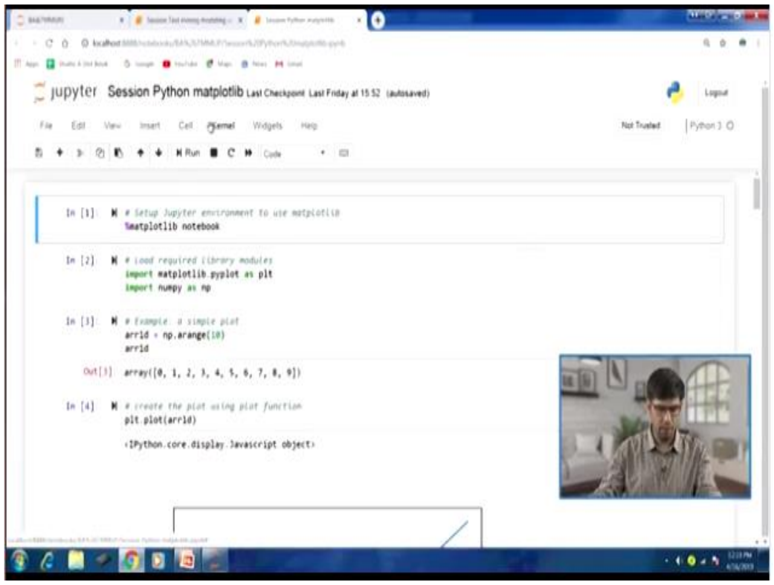
Business Analytics And Text Mining Modeling Using python

Prof. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology Roorkee

Lecture-35
Data Visualization Using Python

Welcome to the course business analytics and text mining modeling using python. So, in last few lectures we have been into the last leg of our you know covering python for analytics. So, that is about the you know last library package that we want to cover matplotlib. So, few things we have been able to discuss, so what we will do, we will recap of what we discuss and then pickup from the last point where we stopped.

(Refer Slide Time: 00:54)



jupyter Session Python matplotlib Last Checkpoint: Last Friday at 15:52 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
In [1]:  # Setup Jupyter environment to use matplotlib
         %matplotlib notebook

In [2]:  # Load required library modules
         import matplotlib.pyplot as plt
         import numpy as np

In [3]:  # Example: a simple plot
         arr1d = np.arange(10)
         arr1d

Out[3]:  array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

In [4]:  # create the plot using plot function
         plt.plot(arr1d)

