Course name: **Python programming and analytics by Rahul Sir**

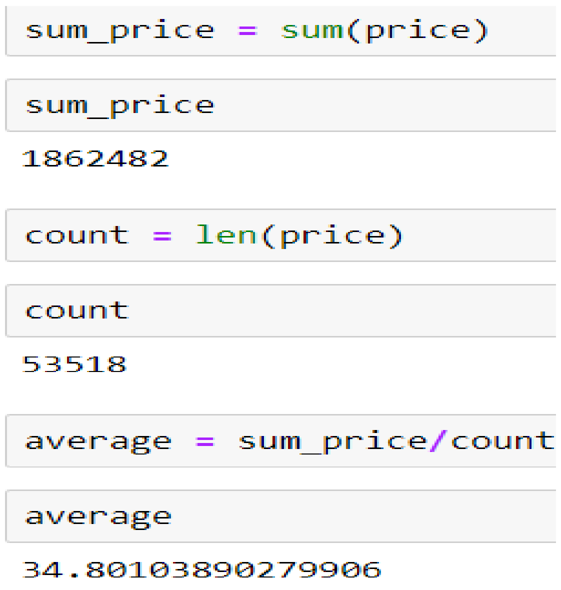
Topic name: **boxplot, histogram (Manav batch)**

Video name: **correlation, boxplot, histogram, outlier, standard deviation, variance Manav batch**

Video length: **1 hour 12 minutes 10 seconds**

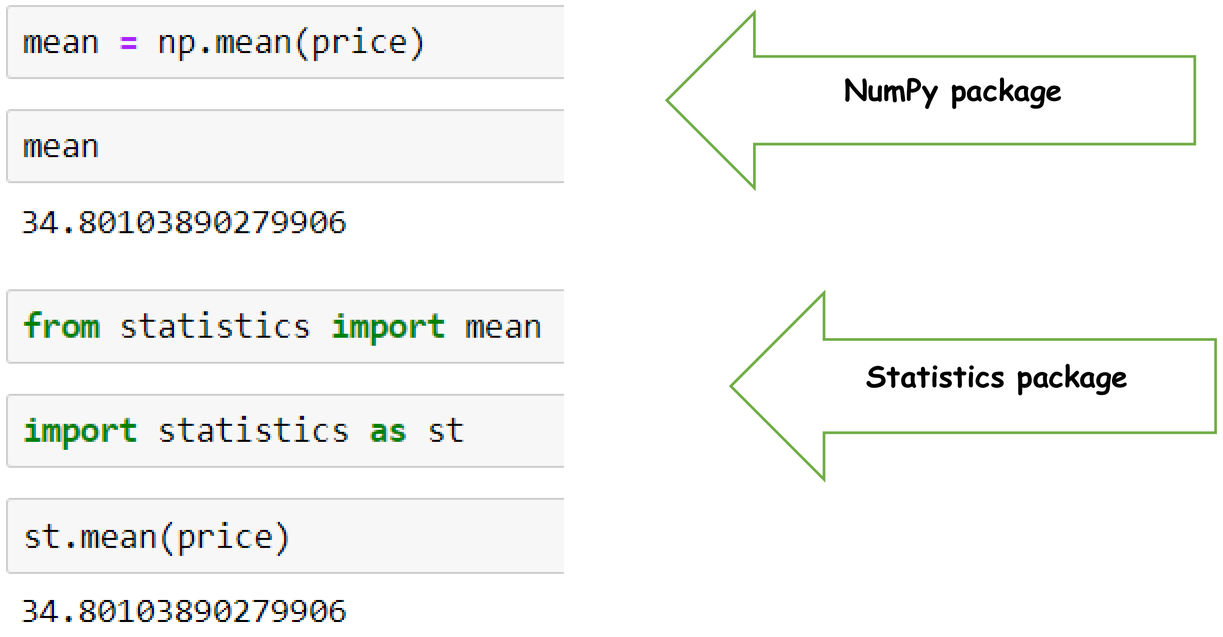
Mean:

The mean (average) of a data set is found by adding all numbers in the data set and then dividing by the number of values in the set. If you want to find the mean of the price column in the above pictured data, then first you need to find the sum and then count and then for mean, divide the sum by count. Like this,



```
sum_price = sum(price)

sum_price

1862482

count = len(price)

count

53518

average = sum_price/count

average

34.80103890279906
```

With the help of packages,



```
mean = np.mean(price)
```

```
mean
```

```
34.80103890279906
```

```
from statistics import mean
```

```
import statistics as st
```

```
st.mean(price)
```

```
34.80103890279906
```

statistics.**harmonic_mean**(*data*)

Return the harmonic mean of *data*, a sequence or iteration of real-valued numbers.

