ASSIGNMENT 1.B

- **Name : Achal Rajesh Mate**
- **Roll No : 2203541**
- **Enroll No : MITU20BTCSD001**
- **Branch : CSE**
- **Class : TY CSE Is - 3**
- **Batch : B**
- **Guided By : Prof Nagesh Jadhav Sir**

Title:- B. Installation, Confguration and checking the current Version of numpy, scipy, scikit, pandas and matplot lib using anaconda prompt and list their uses in machine learning

Objectives:- To explore ML libraries for pre-processing and data visualization

Theory:

Numpy

1. Numpy and important functions in numpy

- NumPy is a Python library, which stands for 'Numerical Python'.
- It is the core library for scientific computing, which contains a powerful n-dimensional array object, provide tools for integrating C, C++ etc.
- NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.
- It is the fundamental package for scientific computing with Python. It contains various features including these important ones:
  - A powerful N-dimensional array object
  - ophisticated (broadcasting) functions
  - Tools for integrating C/C++ and Fortran code
  - Useful linear algebra, Fourier transform, and random number capabilities
- NumPy array can also be used as an efficient multi-dimensional container for generic .data.
  - The ndarray (NumPy Array) is a multidimensional array used to store values of same datatype.
  - These arrays are indexed just like Sequences, starts with zero.
  - The ndarrays are better than regular arrays in terms of faster computation and ease of manipulation.
  - In different algorithms of Machine Learning like K-means Clustering, Random Forest etc. we have to store the values in an array. So, instead of using regular array, ndarray helps us to manipulate and execute easily

Install NumPy via pip command:

pip install numpy

Import Library

In [2]:

```
import numpy as np
```

Numpy ndarray

In [9]:

```
a=np.array([1,2,3]) #Single dimension array
print(a)
print(type(a))
```

```
[1 2 3]
<class 'numpy.ndarray'>
```

In [10]:

```
a=np.array([(1,2,3),(4,5,6)])# 2 Dimension array
print(a)
```

```
[[1 2 3]
 [4 5 6]]
```

In [11]:

```
#Check Number of Dimensions
a = np.array(42)
b = np.array([1, 2, 3, 4, 5])
c = np.array([[1, 2, 3], [4, 5, 6]])
d = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

print(a.ndim)
print(b.ndim)
print(c.ndim)
print(d.ndim)
```

```
0
1
2
3
```

Check Item Size

In [21]:

```
d = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])
print(d.itemsize)
```

```
4
```

Square Root & Standard Deviation

In [13]:

```
a=np.array([(1,2,3),(3,4,5,)])
print(np.sqrt(a))
print(np.std(a))
```

```
[[1.         1.41421356 1.73205081]
 [1.73205081 2.         2.23606798]]
1.2909944487358056
```

Numpy Random

In [14]:

```
print(np.random.rand(2,3))
```

```
[[0.55899119 0.57358781 0.00799057]
 [0.83003806 0.47845854 0.36721098]]
```

Numpy arange

In [16]:

```
print(np.arange(0,10,0.3))  #numpy.arange([start, ]stop, [step, ])
```

```
[0.  0.3 0.6 0.9 1.2 1.5 1.8 2.1 2.4 2.7 3.  3.3 3.6 3.9 4.2 4.5 4.8 5.1
 5.4 5.7 6.  6.3 6.6 6.9 7.2 7.5 7.8 8.1 8.4 8.7 9.  9.3 9.6 9.9]
```

Reshape ndarray

In [25]:

```
a = np.array([4,5,6,7,8,9,0,3,6]).reshape((3,3))
print(a)
```

```
[[4 5 6]
 [7 8 9]
 [0 3 6]]
```

Create an array of zeros

In [26]:

```
arr0 = np.zeros(10)
arr0
```

Out[26]:

```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

Create an array of ones

In [27]:

```
arr1 = np.ones(10)
arr1
```

Out[27]:

```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1.])
```

Pandas

2.Pandas and important functions in pandas

- Python Pandas is defined as an open-source library that provides high-performance data manipulation in Python.
- Pandas is defined as an open-source library that provides high-performance data manipulation in Python.
- The name of Pandas is derived from the word Panel Data, which means an Econometrics from Multidimensional data. It is used for data analysis in Python and developed by Wes McKinney in 2008.
- Data analysis requires lots of processing, such as restructuring, cleaning or merging, etc. There are different tools are available for fast data processing, such as Numpy, Scipy, Cython, and Panda. But we prefer Pandas because working with Pandas is fast, simple and more expressive than other tools.

Key Features of Pandas

- It has a fast and efficient DataFrame object with the default and customized indexing.
- Used for reshaping and pivoting of the data sets.
- Group by data for aggregations and transformations.
- It is used for data alignment and integration of the missing data.
- Provide the functionality of Time Series.
- Process a variety of data sets in different formats like matrix data, tabular heterogeneous, time series.
- Handle multiple operations of the data sets such as subsetting, slicing, filtering, groupBy, re-ordering, and re-shaping.
- It integrates with the other libraries such as SciPy, and scikit-learn.
- Provides fast performance, and If you want to speed it, even more, you can use the Cython.

Benefits of Pandas

- Data Representation: It represents the data in a form that is suited for data analysis through its DataFrame and Series.
- Clear code: The clear API of the Pandas allows you to focus on the core part of the code. So, it provides clear and concise code for the user.

Python Pandas Data Structure The Pandas provides two data structures for processing the data, i.e., Series and DataFrame
1) Series: It is defined as a one-dimensional array that is capable of storing various data types. The row labels of series are called the index. We can easily convert the list, tuple, and dictionary into series using "series' method. A Series cannot contain multiple * columns. It has one parameter: * Data: It can be any list, dictionary, or scalar value.
2) Python Pandas DataFrame: It is a widely used data structure of pandas and works with a two-dimensional array with labeled axes (rows and columns). DataFrame is defined as a standard way to store data and has two different indexes, i.e., row index and column index. It consists of the following properties:

The columns can be heterogeneous types like int, bool, and so on. It can be seen as a dictionary of Series structure where both the rows and columns are indexed. It is denoted as "columns" in case of columns and "index" in case of rows. Create a DataFrame using List:

- We can easily create a DataFrame in Pandas using list

Install Pandas Library

pip install pandas

Import pandas Libaray

In [29]:

```
import pandas as pd
```

pandas series

In [30]:

```
info = np.array(['P','a','n','d','a','s'])
a = pd.Series(info)
print(a)
```

```
0    P
1    a
2    n
3    d
4    a
5    s
dtype: object
```

In [31]:

```
s = pd.Series(["a", "b", "c"], name="vals")
s.to_frame()
```

Out[31]:

	vals
0	a
1	b
2	c

Pandas Dataframe

In [33]:

```
# Create dataframe
info = {'ID' :[101, 102, 103],'Department':['B.Sc','B.Tech','M.Tech',]}
df = pd.DataFrame(info)
df
```

Out[33]:

	ID	Department
0	101	B.Sc
1	102	B.Tech
2	103	M.Tech

In [35]:

```
# Create Dataframe from series
info = {'one' : pd.Series([1, 2, 3, 4, 5, 6], index=['a', 'b', 'c', 'd', 'e', 'f']),
   'two' : pd.Series([1, 2, 3, 4, 5, 6, 7, 8], index=['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h

d1 = pd.DataFrame(info)
d1
```

Out[35]:

	one	two
a	1.0	1
b	2.0	2
c	3.0	3
d	4.0	4
e	5.0	5
f	6.0	6
g	NaN	7
h	NaN	8

In [48]:

```
# check Nan values
d1.isnull().sum()
```

Out[48]:

```
one    2
two    0
dtype: int64
```

In [49]:

```
# Drop Null Values
d1.dropna()
```

Out[49]:

	one	two
a	1.0	1
b	2.0	2
c	3.0	3
d	4.0	4
e	5.0	5
f	6.0	6

In [36]:

```
# Astype
a = {'col1': [1, 2], 'col2': [3, 4]}
d2 = pd.DataFrame(data=a)
d2.dtypes
```

Out[36]:

```
col1    int64
col2    int64
dtype: object
```

In [40]:

```
# We convert it into 'int32' type.
d2.astype({'col2': 'int32'}).dtypes
```

Out[40]:

```
col1    int64
col2    int32
dtype: object
```

In [42]:

```
### GroupBy
data = {'Name': ['Achal', 'Mate', 'abc', 'pqr'],
   'Percentage': [98, 91, 91, 87],
   'Course': ['B.Sc','B.Ed','M.Phill','BA']}
df3 = pd.DataFrame(data)
df3
```

Out[42]:

	Name	Percentage	Course
0	Achal	98	B.Sc
1	Mate	91	B.Ed
2	abc	91	M.Phill
3	pqr	87	BA

In [47]:

```
grouped = df3.groupby('Course')
grouped[['Percentage']].agg(np.mean)
```

Out[47]:

	Percentage
Course	
B.Ed	91
B.Sc	98
BA	87
M.Phill	91

Matplotlib

3. Matplotlib and important functions in matplotlib

- Matplotlib is a visualization library in Python for 2D plots of arrays.
- It consists of several plots like line, bar, scatter, histogram etc.
- Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy tack.It can also be used with graphics toolkits like PyQt and wxPython.
- One of the advantage of visualization is that it allows us visual access to huge amounts of data in easily digestible isuals.
- Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy.
- As such, it offers a viable open source alternative to MATLAB. Developers can also use matplotlib's APIs (Application Programming Interfaces) to embed plots in GUI applications.

Install Matplotlib Library

pip install matplotlib

Importing matplotlib :

from matplotlib import pyplot as plt
or
import matplotlib.pyplot as plt

In [51]:

```
import matplotlib.pyplot as plt
```

#Dataset Link : https://drive.google.com/file/d/1asWu8inTpz6BxJfQLmCDIGIBiNkwUU36/view?usp=sharing (https://drive.google.com/file/d/1asWu8inTpz6BxJfQLmCDIGIBiNkwUU36/view?usp=sharing)

In [75]:

```
df = pd.read_csv(r"F:\New folder (2)\company_sales_data.csv")
```

In [76]:

```
df.head()
```

Out[76]:

	month_number	facecream	facewash	toothpaste	bathingsoap	shampoo	moisturizer	total_
0	1	2500	1500	5200	9200	1200	1500	2
1	2	2630	1200	5100	6100	2100	1200	1
2	3	2140	1340	4550	9550	3550	1340	2
3	4	3400	1130	5870	8870	1870	1130	2
4	5	3600	1740	4560	7760	1560	1740	2

LinePlot

In [56]:

```
profit = list(df.total_profit)
months = list(df['month_number'])
plt.title("profit of all months")
plt.xlabel("Months")
plt.ylabel("Profit")
plt.plot(months,profit, color='r',marker='p',markerfacecolor='k',linestyle='--', linewidth=
plt.legend(loc='best')
```

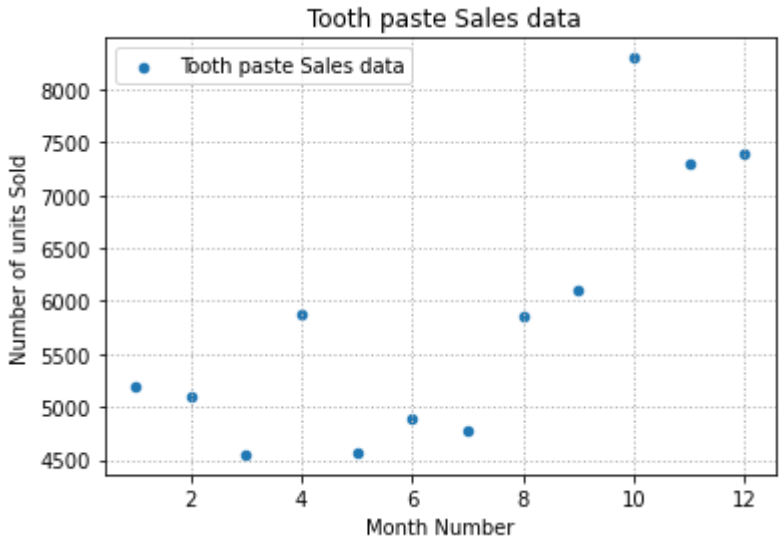
Out[56]:

```
<matplotlib.legend.Legend at 0x16aaf849130>
```

Scatter Plot

In [59]:

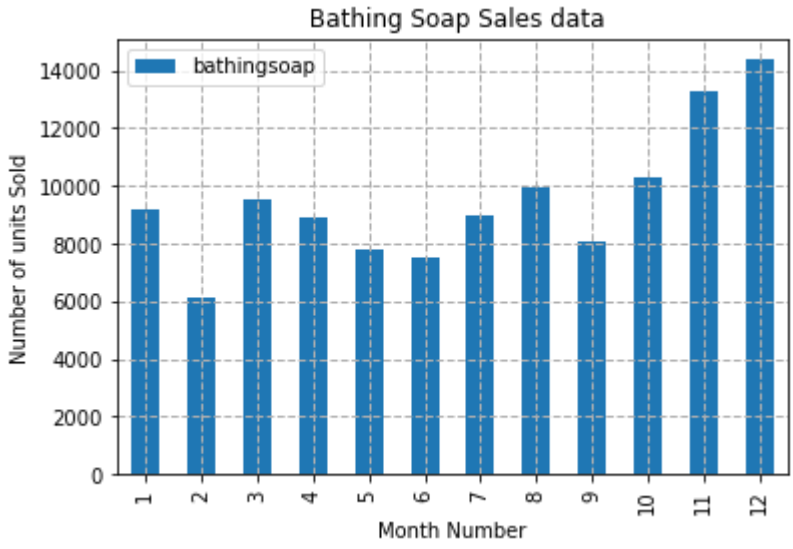
```
df.plot(x = 'month_number',y ='toothpaste' ,kind="scatter",label = 'Tooth paste Sales data'
plt.xlabel('Month Number')
plt.ylabel('Number of units Sold')
plt.legend(loc='upper left')
plt.title(' Tooth paste Sales data')
plt.grid(True, linewidth= 1, linestyle=":")
```



BarPlot

In [60]:

```
df.plot(x = 'month_number',y = ['bathingsoap'],kind = 'bar')
plt.xlabel('Month Number')
plt.ylabel('Number of units Sold')
plt.legend(loc='upper left')
plt.title(' Bathing Soap Sales data')
plt.grid(True, linewidth= 1, linestyle="--")
```



Seaborn

4. Seaborn and important functions in seaborn

- Seaborn is a library in Python predominantly used for making statistical graphics.
- Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python.
- Visualization is the central part of Seaborn which helps in exploration and understanding of data
- Seaborn helps you explore and understand your data.
- Its plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.

Install Package

pip install seaborn

Import Package

In [61]:

```
import seaborn as sns
```

Distplot

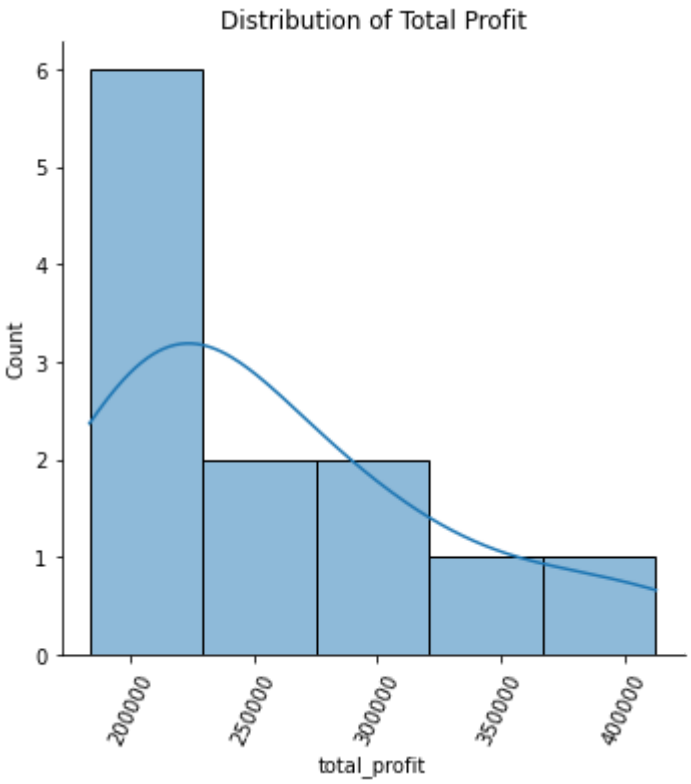
In [67]:

```
plt.figure(figsize = (13,8))
sns.displot(x="total_profit", kde=True,data=df)
plt.xticks(rotation=65)
plt.title("Distribution of Total Profit")
```

Out[67]:

```
Text(0.5, 1.0, 'Distribution of Total Profit')
```

```
<Figure size 936x576 with 0 Axes>
```



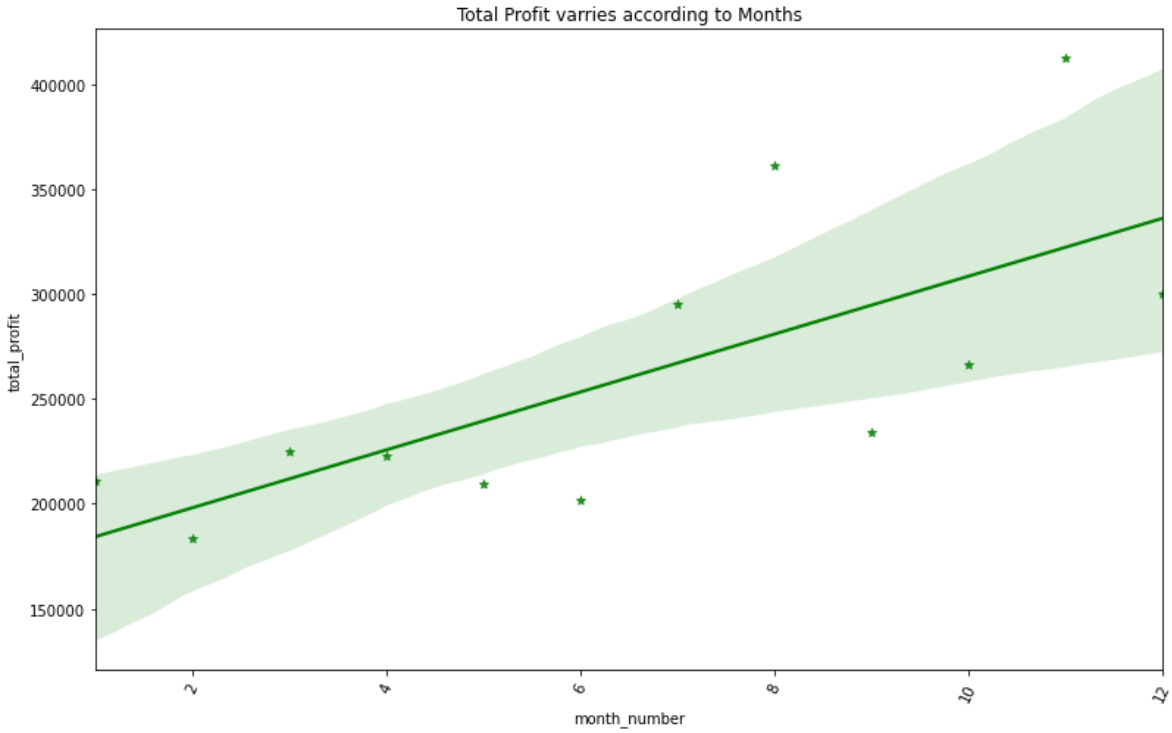
Regplot

In [70]:

```
plt.figure(figsize = (13,8))
sns.regplot(x="month_number",y="total_profit",data= df,color='green',marker='*')
plt.xticks(rotation=65)
plt.title("Total Profit varries according to Months")
```

Out[70]:

```
Text(0.5, 1.0, 'Total Profit varries according to Months')
```



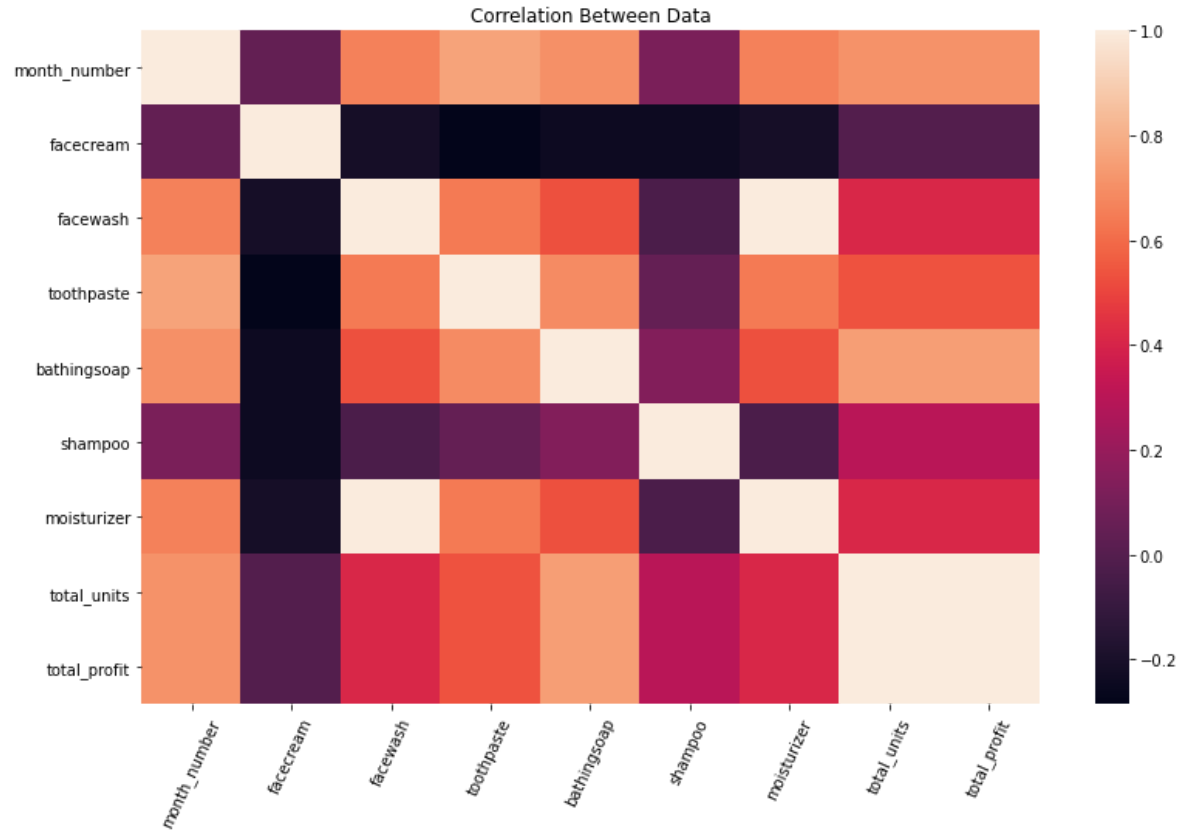
HeatMap

In [71]:

```
plt.figure(figsize = (13,8))
sns.heatmap(df.corr())
plt.xticks(rotation=65)
plt.title("Correlation Between Data")
```

Out[71]:

```
Text(0.5, 1.0, 'Correlation Between Data')
```



Scikit:-

4. scikit-learn and important functions in scikit-learn

- Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python.
- It provides a selection of efficient tools for machine learning and statistical modeling.
- This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.
- The functionality that scikit-learn provides include:
  - Regression, including Linear and Logistic Regression
  - Classification, including K-Nearest Neighbors
  - Clustering, including K-Means and K-Means++
  - Model selection
  - Preprocessing, including Min-Max Normalization.

Conclusion:

We have described the libraries , the installation of the libraries and pratice various examples base on libraries