Intro to Basic understanding of the data using Python

Credits - Image from Internet

Why Python for DA?

- Python is very easy to get started.
- Python is a good fit since it has all the requirements available for Data Analysis and Data Science.
- Get more from less.
- The community of Python is so large that you can find anything easily.

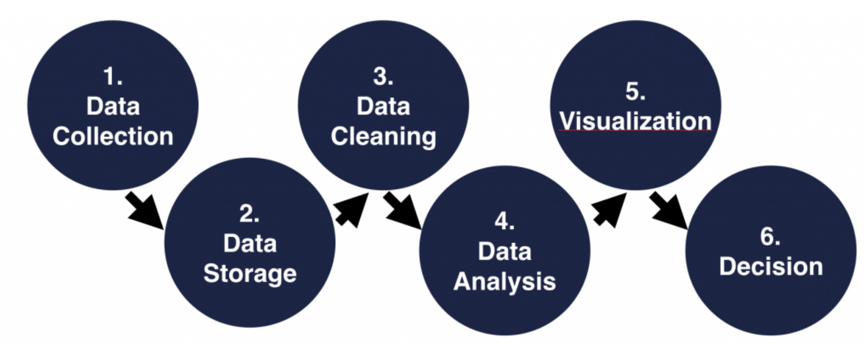
Note - Check google trends for comaprison.

- Before clicking the below link, please sign-in your google account on web.
- Link → https://trends.google.com/trends/explore?cat=1227&q=%2Fm%2F05z1_,%2Fm%2F0212jm,%2Fm%2F0j3djl7

" But when it comes to speed compatibility, Python is slower. "

- Link → https://juliacomputing.com/blog/2020/06/fast-csv/

Data Analysis vs Data Science



Credits - Image from Internet

Data Analyst's role

As a Data Analyst, we often need to do the following things -

- Data Collection
- Data Cleaning
- Data Organaising
- Data Managing
- Identify the goal of the problem statement
- Finding insights in order achieve the goal

Questions like -

- WHY
- HOW
- WHEN

Data Scientist's role

Data Science is story telling process with the valid insights acquired from data to information. Thus helping to take decisions.

- Design Data Models
- Create or use Algorithms
- Predict the future outcomes with accuracy
- Make decisions from the conclusions

More information - https://www.northeastern.edu/graduate/blog/what-does-a-data-scientist-do/

Note - Data Scientist with anlaytical skills is a `Blessing upon the blessed`.

In [ ]:

In [ ]:

Practise problem

1. Collect data from online using Pandas.
2. Check if data cleaning is necessary.
   ## - `yes` → Clean the data
   - `no` → Proceed
3. Identify the relationship between data varaibles.
   - Apply Correlation
   - Plot the relationship

- **Data Source** → http://wiki.stat.ucla.edu/socr/index.php/SOCR_Data_Dinov_020108_HeightsWeights

In [ ]:

In [ ]:

1. Collect data from online

1. Pandas is python library mainly used for data analysis.
2. It is similar to doing analysis on Excel but just we write code here.
3. It is one of the best open source libraries avalibale for doing data manipulation and data wrangling.

More information → https://pandas.pydata.org/

In [1]:
```python
import pandas as pd
```

`read_html()` extracts all the tables from the html page.

In [2]:
```python
data_source = 'http://wiki.stat.ucla.edu/socr/index.php/SOCR_Data_Dinov_020108_HeightsWeights'
data = pd.read_html(data_source)
```

In [3]:
```python
type(data)
```
Out[3]:
```
list
```

In [4]:
```python
len(data)
```
Out[4]:
```
3
```

In [5]:
```python
# data[0]
```

In [6]:
```python
# data[1]
```

In [7]:
```python
# data[2]
```

In [8]:
```python
df = data[1]
df.head()
```
Out[8]:

	Index	Height(Inches)	Weight(Pounds)
0	1	65.78	112.99
1	2	71.52	136.49
2	3	69.40	153.03
3	4	68.22	142.34
4	5	67.79	144.30

In [ ]:

In [ ]:

2. Check if data cleaning is necessary

Data Cleaning is one of the important aspects in both Data Analysis and Data Science roles.

- It is one of the procedural steps where a data analyst or data scientist spends most of their time.

More information → https://en.wikipedia.org/wiki/Data_cleansing

PS: Do not data cleaning for granted.

a. Check for any `NaN` values → Missing values

In [9]:
```python
df.isna()
```
Out[9]:

	Index	Height(Inches)	Weight(Pounds)
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
...	...	...	...
195	False	False	False
196	False	False	False
197	False	False	False
198	False	False	False
199	False	False	False

200 rows × 3 columns

- Since the dataset is sort of big, we cannot see all the values. Infact we cannot comprehend the actual missing values from the `isna()` dataset.
- In order to get the actual values (indices), the below function can be used.

