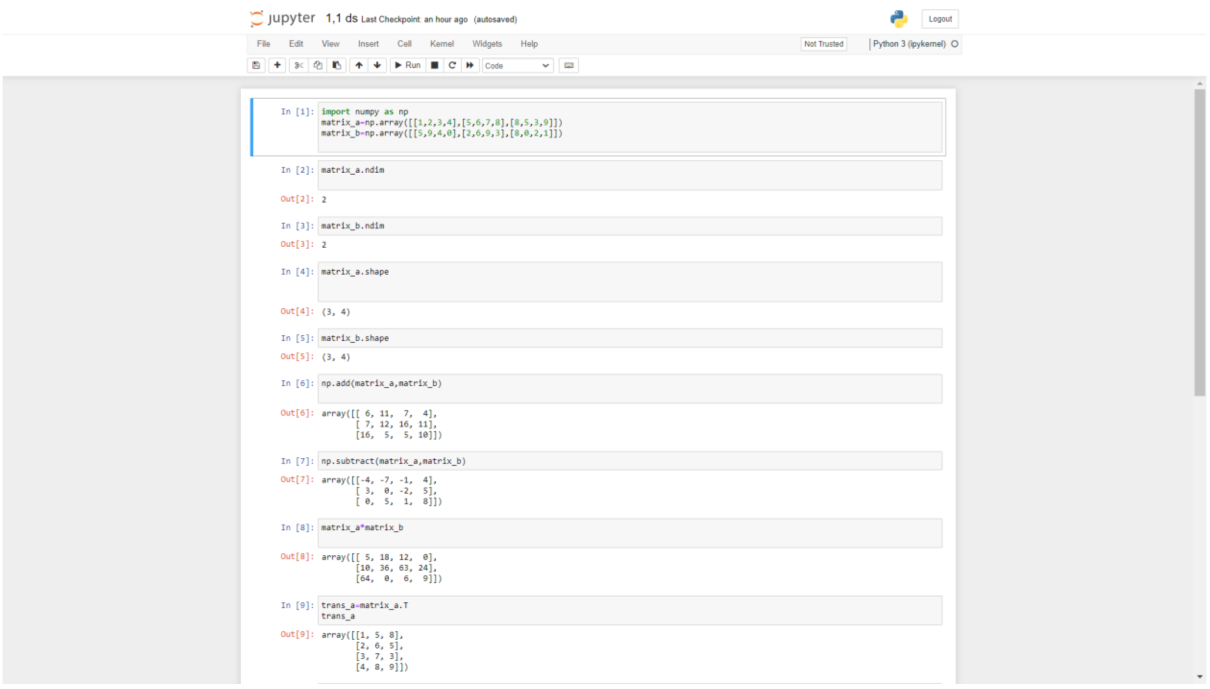
Machine Learning (20MCA241)

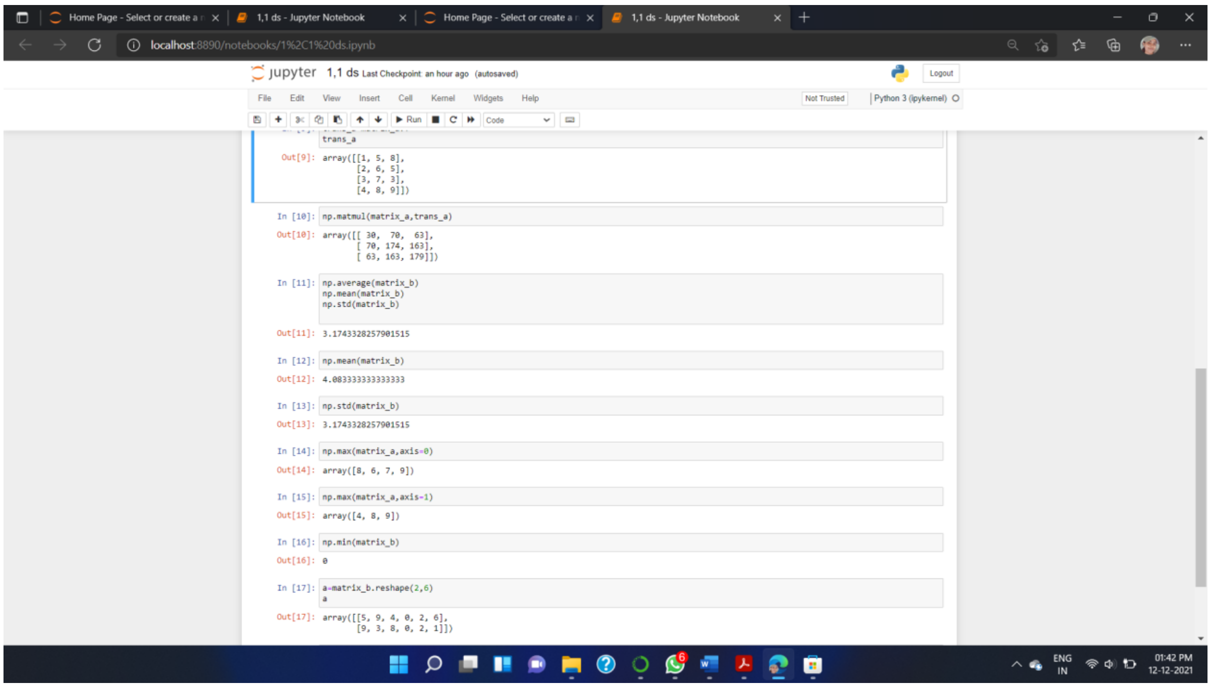
Assignment 1: Review of python programming

1)Review of Python Programming and Matrix Operations

1.1) Create two 3X4 matrices a and b using numpy arrays. Perform the following operations: Display the number of dimensions of the matrices a and b, find the shape of the matrices a and b, find a+b and a-b, a*b(elementwise), multiply the matrix a and its transpose (matrix multiplicaiton), add the value 10 to all elements of a, find transpose of b, calculate average, mean and standard deviation of the elements of the matrix b, find the maximum element in each column and each row of the matrix a, minimum value of the matrix b, reshape b with dimension 2x6 and find the transpose of a.