The harmonic mean, sometimes called the sub-contrary mean, is the reciprocal of the arithmetic mean() of the reciprocals of the data. For example, the harmonic mean of three values *a*, *b* and *c* will be equivalent to 3/(1/a + 1/b + 1/c). If one of the values is zero, the result will be zero.

The harmonic mean is a type of average, a measure of the central location of the data. It is often appropriate when averaging rates or ratios, for example speeds.

statistics.**geometric_mean**(*data*)

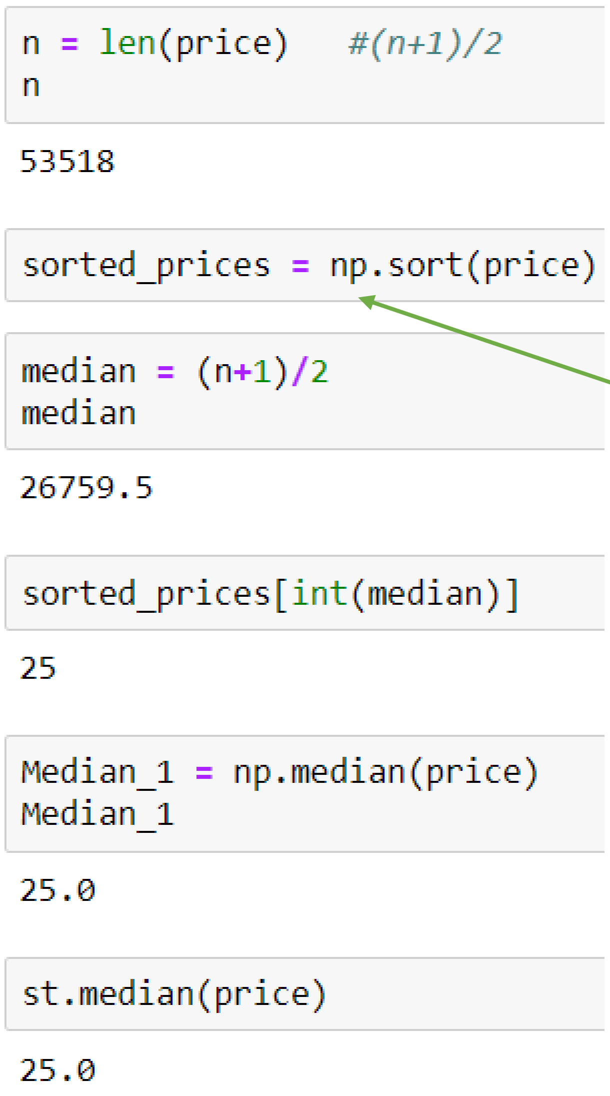
Convert *data* to floats and compute the geometric mean.

The geometric mean indicates the central tendency or typical value of the *data* using the product of the values (as opposed to the arithmetic mean which uses their sum).

Raises a StatisticsError if the input dataset is empty, if it contains a zero, or if it contains a negative value. The *data* may be a sequence or iteration.

Median:

The median is a simple measure of central tendency. To find the median, we arrange the observations in order from smallest to largest value. If there is an odd number of observations, the median is the middle value. If there is an even number of observations, the median is the average of the two middle values.



```
n = len(price)    #(n+1)/2
n
```

```
53518
```

```
sorted_prices = np.sort(price)
```

```
median = (n+1)/2
median
```

```
26759.5
```

```
sorted_prices[int(median)]
```

```
25
```

```
Median_1 = np.median(price)
Median_1
```

```
25.0
```

```
st.median(price)
```

```
25.0
```

Before median calculation, you need to sort the data either in ascending order or descending order

Mode: Mode is the value which occurs most frequently in a set of observations. Simply put, it is the number which is repeated most, i.e. the number with the highest frequency.

```
from statistics import mode
Mode = mode(price)
```

```
Mode
```

```
20
```

Outliers: Outliers are data values that differ greatly from the majority of a set of data. These values fall outside of an overall trend that is present in the data. Strong Outliers show extreme deviation from the rest of a data set. If a data value is an outlier, but not a strong outlier, then we say that the value is a weak outlier.

```
min_price = min(price)
max_price = max(price)

print(min_price, max_price)
```

```
4 2300
```

Range: The difference between the lowest and highest values.

```
price_range = max_price - min_price
print(price_range)
```

```
2296
```

Quartile and **inter-quartile range** : The interquartile range (IQR) is a measure of variability, based on dividing a data set into quartiles.