In [10]:
```python
# check if there is any missing data

def check_for_nan(dframe):
    """
    dframe → pandas data frame object
    returns `nan_places` a dictionary of column names and the `nan_indices`
    """
    nan_frame = dframe.isna()
    d_cols = dframe.columns

    nan_places = {}
    for col in d_cols:
        col_lvals = nan_frame[col].to_list()
        nan_indices = [index for index, val in enumerate(col_lvals) if val == True]
        if nan_indices:
            nan_places[col] = nan_indices
        else:
            nan_places[col] = None

    return nan_places
```

In [11]:
```python
check_for_nan(dframe=df)
```
Out[11]:
```
{'Index': None, 'Height(Inches)': None, 'Weight(Pounds)': None}
```

Above result is clear, every column has `non-nan` values. Hence we can proceed with further steps.

b. Check for the `datatypes` from each column

In [12]:
```python
df.info()
```
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Index           200 non-null    int64
 1   Height(Inches)  200 non-null    float64
 2   Weight(Pounds)  200 non-null    float64
dtypes: float64(2), int64(1)
memory usage: 4.8 KB
```

Seems like every column has a unique data type. If at all there is then it is required to purify the data - make sure all the values are of same type.

c. Overall description of the data frame

In [13]:
```python
df.describe()
```
Out[13]:

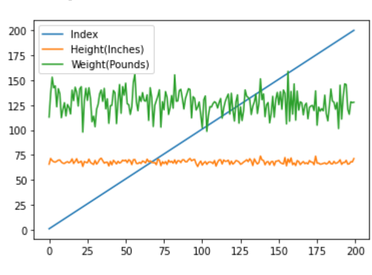
	Index	Height(Inches)	Weight(Pounds)
count	200.000000	200.000000	200.000000
mean	100.500000	67.949800	127.221950
std	57.879185	1.940363	11.960959
min	1.000000	63.430000	97.900000
25%	50.750000	66.522500	119.895000
50%	100.500000	67.935000	127.875000
75%	150.250000	69.202500	136.097500
max	200.000000	73.900000	158.960000

d. Some visualization to explore more about the data

- We can use pandas plotting functions like `plot()` to explore about the data visually.
- `plot()` can show the following plots -
  - **line** → line plot (default)
  - **bar** → vertical bar plot
  - **barh** → horizontal bar plot
  - **hist** → histogram
  - **box** → boxplot
  - **kde** → Kernel Density Estimation plot
  - **density** → same as 'kde'
  - **area** → area plot
  - **pie** → pie plot
  - **scatter** → scatter plot
  - **hexbin** → hexbin plot.

Ugly plot example

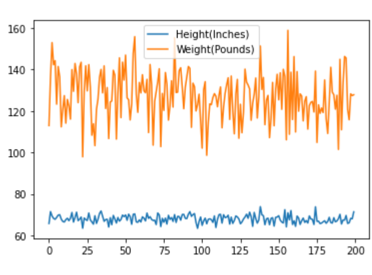
In [14]:
```python
df.plot()
```
Out[14]:
```
<AxesSubplot:>
```



The above is the plot of all the data variables. This is not something we should do.

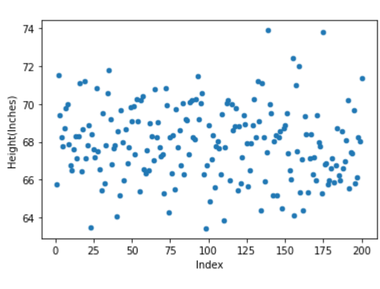
Plotting without unimportant data variables - excluded `Index`

In [15]:
```python
df[['Height(Inches)', 'Weight(Pounds)']].plot()
```
Out[15]:
```
<AxesSubplot:>
```



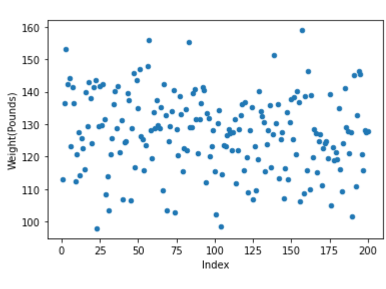
The above is the plot of both `Heights` and `Weights` from the data frame `df`.

In [16]:
```python
df.plot(x='Index', y='Height(Inches)', kind='scatter')
```
Out[16]:
```
<AxesSubplot:xlabel='Index', ylabel='Height(Inches)'>
```



The above is the scatter plot of `Index` and `Heights` data variables from the data frame `df`.

In [17]:
```python
df.plot(x='Index', y='Weight(Pounds)', kind='scatter')
```
Out[17]:
```
<AxesSubplot:xlabel='Index', ylabel='Weight(Pounds)'>
```



The above is the scatter plot of `Index` and `Weights` data variables from the data frame `df`.

In [ ]:

In [ ]:

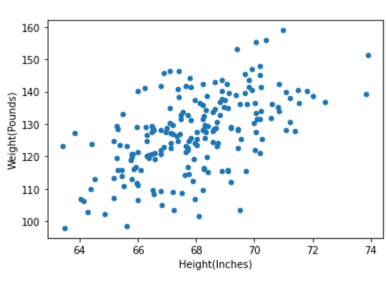
3. Relationship between data variables

Correlation - one of the statistical measurements applied to find out if any two variables are linrealy related.

