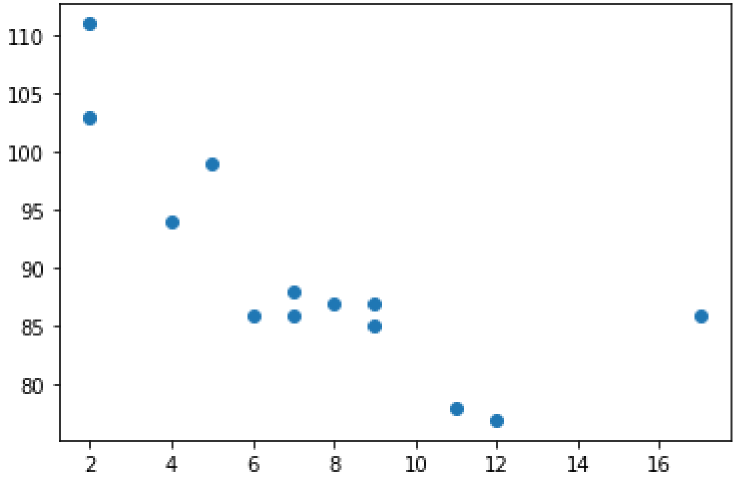
SESSION 8

1. Scatter Plots in matplotlib

- The `scatter()` function plots one dot for each observation.
- It needs two arrays of the same length, one for the values of the x-axis, and one for values on the y-axis.

In [ ]:

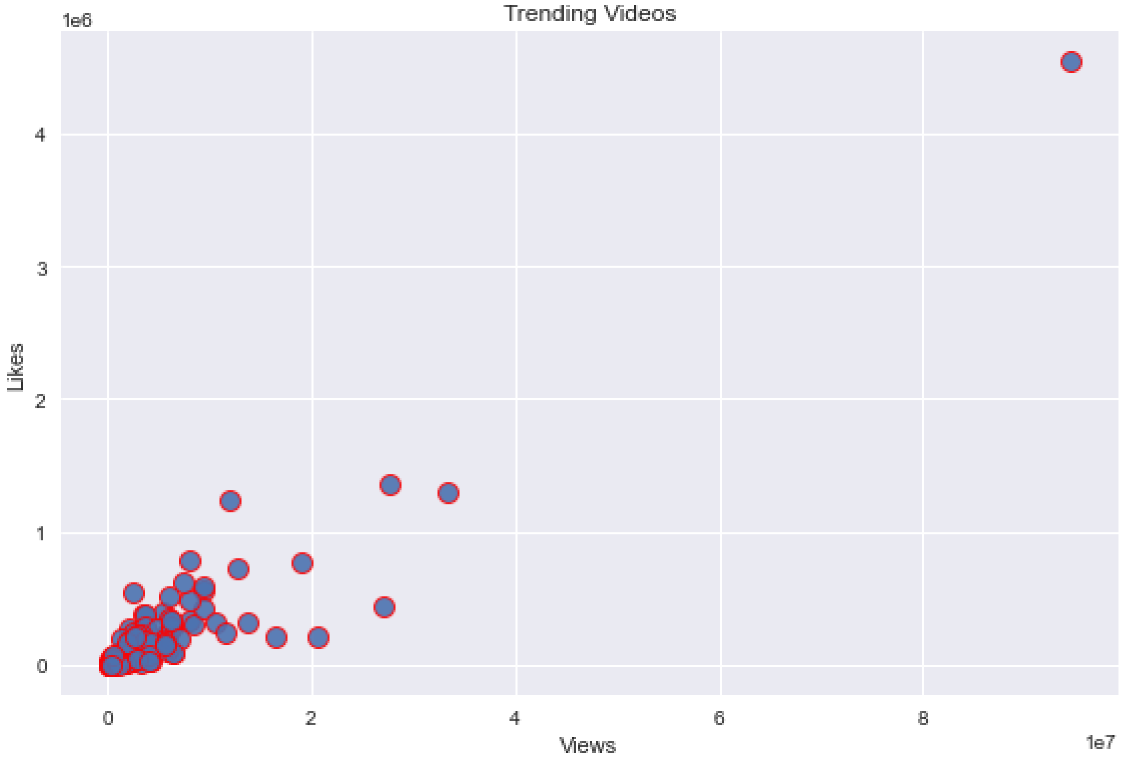
```
# Scatter Function
import matplotlib.pyplot as plt
import numpy as np

x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])

plt.scatter(x, y)
plt.show()
```