Quartiles divide a rank-ordered data set into four equal parts. The values that divide each part are called the first, second, and third quartiles; and they are denoted by Q1, Q2, and Q3, respectively.

- Q1 is the "middle" value in the *first* half of the rank-ordered data set.
- Q2 is the median value in the set.
- Q3 is the "middle" value in the *second* half of the rank-ordered data set.

The interquartile range is equal to Q3 minus Q1

```
print(price.quantile([0.25]))
print(price.quantile([0.50]))
print(price.quantile([0.75]))
print(price.quantile([1]))
print(price.quantile([0]))
```

```
0.25    17.0
Name: price, dtype: float64
0.5    25.0
Name: price, dtype: float64
0.75    40.0
Name: price, dtype: float64
1.0    2300.0
Name: price, dtype: float64
0.0    4.0
Name: price, dtype: float64
```

```
price_quantile_Q1 = price.quantile([0.25])
price_quantile_Q3 = price.quantile([0.75])
```

```
IQR = price_quantile_Q3[0.75]-price_quantile_Q1[0.25]
IQR
```

```
23.0
```

Standard deviation: The standard deviation is also a measure of the spread of your observations, but is a statement of how much your data deviates from a typical data point. That is to say, the standard deviation summarizes how much your data differs from the mean. This relationship to the mean is apparent in standard deviation's calculation.

Standard deviation is a measure that is used to quantify the amount of variation or dispersion of a set of data values.

```
print(stdev(price))
```

```
41.453611192987836
```

```
std_dev = np.std(price)
```

```
std_dev
```

```
41.45322390455368
```

```
import statistics
from statistics import stdev
```

```
stdev(price)
```

```
41.45361119298847
```

Variance: Variance and standard deviation are almost the exact same thing! Variance is just the square of the standard deviation. Likewise, variance and standard deviation represent the same thing — a measure of spread — but it’s worth noting that the units are different. Whatever units your data are in, standard deviation will be the same, and variation will be in that units-squared.

variance is the expectation of the squared deviation of a random variable from its mean

```
variance = np.square(std_dev)
```

```
variance
```

```
1718.3697720810608
```

```
from statistics import variance
variance(price)
```

```
1718.4018809394586
```

```
data.dtypes
```

```
SN            int64
country      object
points        int64
price         int64
province     object
region_1     object
variety      object
winery       object
dtype: object
```

Data.dtypes is used when you want to find the data type of each column in your data

```
data.describe()
```

	SN	points	price
count	53518.00000	53518.000000	53518.000000
mean	26759.50000	88.186106	34.801039
std	15449.46019	3.102856	41.453611
min	1.00000	80.000000	4.000000
25%	13380.25000	86.000000	17.000000
50%	26759.50000	88.000000	25.000000
75%	40138.75000	90.000000	40.000000
max	53518.00000	100.000000	2300.000000

Data.describe() function is used to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values.

How to create Boxplot?

Boxplot is a graph that gives you a good indication of how the values in the data are spread out. Boxplots are a standardized way of displaying the distribution of data based on "minimum", first quartile (Q1), median, third quartile (Q3), and "maximum".