         <IPython.core.display.Javascript object>
```

So, let us start, so as we discuss that you know we need to setup Jupyter environment to use matplotlib. So, for that we can run this particular you know magic command matplotlib space notebook %matplotlib space notebook. However it might not be required in you know the latest you know Jupyter environments but anyway we will run it.

(Video starts: 01:15)

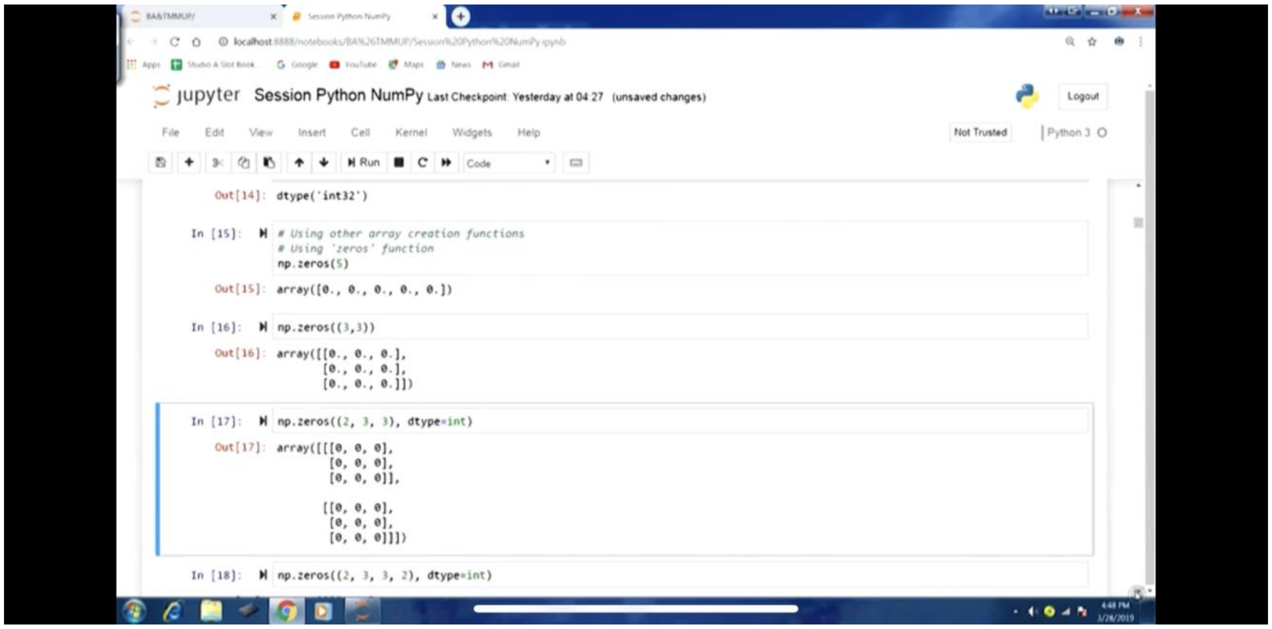
So, this is the first thing then as the next few lines of code that we require that we are going to execute we will require these library modules matplotlib.oyplot as plt and numpy as np. So, let us load these 2 library modules, now first thing we will talk about generating a simple plot. So, before that let us generate some data, so we are going to use a 1 dimensional array object has an example here.

So, let us we know define this array 1d using np.a range function and 10 values. So, if I run this will have though that particular array, now the next thing is a plot using plot function. So, this plot function is coming from the matplotlib library plt.plot and then will pass on this data that we have array 1d. So, typically as you would understand for a 2d plot we need data along x-axis and y-axis.

So, both the coordinates and therefore that is to be used to generate a plots. In case we are passing just 1 you know for passing data for just 1 argument either x or y. So, other one is going to be you know copied and the same data is going to be used for plotting. So, if I run this plt.plot using array 1d so, you can have a look at the graph plot that we have generated.

You can see the same values have been use and therefore we have got a line passing through the origin here and this is how a simple plot can actually be generate as using matplotlib you know functionality. Now let us move forward, so the next thing that we discuss to under this topic is the figure object. So, most of the plots that we are going to generate they are actually generated in the frame define by this figure object.

(Refer Slide Time: 03:04)


jupyter Session Python NumPy Last Checkpoint: Yesterday at 04:27 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
Out[14]: dtype('int32')

In [15]: # Using other array creation functions
         # Using 'zeros' function
         np.zeros(5)

Out[15]: array([0., 0., 0., 0., 0.])

In [16]: np.zeros((3,3))

Out[16]: array([[0., 0., 0.],
                [0., 0., 0.],
                [0., 0., 0.]])

In [17]: np.zeros((2, 3, 3), dtype=int)

Out[17]: array([[[0, 0, 0],
                 [0, 0, 0],
                 [0, 0, 0]],

                [[0, 0, 0],
                 [0, 0, 0],
                 [0, 0, 0]]])

In [18]: np.zeros((2, 3, 3, 2), dtype=int)
```

So, all these parts are going to be residing in this particular fame that is defined by this figure object. So, now will talk about figure object and how we can work around this object to generate more number of plots . So, first thing creating a figure object for this we can use figure function, so plt.figure this is also coming from matplotlib library we can store this object.