2. Scatter Plots using CSV in matplotlib

In [ ]:

```
#scatter_plotting using CSV Input

import pandas as pd
import matplotlib.pyplot as plt

plt.style.use('seaborn')  # to get seaborn scatter plot

# read the csv file to extract data

data = pd.read_csv('Images\data.csv')
view_count = data['view_count']
likes = data['likes']
ratio = data['ratio']

plt.scatter(view_count, likes, s=100, alpha=0.9, edgecolor='red', linewidth=1)

plt.title('Trending Videos')
plt.xlabel('Views')
plt.ylabel('Likes')

plt.tight_layout()
plt.show()
```



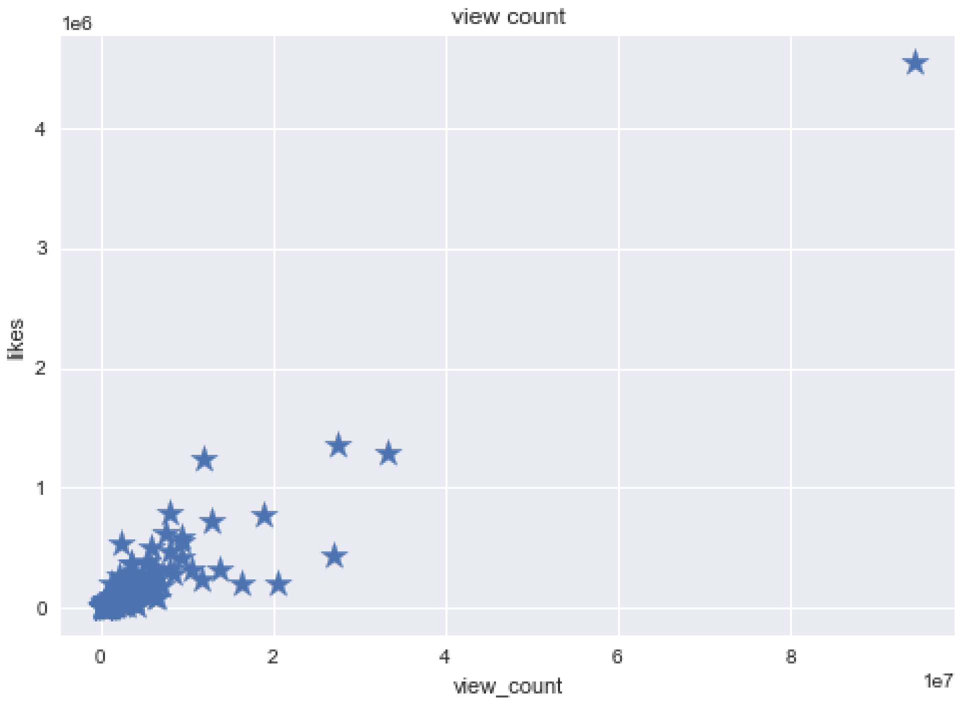
3. Change marker and marker size using CSV

- The scatter plots above have round markers. You can alter the shape of the marker with the marker parameter and size of the marker with the s parameter of the `scatter()` function.
- For instance, to make the markers start-shaped instead of the round with larger size

In [ ]:

```
# Change Marker to Star (*)
# read the csv file to extract data
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv('Images\data.csv')
view_count = data['view_count']
likes = data['likes']

# plot a scatter plot with star markers
```

```
plt.scatter(view_count, likes, marker='*', s=200)
# set axis lables
plt.xlabel("view_count")
plt.ylabel("likes")
# set chart title
plt.title("view count")
plt.show()
```



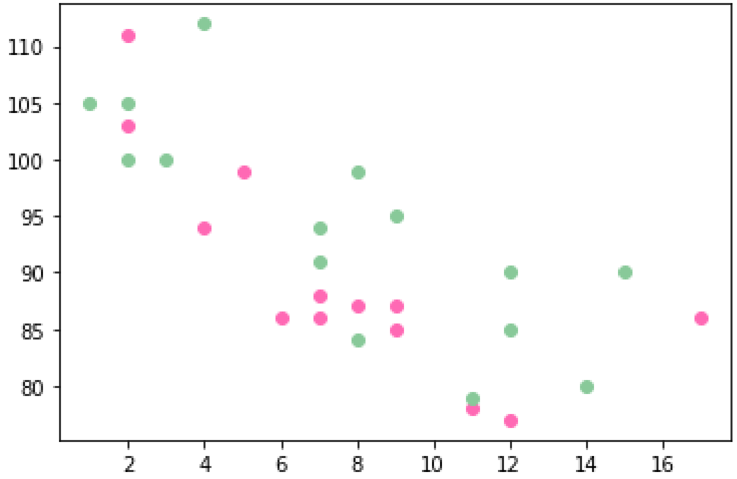
4. Scatter plot colored by category

- You can also have different colors for different data points in matplotlib’s scatter plot.
- This is very useful if your data points belonging to different categories.
- For instance, in the above example, if we add data corresponding to the nationalities of the students say country A and B and want to display each country with a different color

In [ ]:

```
import matplotlib.pyplot as plt
import numpy as np
```

```
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])
plt.scatter(x, y, color = 'hotpink')

x = np.array([2,2,8,1,15,8,12,9,7,3,11,4,7,14,12])
y = np.array([100,105,84,105,90,99,90,95,94,100,79,112,91,80,85])
plt.scatter(x, y, color = '#88c999')

plt.show()
```



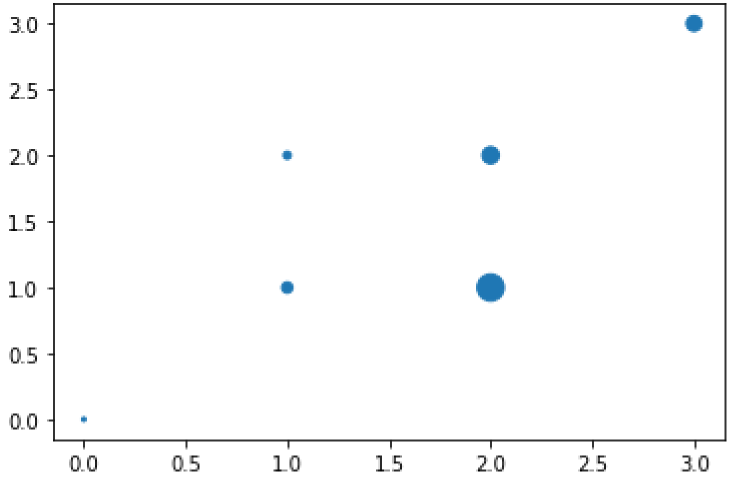
5. Markers’ Size in Scatter Plot

- *To set specific size for markers in Scatter Plot in Matplotlib, pass required sizes for markers as list, to s parameter of scatter() function, where each size is applied to respective data point.*

In [ ]:

```
#example
import matplotlib.pyplot as plt

#data
x = [0, 1, 1, 2, 2, 3]
y = [0, 1, 2, 1, 2, 3]

#markers' size
size = [4, 30, 15, 180, 72, 60]
```

```
#scatter plot
plt.scatter(x, y, s = size)

plt.show()
```



6. DataFrame in Heatmap using matplotlib

A heatmap is a matrix kind of 2-dimensional figure which gives a visualisation of numerical data in the form of cells. Each cell of the heatmap is coloured and the shades of colour represent some kind of relationship of the value with the dataframe.

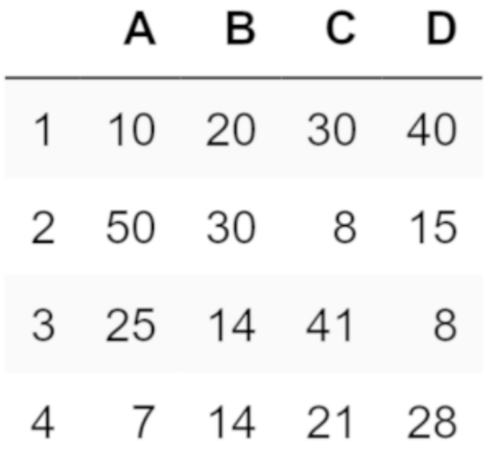
Seaborn library:

Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures. Seaborn helps you explore and understand your data.

6.1 Creating Heatmap using matplotlib library:

- In this matplotlib method, the Panda dataframe will be displayed as a heatmap where the cells of the heatmap will be colour-coded according to the values in the dataframe.
- A colour bar will be present besides the heatmap which acts as a legend for the figure. Below is the implementation.

Consider this dataframe as an example:

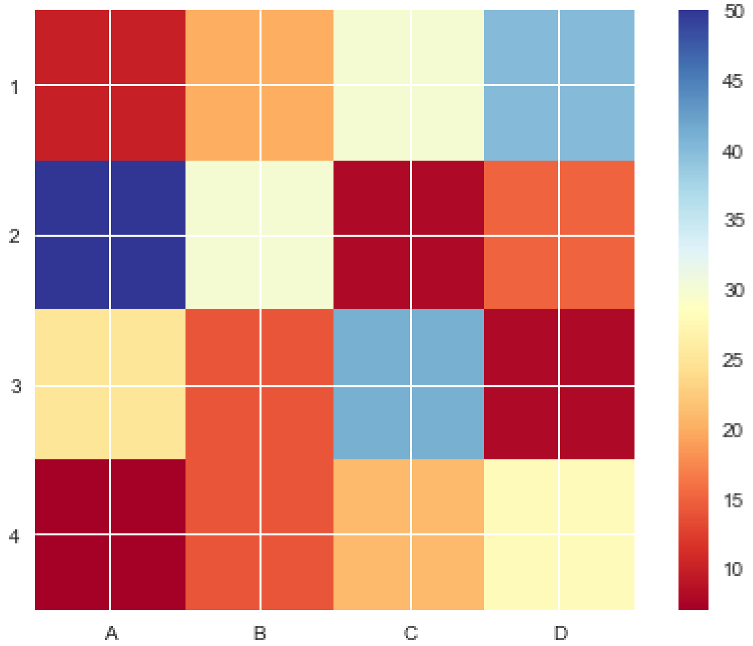


	A	B	C	D
1	10	20	30	40
2	50	30	8	15
3	25	14	41	8
4	7	14	21	28

In [ ]:

```
# Python program to generate a heatmap
# which represents panda dataframe
# in colour coding schemes

# import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Defining index for the dataframe
idx = ['1', '2', '3', '4']

# Defining columns for the dataframe
cols = list('ABCD')

# Entering values in the index and columns
# and converting them into a panda dataframe
df = pd.DataFrame([[10, 20, 30, 40], [50, 30, 8, 15],
                   [25, 14, 41, 8], [7, 14, 21, 28]],
                   columns = cols, index = idx)

# Displaying dataframe as an heatmap
# with diverging colourmap as RdYlBu
plt.imshow(df, cmap ="RdYlBu")

# Displaying a color bar to understand
# which color represents which range of data
plt.colorbar()

# Assigning labels of x-axis
# according to dataframe
plt.xticks(range(len(df)), df.columns)

# Assigning labels of y-axis
# according to dataframe
plt.yticks(range(len(df)), df.index)

# Displaying the figure
plt.show()
```

```
<ipython-input-22-938d3e5699ab>:29: MatplotlibDeprecationWarning: Auto-removal of grids by pcolor() and pcolormesh() is deprecated
since 3.5 and will be removed two minor releases later; please call grid(False) first.
  plt.colorbar()
```



7. Confusion Matrix using matplotlib

- Classification models are a fundamental part of machine learning and are used extensively in various industries.