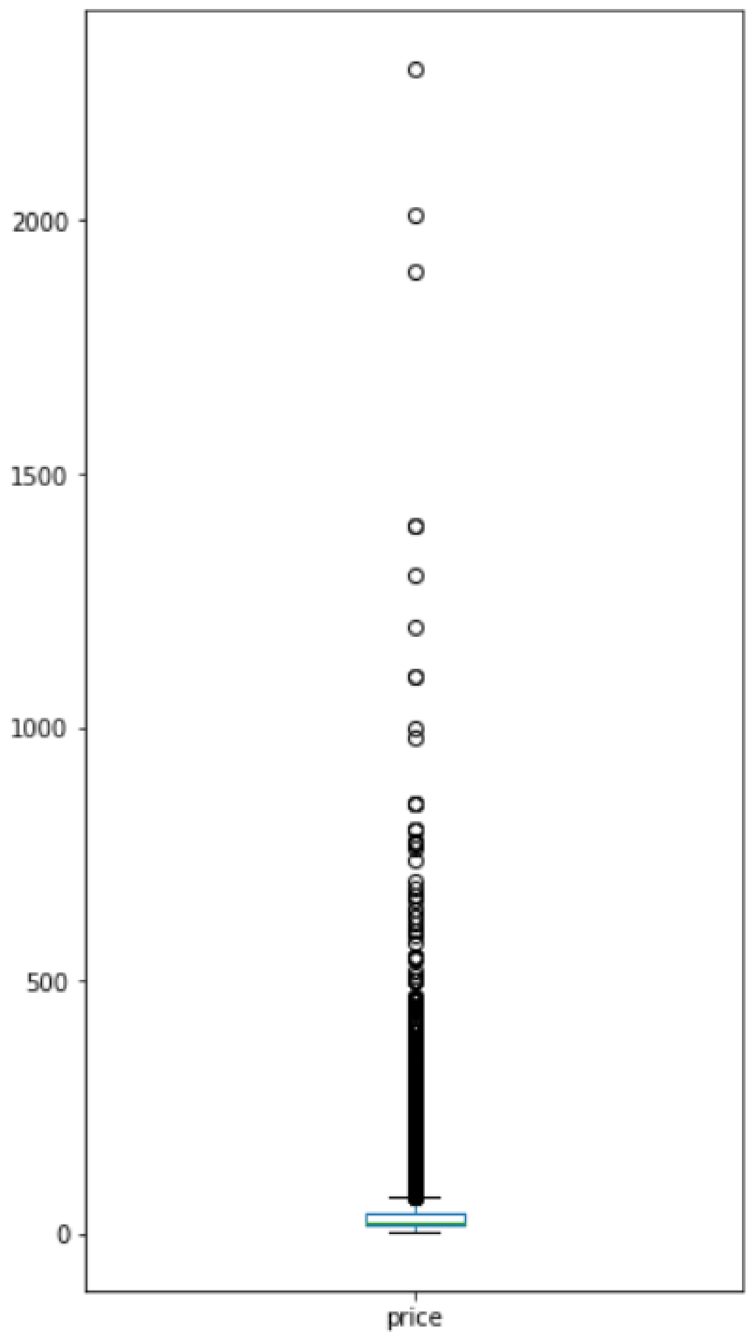
Code:

```python
import matplotlib.pyplot as plt
plt.figure(figsize = (5, 10))
data['price'].plot(kind ="box")
```

Import **matplotlib** for creating graphs

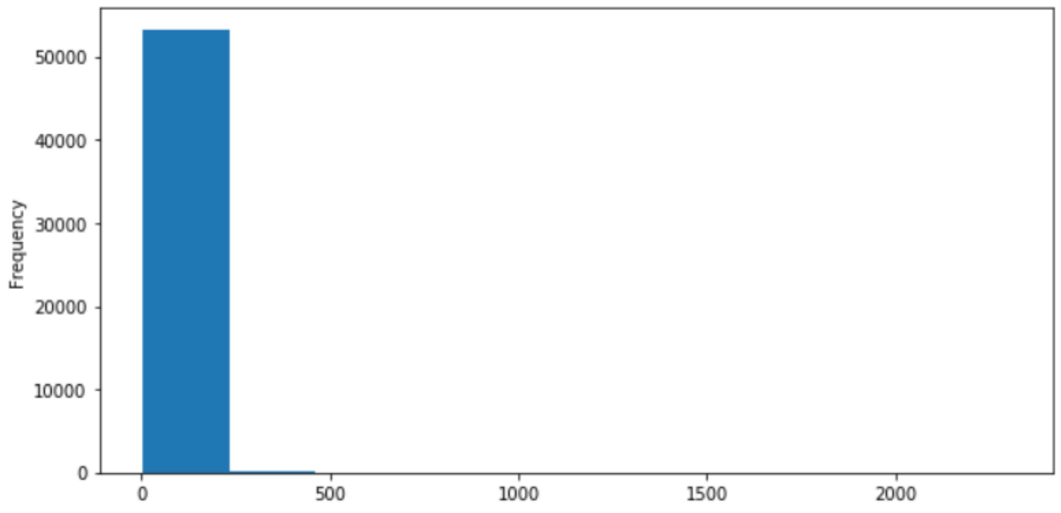
Output:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e5d9fd2400>
```


2000
1500
1000
500
0
price