- If one varibles is increasing, then other variable also increases. Vice versa.
- For example
  - If income of an employee increases then the household expenses increase.
  - If income of an employee decreases then the household expenses decrease.
- Scatter plot is really helpful to find the relationship between two variables. With this, it can be easily noticed the linear trend as well.

b. Plot the relationship

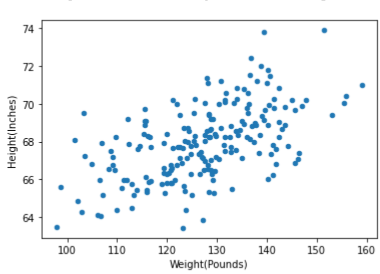
In [18]:
```python
df.plot(x='Height(Inches)', y='Weight(Pounds)', kind='scatter')
```
Out[18]:
```
<AxesSubplot:xlabel='Height(Inches)', ylabel='Weight(Pounds)'>
```



From the above plot, we can see that when `Heights` increase, then `Weights` also increased.

What if we interchange the values?

In [19]:
```python
df.plot(y='Height(Inches)', x='Weight(Pounds)', kind='scatter')
```
Out[19]:
```
<AxesSubplot:xlabel='Weight(Pounds)', ylabel='Height(Inches)'>
```



a. Find the Correlation

Correlation value ranges from `-1` to `1`.

- If the calculated correlation value is -
  - **-1**, then it is perfectly **negative correlation**
  - **1**, then it is perfectly **positive correlation**
  - **< -1**, then it means that **error** in the correlation measurement
  - **> 1**, then it means that **error** in the correlation measurement

More information → https://www.investopedia.com/terms/c/correlationcoefficient.asp

In [20]:
```python
df.corr()
```
Out[20]:

	Index	Height(Inches)	Weight(Pounds)
Index	1.000000	-0.094260	-0.128882
Height(Inches)	-0.094260	1.000000	0.556865
Weight(Pounds)	-0.128882	0.556865	1.000000

In [21]:
```python
relation = df.corr()
relation.style.background_gradient(cmap='viridis_r')
```
Out[21]:

	Index	Height(Inches)	Weight(Pounds)
Index	1.000000	-0.094260	-0.128882
Height(Inches)	-0.094260	1.000000	0.556865
Weight(Pounds)	-0.128882	0.556865	1.000000

Correlation plots based on the correlation value obtained. → https://en.wikipedia.org/wiki/Correlation_and_dependence#/media/File:Correlation_examples2.svg

In [ ]:

In [ ]:

Well, the above results are obtained for the data which was already stored.

But, what about Streaming data?

Streaming data - that data that is continuously generated by different sources is called streaming data.

- For example -
  - Tesla Self-driving Car generates the data continuously.
  - One tesla car generates 11 TB and 152 TB data per day.
    - Big data problem
  - More information → https://www.tuxera.com/blog/autonomous-and-adas-test-cars-produce-over-11-tb-of-data-per-day/

In [ ]:

In [ ]:

Case Study → Activity

1. Select any one of these or you can find your own topic of interest not specifically from below.
    - Study to analyse peoples' habits on YouTube platform
    - Study to analyse the changes occurred in peoples' life due to Demonotization
    - Study to analyse the students' overall development due to online education
2. Create a google form where you can have a set of questions and answer options.
    - Have atleast 8 to 10 questions
3. Collect the data from your friends, families etc (by sharing the link).
    - The data will be stored in your drive (in a spreasheet)
4. Once the data is collected -
    - Create your own data variables from the questions
    - Try to basic analysis like processing and visualization

Note - To learn how to create google forms (Questionnaires) and collect the data,

- please watch this video → **https://www.youtube.com/watch?v=vQw2jDIyIDU**