- The confusion matrix is a key tool in evaluating the performance of classification models. It provides a visual representation of how well the model is predicting true positives, false positives, true negatives, and false negatives.
- Confusion Matrix is a table that is used in classification problems to assess where errors in the model were made.

- The rows represent the actual classes the outcomes should have been. While the columns represent the predictions we have made. Using this table it is easy to see which predictions are wrong.

7.1 Creating a Confusion Matrix:

Confusion matrixes can be created by predictions made from a logistic regression.

For now we will generate actual and predicted values by utilizing NumPy & we need to import metrics from the sklearn module.

> import numpy
> from sklearn import metrics

Next we will need to generate the numbers for "actual" and "predicted" values.

```
actual = numpy.random.binomial(1, 0.9, size = 1000)
predicted = numpy.random.binomial(1, 0.9, size = 1000)
```

Once metrics is imported we can use the confusion matrix function on our actual and predicted values.

```
confusion_matrix = metrics.confusion_matrix(actual, predicted)
```

To create a more interpretable visual display we need to convert the table into a confusion matrix display.

```
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [False, True])
```

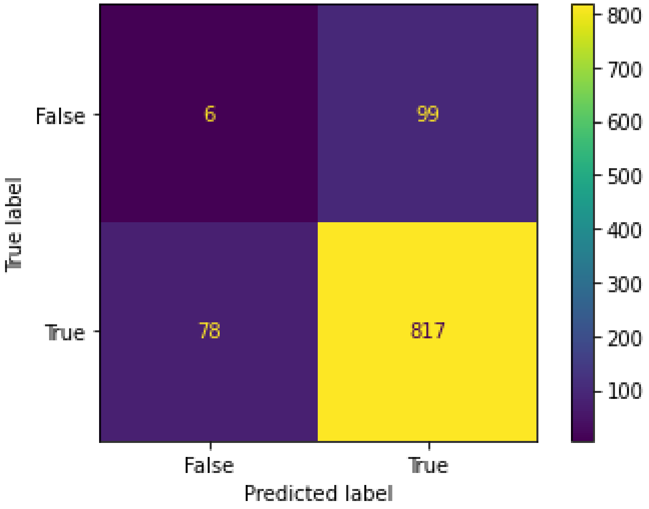
Vizualizing the display requires that we import pyplot from matplotlib.

> import matplotlib.pyplot as plt

Finally to display the plot we can use the functions plot() and show() from pyplot.

```
cm_display.plot()
plt.show()
```

```
In [ ]:
import matplotlib.pyplot as plt
import numpy
from sklearn import metrics

actual = numpy.random.binomial(1,.9,size = 1000)
predicted = numpy.random.binomial(1,.9,size = 1000)

confusion_matrix = metrics.confusion_matrix(actual, predicted)

cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [False, True])

cm_display.plot()
plt.show()
```



