%	1.04	62.50	59.00	5,324.25	6.54	6.54	4.04
max	5.01	79.00	95.00	18,823.00	10.74	58.90	31.80
var	0 0.23 1 0.21 2 0.23 3 ...	0 61.50 1 59.80 2 56.90 3 ...	0 55.00 1 61.00 2 65.00 3 ...	0 326 1 326 2 327 3 ...	0 3.95 1 3.89 2 4.05 3 ...	0 3.98 1 3.84 2 4.07 3 ...	0 2.43 1 2.31 2 2.31 3 ...

In [ ]: