DATE: 18/01/2024

EXPERIMENT NO.1

AIM:

- Introduction to Data Science and Python
- Installation of Jupyter Notebook

SOFTWARE USED: Jupyter Notebook

THEORY: Data Science is about data gathering, analysis and decision-making.Data Science is about finding patterns in data, through analysis, and make future predictions.By using Data Science, companies are able to make:

- Better decisions (should we choose A or B)
- Predictive analysis (what will happen next?)
- Pattern discoveries (find pattern, or maybe hidden information in the data)

A Data Scientist requires expertise in several backgrounds:

- Machine Learning
- Statistics
- Programming (Python or R)
- Mathematics
- Databases

A Data Scientist must find patterns within the data. Before he/she can find the patterns, he/she must organize the data in a standard format.

NumPy (Numerical Python):

- NumPy is a fundamental package for scientific computing in Python.
- It provides support for multidimensional arrays, matrices, and mathematical functions.
- NumPy arrays are used extensively in data manipulation and numerical computations.

pandas:

- pandas is a powerful data manipulation and analysis library.
- It provides DataFrame objects for handling structured data and Series objects for one-dimensional data.
- pandas offers functionalities for reading and writing data from various file formats, data cleaning, filtering, grouping, and aggregation.

Matplotlib:

- matplotlib is a versatile plotting library for creating static, interactive, and animated visualizations in Python.
- It offers a MATLAB-like interface for generating plots and charts.
- matplotlib can be used for exploratory data analysis, visualization of statistical results, and communication of findings.

Seaborn:

- Seaborn is a statistical data visualization library based on matplotlib.
- It provides high-level functions for creating informative and attractive statistical graphics.
- Seaborn simplifies the creation of complex visualizations such as scatter plots, histograms, heatmaps, and regression plots.

scikit-learn:

- scikit-learn is a comprehensive machine learning library for Python.
- It includes various algorithms for classification, regression, clustering, dimensionality reduction, and model selection.
- scikit-learn provides a consistent interface for model training, evaluation, and deployment, making it suitable for both beginners and experts in machine learning.

SciPy (Scientific Python):

- SciPy is a collection of mathematical algorithms and functions built on top of NumPy.
- It offers modules for optimization, interpolation, integration, linear algebra, signal processing, and more.
- SciPy complements NumPy by providing additional scientific computing capabilities.

statsmodels:

- statsmodels is a library for statistical modeling and hypothesis testing in Python.
- It includes a wide range of statistical models, such as linear regression, generalized linear models, time series analysis, and ANOVA.
- statsmodels is particularly useful for conducting hypothesis tests, estimating parameters, and performing statistical inference.

Installing Jupyter Notebook using Anaconda is straightforward and recommended for users who want a hassle-free installation process along with a comprehensive Python distribution. Here's how to do it:

1. Download Anaconda:Visit the Anaconda website and download the appropriate Anaconda installer for your operating system (Windows, macOS, or Linux).

2. Install Anaconda:Run the downloaded Anaconda installer and follow the installation instructions provided in the Anaconda installer wizard. Make sure to select the option to add Anaconda to your system PATH during the installation process.
3. Launch Anaconda Navigator:Once Anaconda is installed, you can launch Anaconda Navigator, which is a graphical user interface (GUI) that provides access to various tools and applications included in the Anaconda distribution.
4. Open Jupyter Notebook:In Anaconda Navigator, locate and launch Jupyter Notebook from the list of available applications. This will open Jupyter Notebook in your default web browser, allowing you to start creating and running Jupyter notebooks immediately.
5. Create a New Notebook:Once Jupyter Notebook is opened in your web browser, you can create a new notebook by clicking on the "New" button and selecting "Python 3" (or any other available kernel) from the dropdown menu. This will create a new notebook where you can write and execute Python code, add markdown cells for documentation, and create visualizations.

OUTPUT CODE:

In [1]:
```
#Experiment no. 1: Introduction to Data Science and Python modules and installing jupyter notebook
#Subject :DSA lab
```

In [ ]:
```
import numpy as np
```

In [2]:
```
b = np.empty(2, dtype=int)
print(b)
c = np.empty([2,2], dtype=int)
print(c)
d = np.empty([3,3], dtype=float)
print(d)
```

```
[1065353216 1065353216]
[[  780542668 -1229774382]
 [ 1475572606  -435929117]]
[[0.00000000e+000 0.00000000e+000 0.00000000e+000]
 [0.00000000e+000 0.00000000e+000 7.94457559e-321]
 [1.24610723e-306 1.29061142e-306 5.53353523e-322]]
```

In [3]:
```
a = np.zeros(2)
print(a)
```

```
[0. 0.]
```

In [4]:
```
c = np.array([1,2,3,4])
d = np.array([5,6,7,8])
e = np.add(c,d)
print(e)
```

```
[ 6  8 10 12]
```

In [5]:
```
f = np.multiply(c,d)
print(f)
```

```
[ 5 12 21 32]
```

In [6]:
```
g = np.arange(10,4,-1)
print(g)
```

```
[10  9  8  7  6  5]
```

In [7]:
```
import pandas as pd
```

In [8]:
```
data = pd.read_csv('2019.csv')
data.head()
```

Out[8]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298

In [9]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 9 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Overall rank                  156 non-null    int64  
 1   Country or region             156 non-null    object 
 2   Score                         156 non-null    float64
 3   GDP per capita                156 non-null    float64
 4   Social support                156 non-null    float64
 5   Healthy life expectancy       156 non-null    float64
 6   Freedom to make life choices  156 non-null    float64
 7   Generosity                    156 non-null    float64
 8   Perceptions of corruption     156 non-null    float64
dtypes: float64(7), int64(1), object(1)
memory usage: 11.1+ KB
```

In [10]:
```
data.tail()
```

Out[10]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
151	152	Rwanda	3.334	0.359	0.711	0.614	0.555	0.217	0.411
152	153	Tanzania	3.231	0.476	0.885	0.499	0.417	0.276	0.147
153	154	Afghanistan	3.203	0.350	0.517	0.361	0.000	0.158	0.025
154	155	Central African Republic	3.083	0.026	0.000	0.105	0.225	0.235	0.035
155	156	South Sudan	2.853	0.306	0.575	0.295	0.010	0.202	0.091

In [11]:
```
data.shape
```

Out[11]:
```
(156, 9)
```

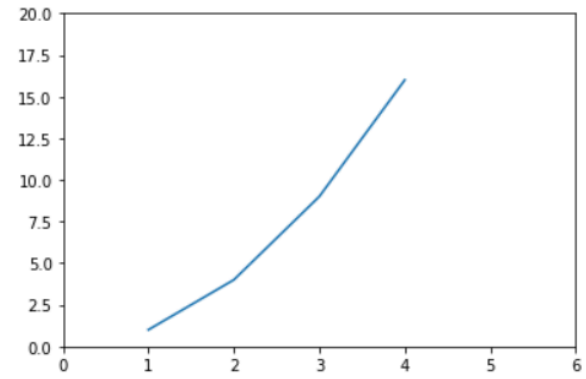
In [12]:
```
data.describe()
```

Out[12]:

	Overall rank	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
count	156.000000	156.000000	156.000000	156.000000	156.000000	156.000000	156.000000	156.000000
mean	78.500000	5.407096	0.905147	1.208814	0.725244	0.392571	0.184846	0.110603
std	45.177428	1.113120	0.398389	0.299191	0.242124	0.143289	0.095254	0.094538
min	1.000000	2.853000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	39.750000	4.544500	0.602750	1.055750	0.547750	0.308000	0.108750	0.047000
50%	78.500000	5.379500	0.960000	1.271500	0.789000	0.417000	0.177500	0.085500
75%	117.250000	6.184500	1.232500	1.452500	0.881750	0.507250	0.248250	0.141250
max	156.000000	7.769000	1.684000	1.624000	1.141000	0.631000	0.566000	0.453000

In [13]:
```
import matplotlib.pyplot as plt
```

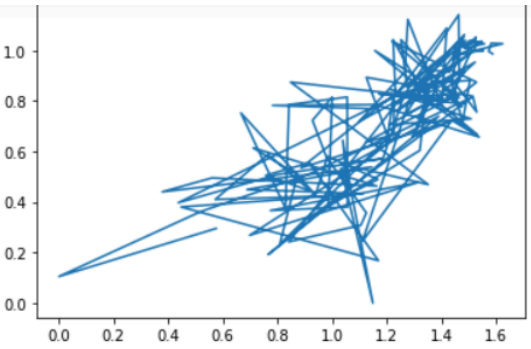
In [14]:
```
plt.plot([1,2,3,4],[1,4,9,16])
plt.axis([0,6,0,20])
plt.show()
```



In [17]:
```
plt.plot(data['Social support'],data['Healthy life expectancy'])
plt.axis(0,3)
plt.show()
```

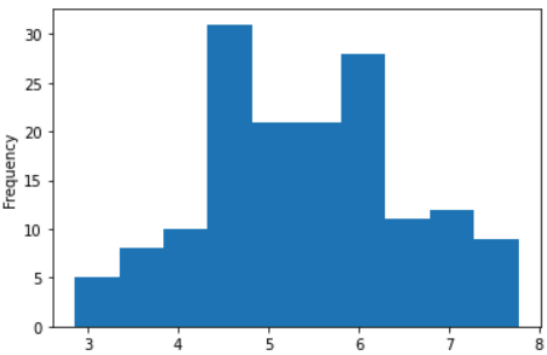
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [17], in <cell line: 2>()
      1 plt.plot(data['Social support'],data['Healthy life expectancy'])
----> 2 plt.axis(0,3)
      3 plt.show()

TypeError: axis() takes from 0 to 1 positional arguments but 2 were given
```



In [18]:
```
data.Score.plot(kind='hist')
```

Out[18]:
```
<Axes: ylabel='Frequency'>
```

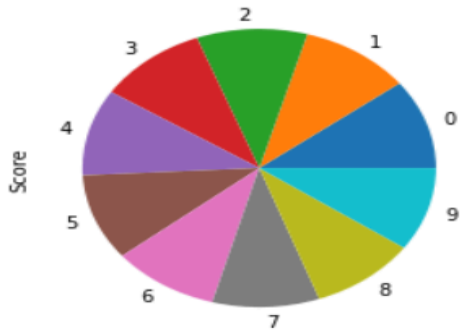


In [19]:
```
data.isnull().sum()
```

Out[19]:
```
Overall rank                    0
Country or region               0
Score                           0
GDP per capita                  0
Social support                  0
Healthy life expectancy         0
Freedom to make life choices    0
Generosity                      0
Perceptions of corruption       0
dtype: int64
```

In [20]:
```
data.Score[0:10].plot(kind='pie')
```

Out[20]:
```
<Axes: ylabel='Score'>
```



In [21]:
```
coca = pd.read_excel('CocaCola_Sales_Rawdata.xlsx')
```

In [22]:
```
coca.describe()
```

Out[22]:

	Sales
count	42.000000
mean	2994.353308
std	977.930896
min	1547.818996
25%	2159.714247
50%	2782.376999
75%	3609.250000
max	5253.000000

In [23]:
```
coca.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 42 entries, 0 to 41
Data columns (total 2 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Quarter  42 non-null     object
 1   Sales    42 non-null     float64
dtypes: float64(1), object(1)
memory usage: 800.0+ bytes
```

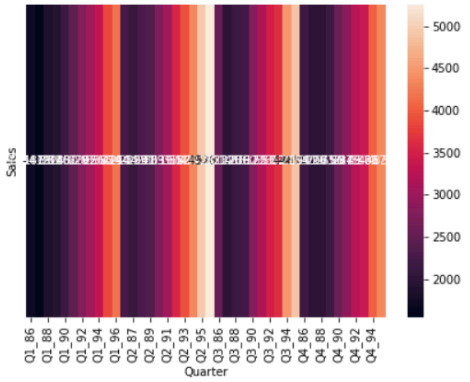
In [24]:
```
from pandas.plotting import lag_plot
```

```
In [25]: v = pd.read_excel('CocaCola_Sales_Rawdata.xlsx',header=0,index_col=0,parse_dates=True)

C:\Users\anjal\AppData\Local\Temp\ipykernel_31604\2971767921.py:1: UserWarning: Could not infer format, so each element will be
parsed individually, falling back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.
  v = pd.read_excel('CocaCola_Sales_Rawdata.xlsx',header=0,index_col=0,parse_dates=True)

In [30]: import seaborn as sns
plt.figure(figsize=(7,5))
plot_month_y = pd.pivot_table(data = coca,values='Sales',columns='Quarter',aggfunc='mean',fill_value=0)
sns.heatmap(plot_month_y,annot=True,fmt='g')

Out[30]: <Axes: xlabel='Quarter'>
```



CONCLUSION:

In summary, the installation of Jupyter Notebook using Anaconda streamlines the setup process for data science endeavors. This approach integrates the powerful features of Python with the interactive capabilities of Jupyter Notebook, facilitating seamless data manipulation, visualization, and modeling. By leveraging Anaconda's comprehensive Python environment, users can quickly dive into data analysis tasks, experiment with machine learning algorithms, and share insights with colleagues or the broader community.