Homework Questions

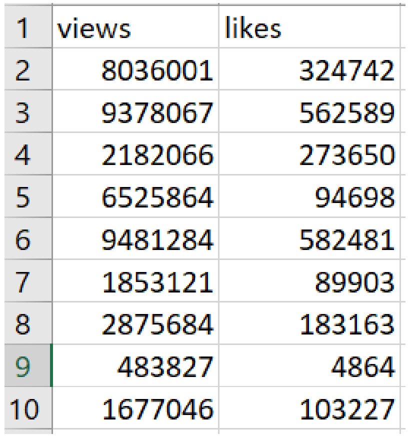
1) Create a scatter plot

```
x = np.array([5,77,8,7,2,17,92,89,14,11,12,9,6])
y = np.array([9,86,87,88,111,86,13,87,94,78,7,85,6])
```

2) Create a scatter plot using marker as STAR (*)

```
height = [178, 175, 120, 186, 103, 187, 178, 169, 183, 180]
weight = [75, 70, 65, 80, 67, 94, 50, 58, 78, 85]
```

3) Create a scatter plot using marker as star , but read the input from a CSV File as below:

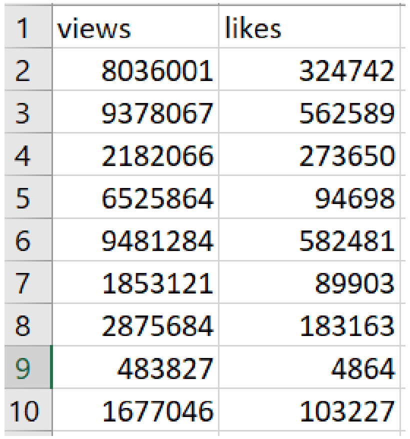


1	views	likes
2	8036001	324742
3	9378067	562589
4	2182066	273650
5	6525864	94698
6	9481284	582481
7	1853121	89903
8	2875684	183163
9	483827	4864
10	1677046	103227

4) Create a scatter plot using marker as STAR and size as 25

```
age = [178, 175, 120, 186, 103, 187, 178, 169, 183, 180]
weight = [75, 70, 65, 80, 67, 94, 50, 58, 78, 85]
```

5) Create a scatter plot using marker as STAR and size as 25, but read the input from a CSV File as below:



1	views	likes
2	8036001	324742
3	9378067	562589
4	2182066	273650
5	6525864	94698
6	9481284	582481
7	1853121	89903
8	2875684	183163
9	483827	4864
10	1677046	103227

6) Create a scatterplot using color (RED & GREEN) by category

7) Create a compare scatter plot for

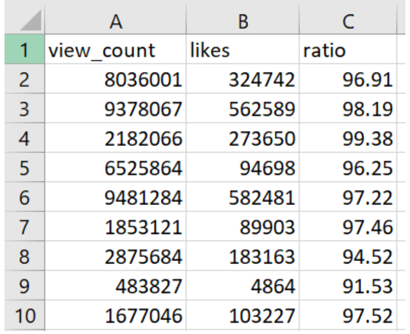
```
#plot1
students_id = np.array([1,2,3,4,5,6,7,8,9,10])
students_marks = np.array([95,98,83,75,67,58,67,78,53,32])
#plot2
students_id = np.array([1,2,3,4,5,6,7,8,9,10])
students_marks = np.array([58,90,67,78,53,32,95,98,83,67,]
```

8) Create a scatter plot and make the marker size as 25

9) Create a scatter plot and make the marker color as RED

```
x = np.array([5,7,8, 2,17])
y1 = np.array([99,103,87,94,78])
y2 = np.array([26, 23, 18, 55, 16])
```

10) Create a Heatmap using matplotlib by reading the below CSV File details



	A	B	C
1	view_count	likes	ratio
2	8036001	324742	96.91
3	9378067	562589	98.19
4	2182066	273650	99.38
5	6525864	94698	96.25
6	9481284	582481	97.22
7	1853121	89903	97.46
8	2875684	183163	94.52
9	483827	4864	91.53
10	1677046	103227	97.52

For solutions of Homework questions, please refer to the `HomeworkSolution.ipynb` file