```
plt.figure(figsize = (10, 5))
data['price'].plot(kind ="hist")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x15ca71d0>
```



Matplotlib and seaborn:

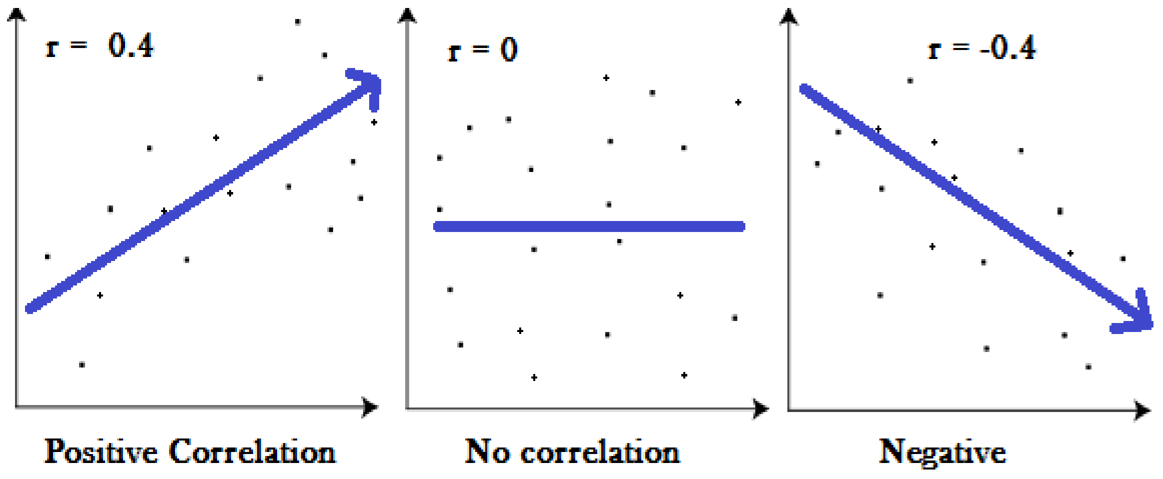
Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc., with just a few lines of code

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

Correlation:

Correlation is used to test relationships between quantitative variables or categorical variables. In other words, it's a measure of how things are related. The study of how variables are correlated is called correlation analysis

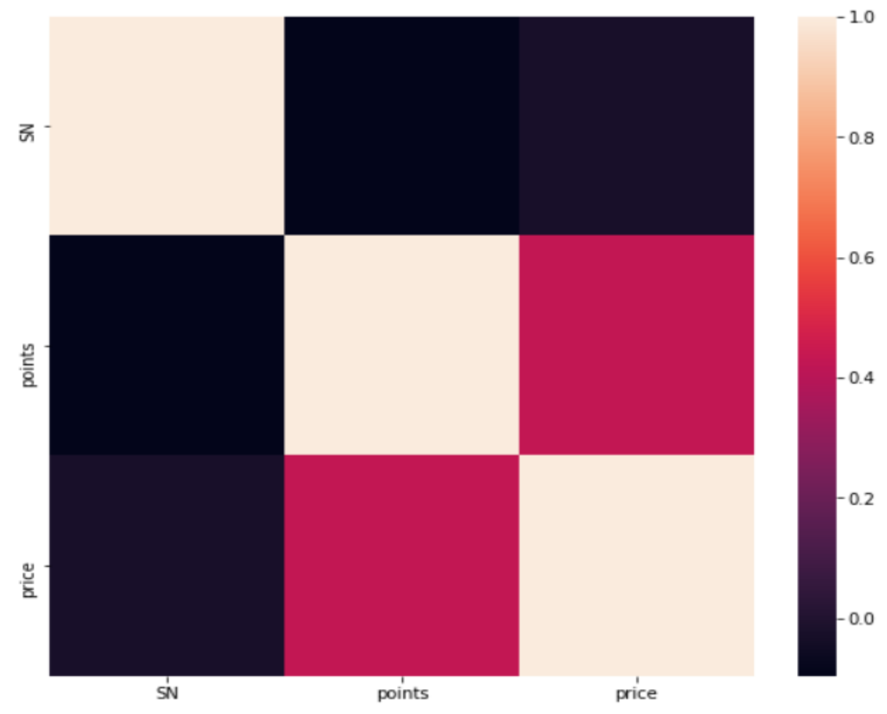
A correlation coefficient is a way to put a value to the relationship. Correlation coefficients have a value of between -1 and 1. A "0" means there is no relationship between the variables at all, while -1 or 1 means that there is a perfect negative or positive correlation (negative or positive correlation here refers to the type of graph the relationship will produce).



Creating a correlation matrix using seaborn:

```
import seaborn as sns
corrmat = data.corr()

f, ax = plt.subplots(figsize =(9, 8))
sns.heatmap(corrmat)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x14a4b0b8>
```



Seaborn used to create correlation matrix which allows to quickly observe the relationship between every variable of your matrix.

In the above pictured correlation matrix, you can see that the correlation between the two same columns shows pale pink color which means 1.0 value that is perfect correlation. correlation between the SN and points column is 0.0 represented by black color means no correlation. The correlation between price and points column is around 0.4 denoted by dark pink color means medium correlation