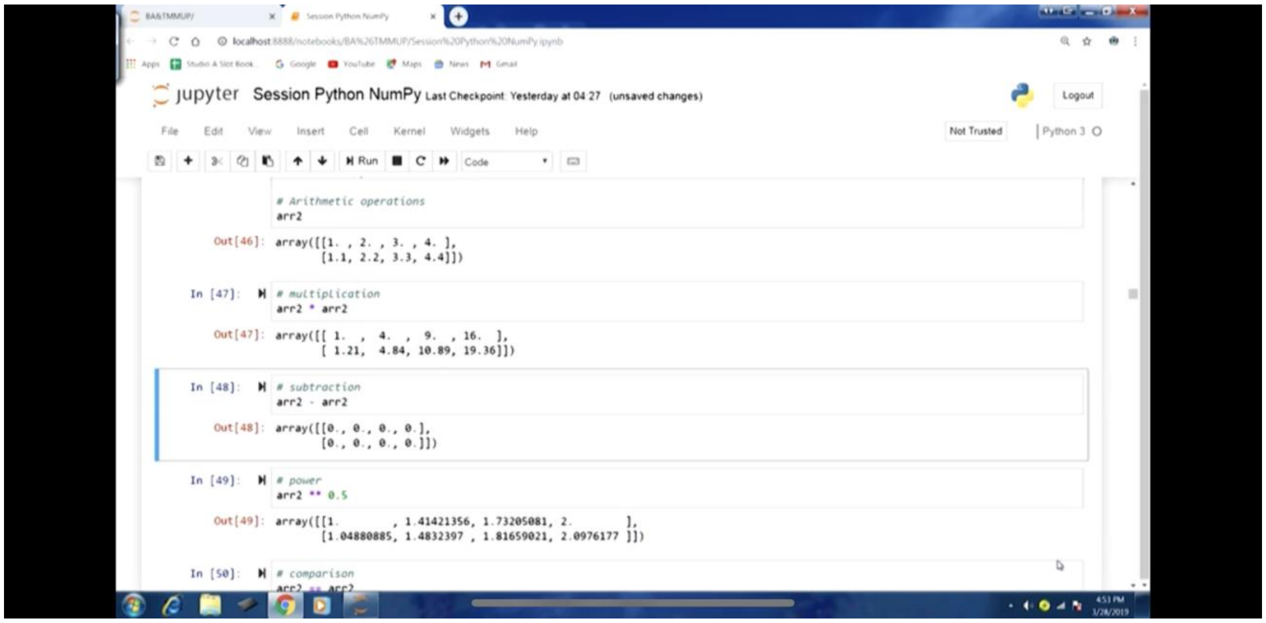
Because we would be using it you know for a few more lines of code, so fig=plt.figure. And if I run this I will get something like this, so you can see this is a blank thing you can see figure2 coming there. So, right now when we just create a figure object we get this blank figure there. Now, if we want to go ahead and generate few plots, we might work upon the number of plots that we might be you know generating in this figure object.

So, if there is just 1 plot that we want to generate, then we can go ahead with that. Otherwise we will have to add access to the figure object, if we you know plan to generate more number of plots. So, how do we do this particular aspect, how do we perform this, execute this. So, for adding more number of plots, we have this add_subplot method. So, this will allow us to add axes to the finger object.

So, in this first and second arguments, in this of method they are used to specify the grid by grid we means number of rows and number of columns that will determine the number of plots you know that we are going to have in this figure object. And the configuration of those parts also, so if we have 2 rows and 2 columns, so we will have a matrix kind of thing you know 4 squares 4 you know subplots that we can create in the in figure object or you know with the same number of plot, you want to go with the same number of plots4.

We can have a different configuration also, we can have 1 row and 4 columns. So, we will have to decide the grid that means the configuration of rows and columns, the way we would like to plot different, you know, we would like to generate different number of plots. So, number of rows and number of columns, the first and second arguments, they can be used to specify the configuration of grid that we would like to use for our plotting.

(Refer Slide Time: 04:30)


jupyter Session Python NumPy Last Checkpoint: Yesterday at 04:27 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
         # Arithmetic operations
         arr2

Out[46]: array([[1. , 2. , 3. , 4. ],
                [1.1, 2.2, 3.3, 4.4]])

In [47]: # multiplication
         arr2 * arr2

Out[47]: array([[ 1.  ,  4.  ,  9.  , 16.  ],
                [ 1.21,  4.84, 10.89, 19.36]])

In [48]: # subtraction
         arr2 - arr2

Out[48]: array([[0., 0., 0., 0.],
                [0., 0., 0., 0.]])

In [49]: # power
         arr2 ** 0.5

Out[49]: array([[1.        , 1.41421356, 1.73205081, 2.        ],
                [1.04880885, 1.4832397 , 1.81659021, 2.0976177 ]])

In [50]: # comparison
         arr2 == arr2
```

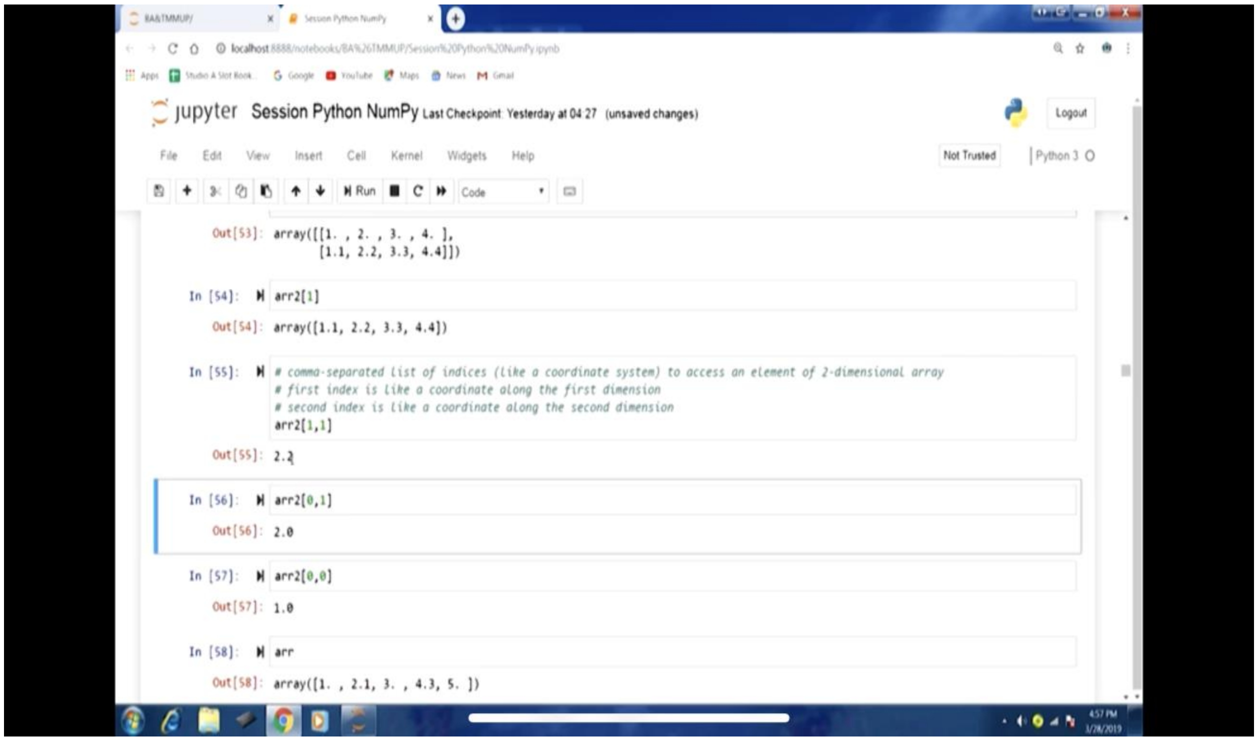
So it will you know, configure the figure object that we have already defined that we have already created. So we will go with this example 2 cross 2 4 plots. And we have another third argument that is to specify the you know index position in the grid. So, when we specify that index position that particular plot is going to be you know selected. So, we can go like index position like 1, 2, 3 and 4.

So, that particular you know plot in that you know figure object in that grid is going to be selected for plotting. So, let me start, so first thing axes first pair of axes that we would like to generate here. So, when we say pair of axes, it will essentially, you know create a mechanism for a subplot for us. So, axes 1 and we are calling you know we are using this fig object that we have already created.

And we were calling add_subplot method, so fig.add_sub plot first argument 2 that means 2 number of rows. Then second argument 2 column number of column then 1 the index position. So, if I run this, so at the this particular index position number 1, the axes are going to be generated. So, if I go back to the figure object you can see this is the very first you know top top left corner of the figure objects.

So, there the index 1 is starts from there and that particular index has been selected to generate these axes essentially you know a subplot. We would be able to you know add a subplot in this using these particular axes. Similarly because we have gone for 2 cross 2 thing, so, therefore, we can add you know axes in more number of you know plots there subplots there.

(Refer Slide Time: 08:07)


jupyter Session Python NumPy Last Checkpoint: Yesterday at 04:27 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
Out[53]: array([[1. , 2. , 3. , 4. ],
                [1.1, 2.2, 3.3, 4.4]])

In [54]: arr2[1]

Out[54]: array([1.1, 2.2, 3.3, 4.4])

In [55]: # comma-separated list of indices (like a coordinate system) to access an element of 2-dimensional array
         # first index is like a coordinate along the first dimension
         # second index is like a coordinate along the second dimension
         arr2[1,1]

Out[55]: 2.2

In [56]: arr2[0,1]

Out[56]: 2.0

In [57]: arr2[0,0]

Out[57]: 1.0

In [58]: arr

Out[58]: array([1. , 2.1, 3. , 4.3, 5. ])
```

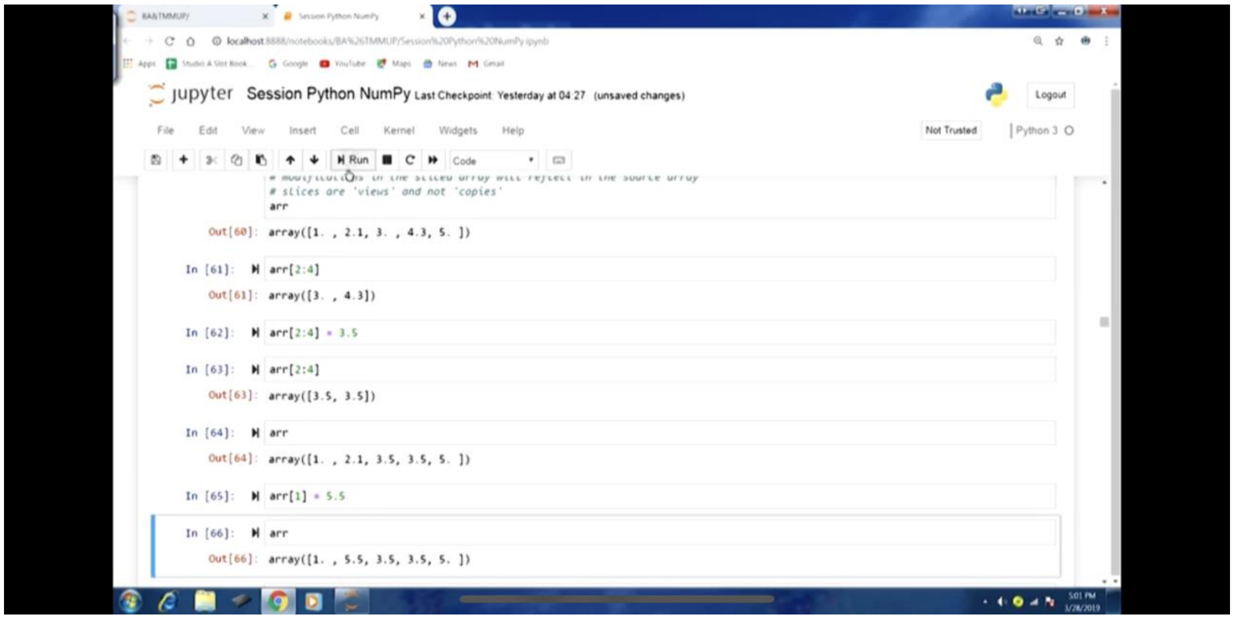
So, for second subplot we can go ahead and get axes again. So, axes2 now figure fig.add_subplot 2, 2, 2. So, in this case we are picking that you know the second index there. So, if I run this and you can go back to the figure and you can see the second index which is the you know, top right corner of the figure object, that frame and the, you know, there the axes have been added.

So, now this is the subplot that can also be used to generate our plots. Similarly access3, so if I run. So, you can see index number 3, which is the bottom left part of the this frame there axes have been added. Now this can also be used to generate plot. Now let us move forward, so what will do will use the subplots to generate certain graphics. So, by default last subplot is used for plotting.

So, last 1 is the index3 that we have you know generated, so that is going to be used. So, as an example we are you know, we are taking this we are going to add this data using np.random.rand and function 50 observations and we will be taking cumulative some of these randomly generated normal. This is these, because we are using random rand and therefore, they are being drawn from the rand and normal distribution.

These 50 values are going to be drawn from there and then will take the cumsum cumulative sum. And then that is going to be the data that our plot you know plt.plot function will use to generate the plot. And in the second argument, we have a specified few things, so this second argument is actually a string abbreviation to indicate color and line is style that is going to be used to generate this plot.

(Refer Slide Time: 12:14)


jupyter Session Python NumPy Last Checkpoint: Yesterday at 04:27 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
# slices are 'views' and not 'copies'
arr
Out[60]: array([1. , 2.1, 3. , 4.3, 5. ])

In [61]: arr[2:4]
Out[61]: array([3. , 4.3])

In [62]: arr[2:4] = 3.5

In [63]: arr[2:4]
Out[63]: array([3.5, 3.5])

In [64]: arr
Out[64]: array([1. , 2.1, 3.5, 3.5, 5. ])

In [65]: arr[1] = 5.5

In [66]: arr
Out[66]: array([1. , 5.5, 3.5, 3.5, 5. ])
```

So, in this case we are going to we using this dashed line. So, dashed line is you know indicated by double dash . So, we can refer help section we want to understand more about this, we can go to help and you can see we have matplotlib reference. So, there you can type you know plot there and we would be able to find out more details about you know, these configuration.

So, if I just type plot there, so it would be you would see that will have more detail here and these details will include the colors and graphics. So, let me find that on the help section you can see there these are formal strings you can see b character you know b is use for blue, g for green, r for red, so in this fashion k is for black. So, similarly, you know other things are also there for different graph marker.

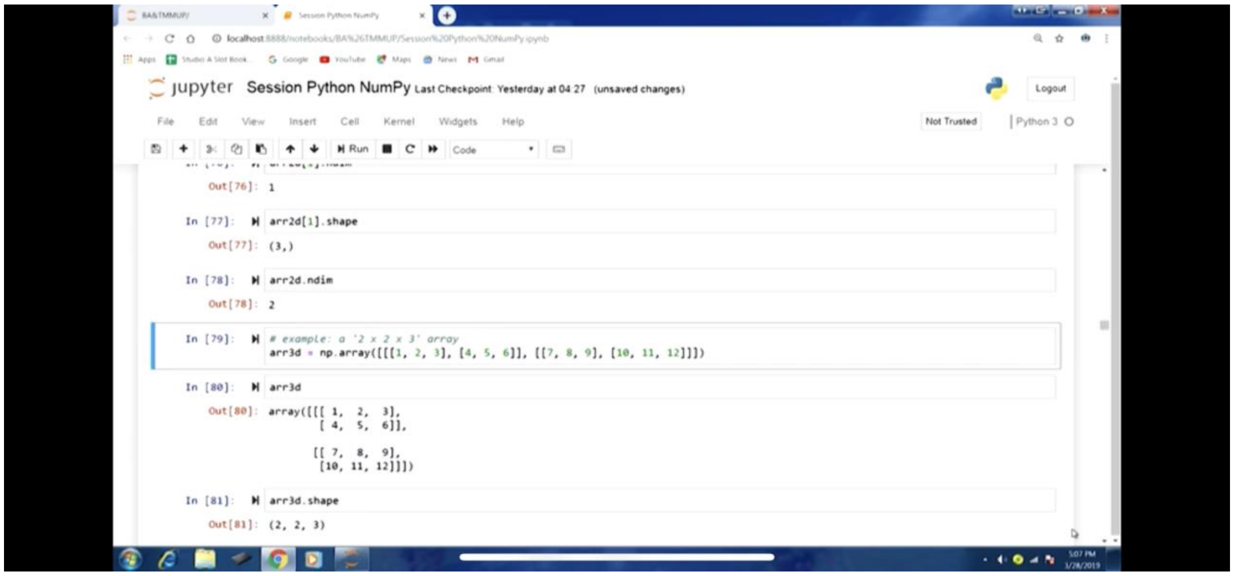
We have this list similarly you know for line style we have this, so you can see 1 dash is solid line style, double dash is dash line style. And then dash.line style and then you know we have column for dotted line style. So these different characters are used to generate different kind of line style there, few examples you can see here k and cap. So, different graph black triangle of markers and this double dash is for you know dash line with the default color.

And in this case in our example that we have taken, you can see we have taken k double dash. So, that is for you know black color and double dash line color plotting. So, in this fashion you can specify you know different parameters to generate your plots. So, data we have in the second argument we have a specified the kind of you know styling features that we want.

So, once we have done that we can go ahead. So, you can refer this is the plot, this is our figure objects, so if I run the next line of code regenerating a plot here. I go back, so you can see here in the third index, you can see a plot has been generated there. So, we had this you know random data you know and cumulative you know, cumsum of these random values and this is the kind of plot that you can see in the figure object the index 3 that has been generated.

We can see this is dashed line and you know black color has been used to generate this part. So, few of the arguments, few of the features that you know we are going to discuss and cover under this topic matplotlib. Now let us take few more examples. So, using you know empty subplots for example histogram using hist method. So, we have these still have few other you know subplots in the fig object that we have created.

(Refer Slide Time: 16:05)


jupyter Session Python NumPy Last Checkpoint: Yesterday at 04:27 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
   Out[76]: 1

In [77]:    arr2d[1].shape

   Out[77]: (3,)

In [78]:    arr2d.ndim

   Out[78]: 2

In [79]:    # example: a '2 x 2 x 3' array
            arr3d = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])

In [80]:    arr3d

   Out[80]: array([[[ 1,  2,  3],
                    [ 4,  5,  6]],

                   [[ 7,  8,  9],
                    [10, 11, 12]]])

In [81]:    arr3d.shape

   Out[81]: (2, 2, 3)
```