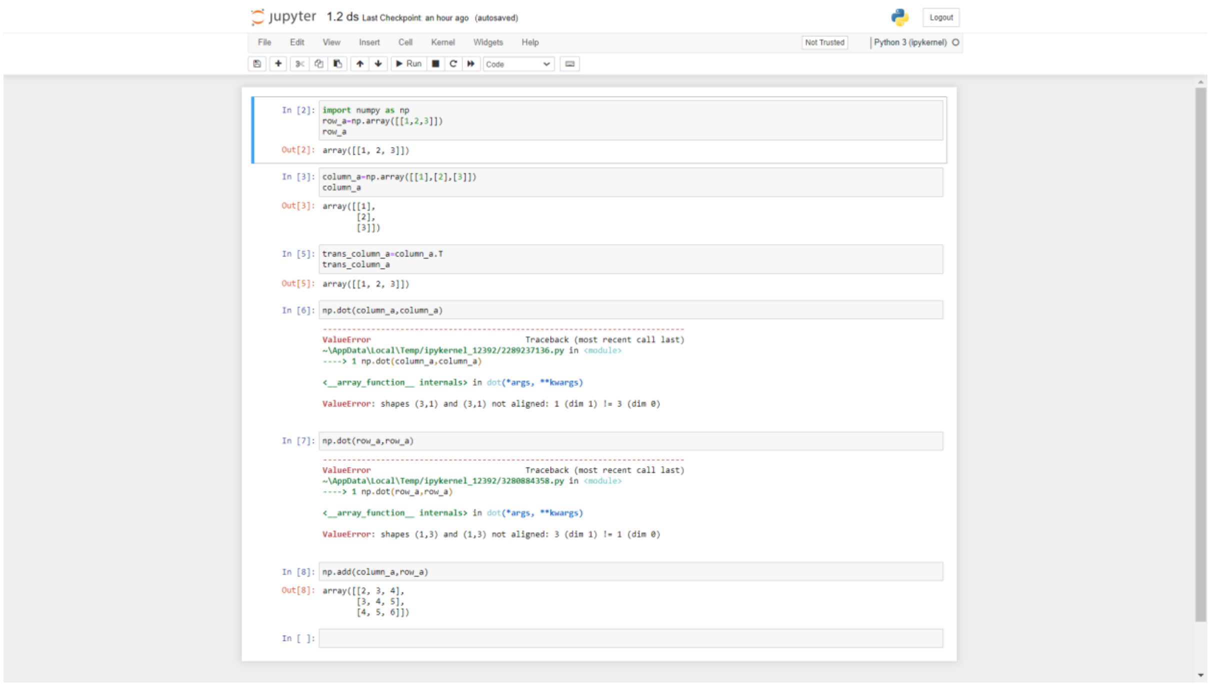

jupyter 1,1 ds Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [1]: import numpy as np
        matrix_a=np.array([[1,2,3,4],[5,6,7,8],[8,5,3,9]])
        matrix_b=np.array([[5,9,4,0],[2,6,9,3],[8,0,2,1]])

In [2]: matrix_a.ndim

Out[2]: 2

In [3]: matrix_b.ndim
Out[3]: 2

In [4]: matrix_a.shape

Out[4]: (3, 4)

In [5]: matrix_b.shape
Out[5]: (3, 4)

In [6]: np.add(matrix_a,matrix_b)

Out[6]: array([[ 6, 11,  7,  4],
               [ 7, 12, 16, 11],
               [16,  5,  5, 10]])

In [7]: np.subtract(matrix_a,matrix_b)
Out[7]: array([[-4, -7, -1,  4],
               [ 3,  0, -2,  5],
               [ 0,  5,  1,  8]])

In [8]: matrix_a*matrix_b

Out[8]: array([[ 5, 18, 12,  0],
               [10, 36, 63, 24],
               [64,  0,  6,  9]])

In [9]: trans_a=matrix_a.T
        trans_a
Out[9]: array([[1, 5, 8],
               [2, 6, 5],
               [3, 7, 3],
               [4, 8, 9]])
```



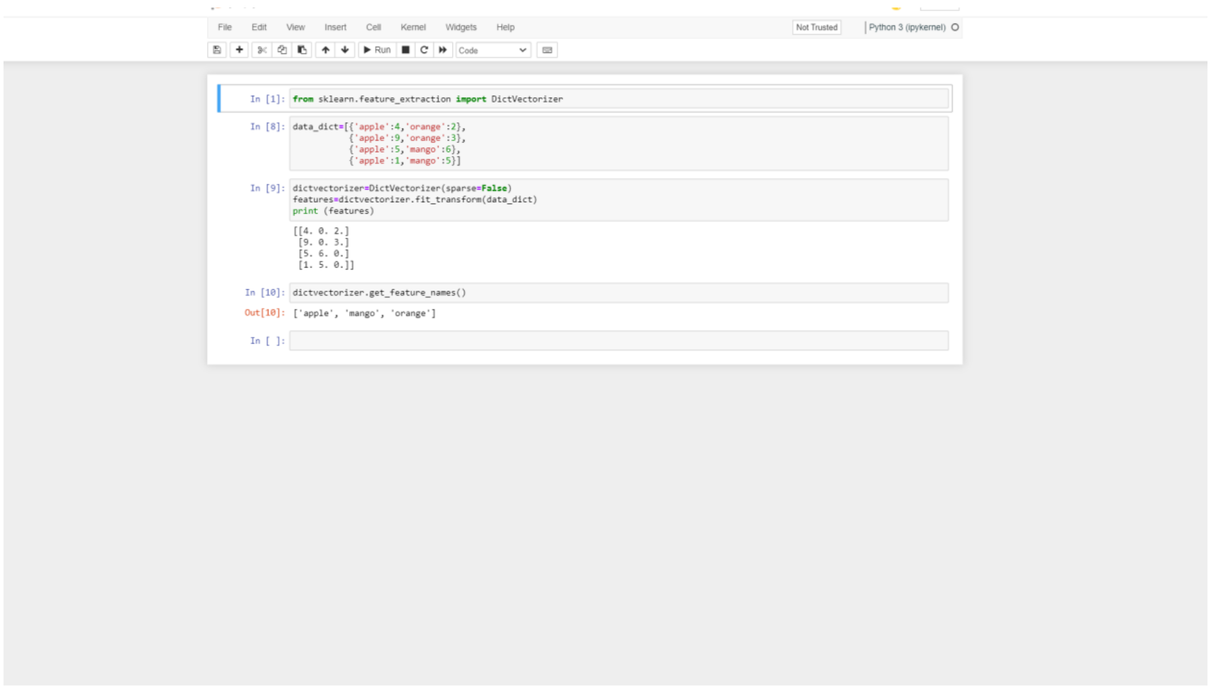
```
        trans_a

Out[9]: array([[1, 5, 8],
               [2, 6, 5],
               [3, 7, 3],
               [4, 8, 9]])

In [10]: np.matmul(matrix_a,trans_a)

Out[10]: array([[ 30,  70,  63],
                [ 70, 174, 163],
                [ 63, 163, 179]])

In [11]: np.average(matrix_b)
         np.mean(matrix_b)
         np.std(matrix_b)

Out[11]: 3.1743328257901515

In [12]: np.mean(matrix_b)

Out[12]: 4.083333333333333

In [13]: np.std(matrix_b)

Out[13]: 3.1743328257901515

In [14]: np.max(matrix_a,axis=0)

Out[14]: array([8, 6, 7, 9])

In [15]: np.max(matrix_a,axis=1)

Out[15]: array([4, 8, 9])

In [16]: np.min(matrix_b)

Out[16]: 0

In [17]: a=matrix_b.reshape(2,6)
         a

Out[17]: array([[5, 9, 4, 0, 2, 6],
                [9, 3, 8, 0, 2, 1]])
```

1.2) Create a row vector row_a, and a column vector col_a. Create the transpose of col_a, calculate the dot product of col_a with itself. Add the vectors row_a and col_a (broadcasting).



```
In [2]: import numpy as np
        row_a=np.array([[1,2,3]])
        row_a

Out[2]: array([[1, 2, 3]])

In [3]: column_a=np.array([[1],[2],[3]])
        column_a

Out[3]: array([[1],
               [2],
               [3]])

In [5]: trans_column_a=column_a.T
        trans_column_a

Out[5]: array([[1, 2, 3]])

In [6]: np.dot(column_a,column_a)

        ---------------------------------------------------------------------------
        ValueError                                Traceback (most recent call last)
        ~\AppData\Local\Temp/ipykernel_12392/2289237136.py in <module>
        ----> 1 np.dot(column_a,column_a)

        <__array_function__ internals> in dot(*args, **kwargs)

        ValueError: shapes (3,1) and (3,1) not aligned: 1 (dim 1) != 3 (dim 0)

In [7]: np.dot(row_a,row_a)

        ---------------------------------------------------------------------------
        ValueError                                Traceback (most recent call last)
        ~\AppData\Local\Temp/ipykernel_12392/3280884358.py in <module>
        ----> 1 np.dot(row_a,row_a)

        <__array_function__ internals> in dot(*args, **kwargs)

        ValueError: shapes (1,3) and (1,3) not aligned: 3 (dim 1) != 1 (dim 0)

In [8]: np.add(column_a,row_a)

Out[8]: array([[2, 3, 4],
               [3, 4, 5],
               [4, 5, 6]])

In [ ]:
```

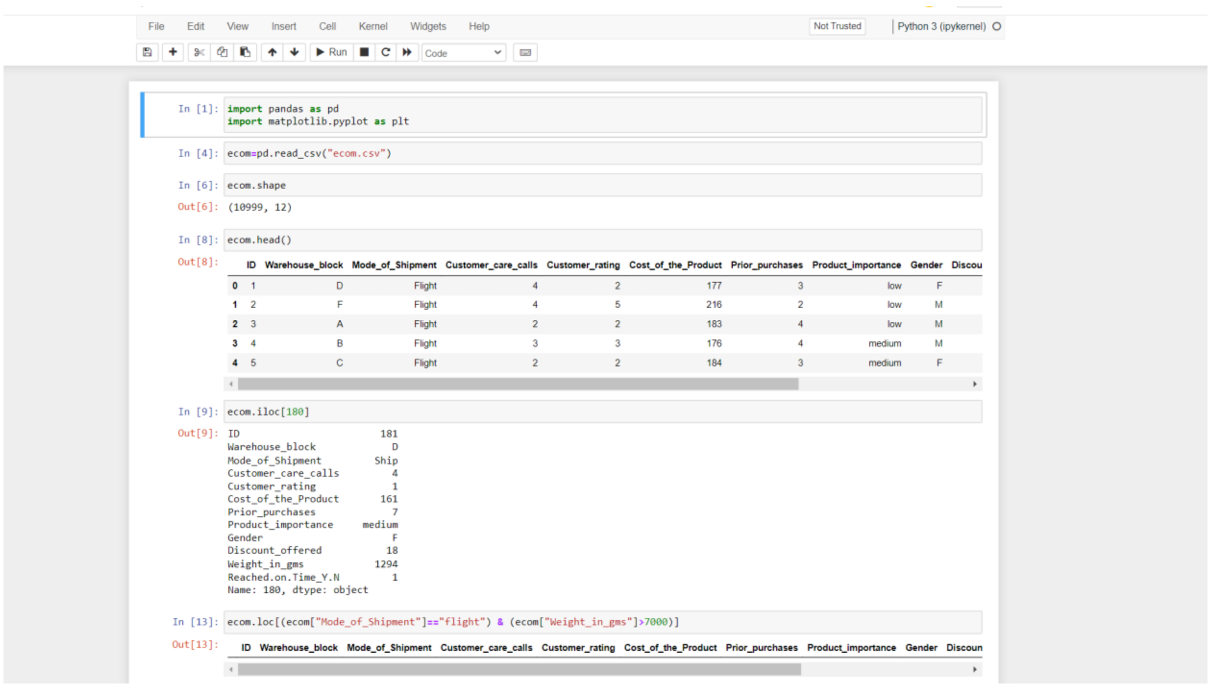
1.3) Create a dictionary of data. Convert the dictionary to a feature matrix. Display the feature matrix and its column names.



```
In [1]: from sklearn.feature_extraction import DictVectorizer

In [8]: data_dict=[{'apple':4,'orange':2},
                   {'apple':9,'orange':3},
                   {'apple':5,'mango':6},
                   {'apple':1,'mango':5}]

In [9]: dictvectorizer=DictVectorizer(sparse=False)
        features=dictvectorizer.fit_transform(data_dict)
        print (features)

        [[4. 0. 2.]
         [9. 0. 3.]
         [5. 6. 0.]
         [1. 5. 0.]]

In [10]: dictvectorizer.get_feature_names()

Out[10]: ['apple', 'mango', 'orange']

In [ ]:
```

2. Data handling

You are given the dataset ecom.csv. It contains various properties of E-commerce transactions. Create a pandas dataframe for the dataset. Use appropriate functions to show the shape (number of feature vectors x number of features) of the dataset. Use appropriate slicing functions to show the head and tail ends of the dataset, to display the feature vector corresponding to the row number 180 and to display the set of tuples where mode of shipment is flight and weight is more than 7000 gms. Find the mean, median, mode and variance of the customer rating. Generate descriptive statistics of the numeric features in the dataset.

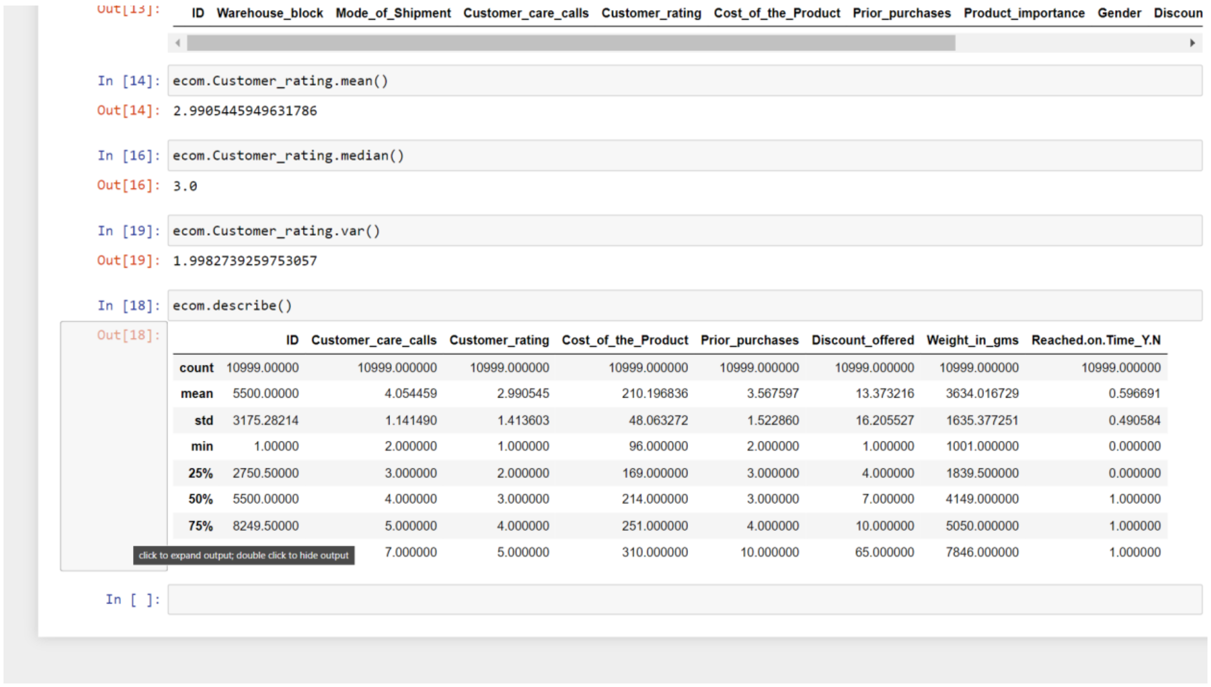


```
In [1]: import pandas as pd
        import matplotlib.pyplot as plt

In [4]: ecom=pd.read_csv("ecom.csv")

In [6]: ecom.shape
Out[6]: (10999, 12)

In [8]: ecom.head()
Out[8]:
```

	ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Product_importance	Gender	Discou
0	1	D	Flight	4	2	177	3	low	F	
1	2	F	Flight	4	5	216	2	low	M	
2	3	A	Flight	2	2	183	4	low	M	
3	4	B	Flight	3	3	176	4	medium	M	
4	5	C	Flight	2	2	184	3	medium	F	

```
In [9]: ecom.iloc[180]
Out[9]: ID                         181
        Warehouse_block              D
        Mode_of_Shipment          Ship
        Customer_care_calls          4
        Customer_rating              1
        Cost_of_the_Product        161
        Prior_purchases              7
        Product_importance      medium
        Gender                       F
        Discount_offered            18
        Weight_in_gms             1294
        Reached.on.Time_Y.N          1
        Name: 180, dtype: object

In [13]: ecom.loc[(ecom["Mode_of_Shipment"]=="flight") & (ecom["Weight_in_gms"]>7000)]
Out[13]:
```

ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Product_importance	Gender	Discoun



```
Out[13]:
```

ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Product_importance	Gender	Discoun

```
In [14]: ecom.Customer_rating.mean()
Out[14]: 2.9905445949631786

In [16]: ecom.Customer_rating.median()
Out[16]: 3.0

In [19]: ecom.Customer_rating.var()
Out[19]: 1.9982739259753057

In [18]: ecom.describe()
Out[18]:
```

	ID	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Discount_offered	Weight_in_gms	Reached.on.Time_Y.N
count	10999.00000	10999.000000	10999.000000	10999.000000	10999.000000	10999.000000	10999.000000	10999.000000
mean	5500.00000	4.054459	2.990545	210.196836	3.567597	13.373216	3634.016729	0.596691
std	3175.28214	1.141490	1.413603	48.063272	1.522860	16.205527	1635.377251	0.490584
min	1.00000	2.000000	1.000000	96.000000	2.000000	1.000000	1001.000000	0.000000
25%	2750.50000	3.000000	2.000000	169.000000	3.000000	4.000000	1839.500000	0.000000
50%	5500.00000	4.000000	3.000000	214.000000	3.000000	7.000000	4149.000000	1.000000
75%	8249.50000	5.000000	4.000000	251.000000	4.000000	10.000000	5050.000000	1.000000
click to expand output; double click to hide output		7.000000	5.000000	310.000000	10.000000	65.000000	7846.000000	1.000000

```
In [ ]:
```

3. Data Visualization

Create a dataframe from the python dictionary consisting of three attributes and values: age:[12, 14, 3,8, 7, 5,12, 18,21,19], height:[140, 150, 110, 130, 135, 120, 150, 170, 178, 180], and weight:[40, 50, 10, 30, 35, 20, 50, 70, 78, 80].

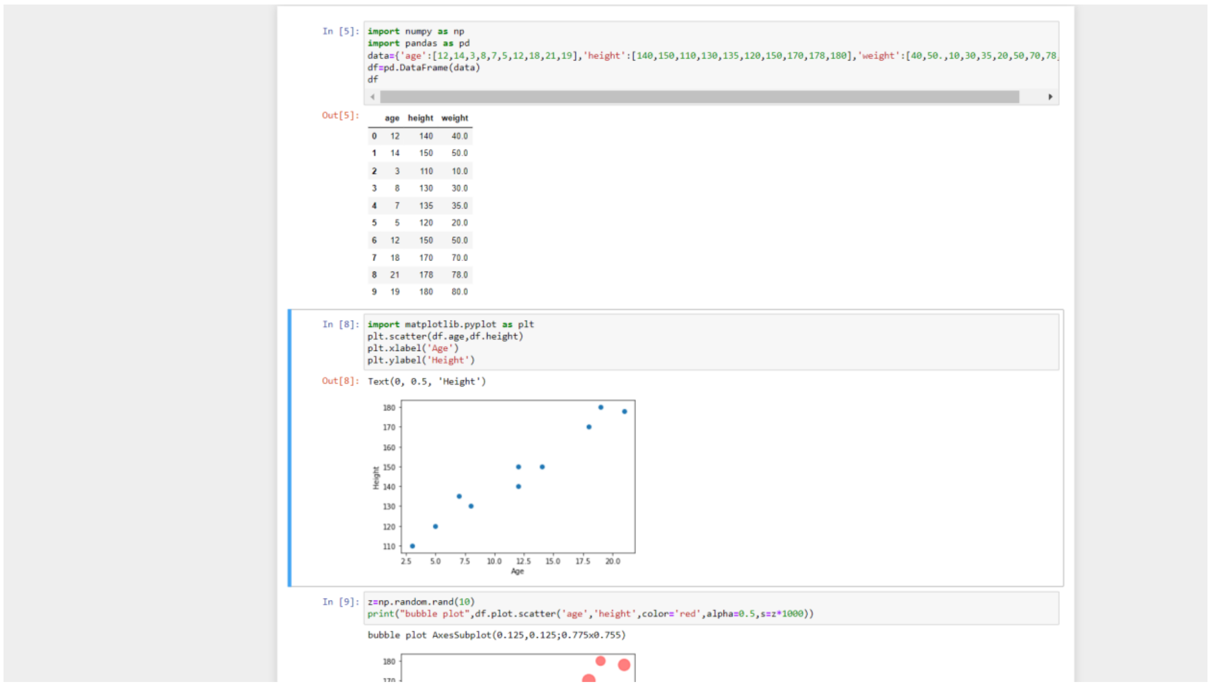
Draw a scatter plot with age and height on the x and y axis respectively.

Draw a bubble plot with age and height on the x and y axis respectively.

Draw a density plot for the attribute weight

Draw a histogram for the attribute age.

Draw a boxplot for the attribute height.



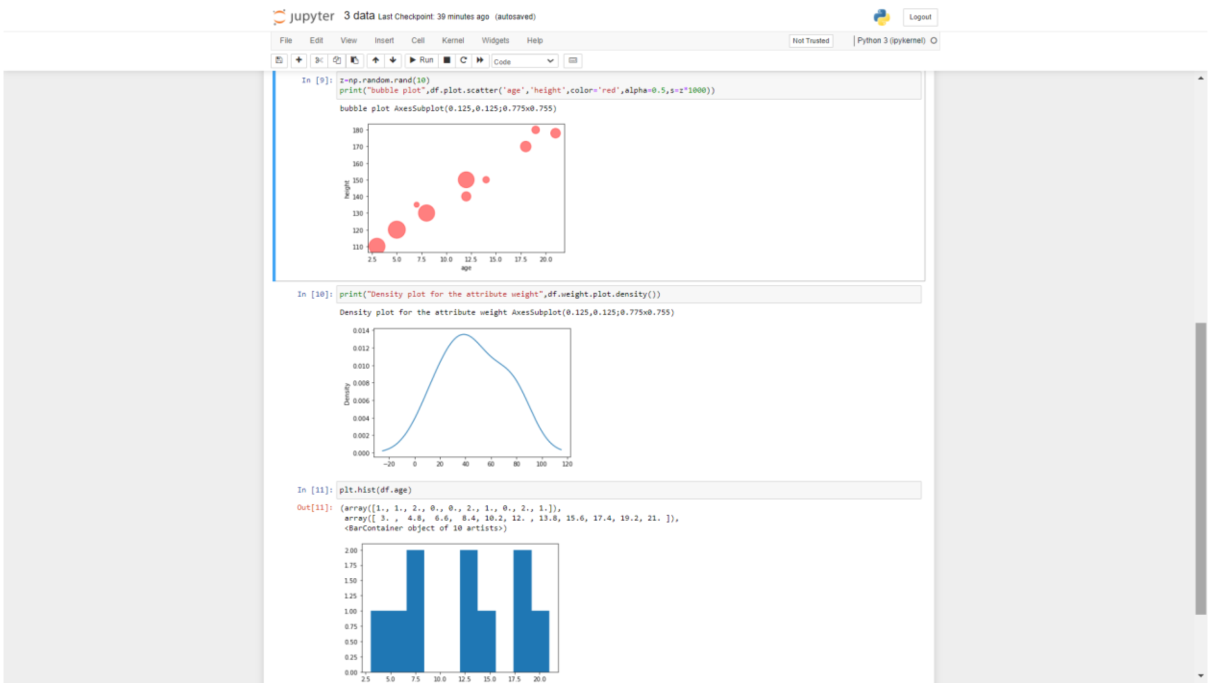
```
In [5]: import numpy as np
        import pandas as pd
        data={'age':[12,14,3,8,7,5,12,18,21,19],'height':[140,150,110,130,135,120,150,170,178,180],'weight':[40,50.,10,30,35,20,50,70,78,
        df=pd.DataFrame(data)
        df
```

Out[5]:

	age	height	weight
0	12	140	40.0
1	14	150	50.0
2	3	110	10.0
3	8	130	30.0
4	7	135	35.0
5	5	120	20.0
6	12	150	50.0
7	18	170	70.0
8	21	178	78.0
9	19	180	80.0

```
In [8]: import matplotlib.pyplot as plt
        plt.scatter(df.age,df.height)
        plt.xlabel('Age')
        plt.ylabel('Height')

Out[8]: Text(0, 0.5, 'Height')
```

```
In [9]: z=np.random.rand(10)
        print("bubble plot",df.plot.scatter('age','height',color='red',alpha=0.5,s=z*1000))

        bubble plot AxesSubplot(0.125,0.125;0.775x0.755)
```



```
In [9]: z=np.random.rand(10)
        print("bubble plot",df.plot.scatter('age','height',color='red',alpha=0.5,s=z*1000))

        bubble plot AxesSubplot(0.125,0.125;0.775x0.755)
```

```
In [10]: print("Density plot for the attribute weight",df.weight.plot.density())

         Density plot for the attribute weight AxesSubplot(0.125,0.125;0.775x0.755)
```

```
In [11]: plt.hist(df.age)

Out[11]: (array([1., 1., 2., 0., 0., 2., 1., 0., 2., 1.]),
          array([ 3. ,  4.8,  6.6,  8.4, 10.2, 12. , 13.8, 15.6, 17.4, 19.2, 21. ]),
          <BarContainer object of 10 artists>)
```