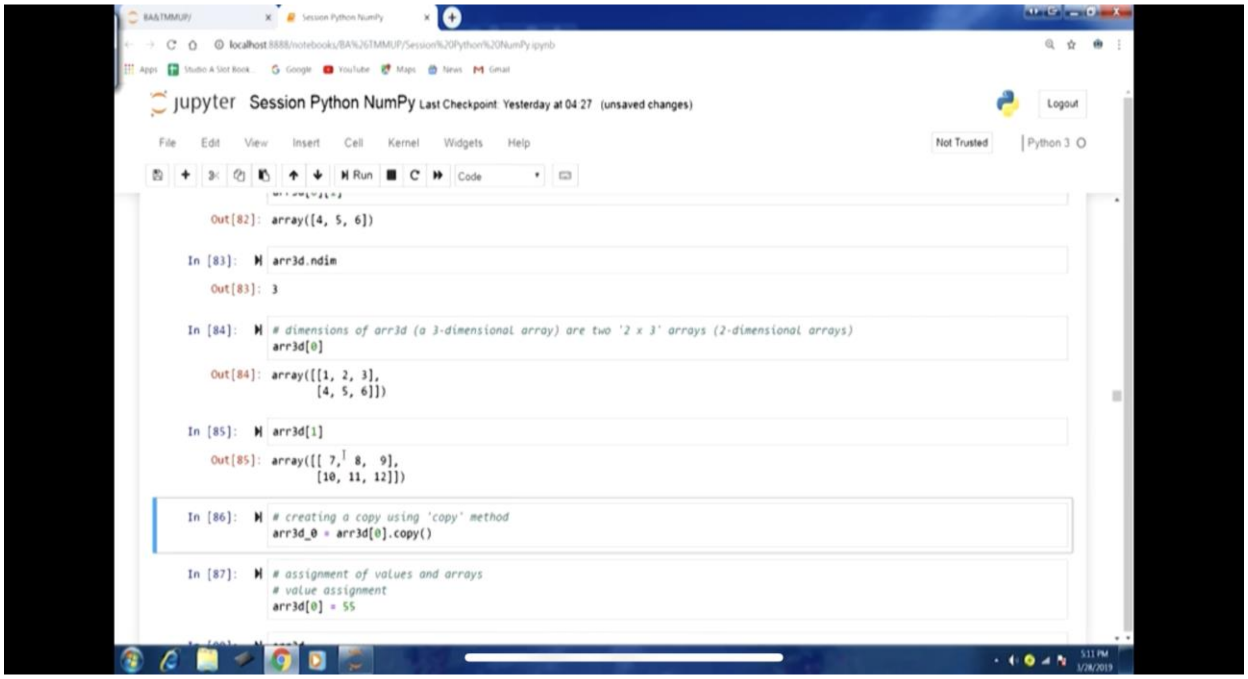
So, this time you know, we are going to call axes1. So in this case, you can see that we are referring to the subplot here. So, axes1 the first the subplot at index1, so we are going to use that and will generate histogram that using hist method here. So, first argument is we are passing the data that we would like to plot So, again 100 values drawn from normal distribution then bins that we want number of bins that we want we have a specified color we have specified and 1 more parameter for we have specified.

So, if I run this then you know if I go back you can see that figure2 you can see that you know, the plot has been generated here. Now, let us move forward. Now next example is scatter plot, so, in this we are using the access to the, you know another subplot that we in that we have there, you know, empty subplot that we still have there at index 2. So, we are going to use a scatter method here will be you know generating a scatter plot.

Again first argument is about data the d values that we want and you know, other details, that of course, you can refer the matplot section to find out more details about any function. So, if I run this, then, in the matplot in the figure object that we have the remaining 1, you can see this is the scatter plot with blue dots, you know and the marker you can see here.

So, this is the scatter plot that has been added, so in this fashion now you can see how the figure object can be used to generate a number of plots you know kind of matrix, different kind of grid, we can configure and different number of plots we can you know, generate there. If we do not want, if you are just going with 1 plot, then we can simply use the plot function and will have that.

(Refer Slide Time: 21:15)


jupyter Session Python NumPy Last Checkpoint: Yesterday at 04:27 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
Out[82]: array([4, 5, 6])

In [83]: arr3d.ndim

Out[83]: 3

In [84]: # dimensions of arr3d (a 3-dimensional array) are two '2 x 3' arrays (2-dimensional arrays)
         arr3d[0]

Out[84]: array([[1, 2, 3],
                [4, 5, 6]])

In [85]: arr3d[1]

Out[85]: array([[ 7,  8,  9],
                [10, 11, 12]])

In [86]: # creating a copy using 'copy' method
         arr3d_0 = arr3d[0].copy()

In [87]: # assignment of values and arrays
         # value assignment
         arr3d[0] = 55
```

So in this fashion, different kind of plotting and can be generated we also touched upon a bit on the styling thing and so that can also be effectively used. Now let us move forward. Now will talk about creating a figure with a you know grid of subplots, and this we are going to use subplot function what this particular this function is going will return an area of subplot objects .

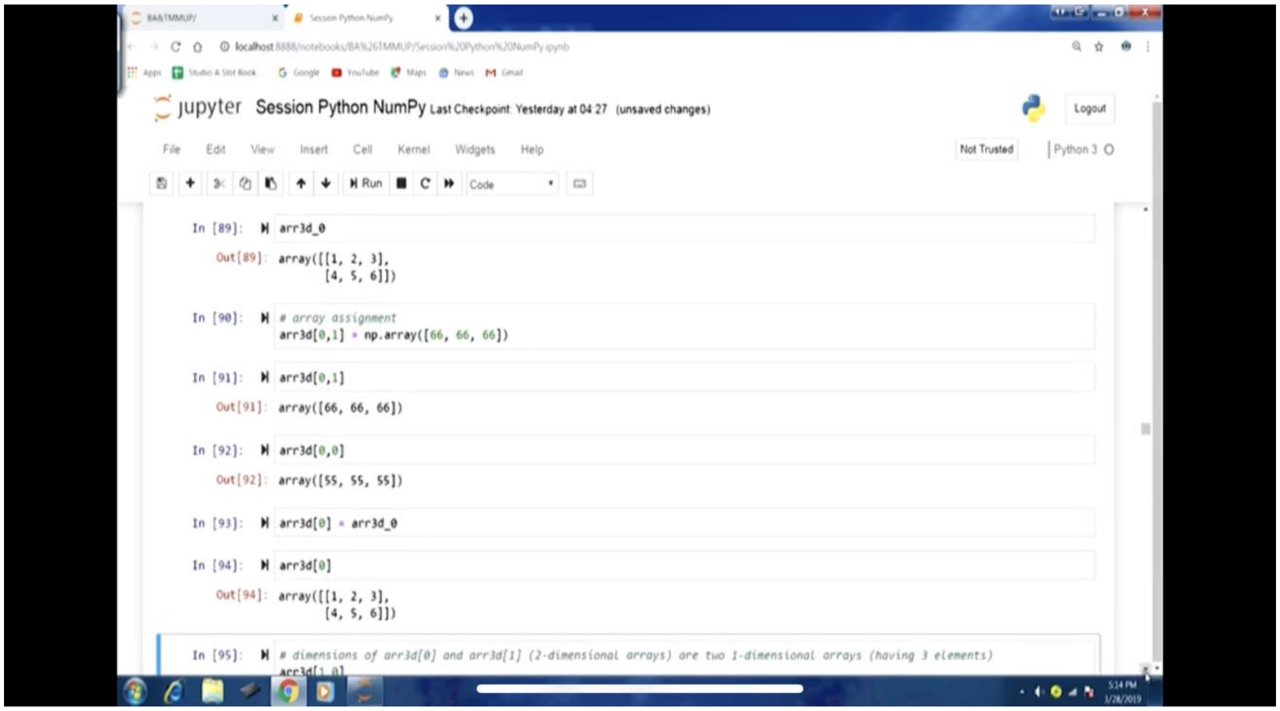
So let me run this here fig 1 axes, plt.subplots and you can see 2, 3. So this is the kind of you know grid that is going to be returned here, so if I run this will have fig 1. So, you can see, so this is another example just to demonstrate the different kind of grid configuration that can be created using the fig object within the you know, fig object fame for our you know generating subplots.

So, you can see we have 2 rows and 3 columns, essentially 6 sub plots, we can generate and in 1 go we have been able to generate this layout and also axes as well. So, this is how we can go ahead and define our configuration get configuration and then plot also. So, here once we have this kind of you know configuration we can go ahead with the you know, the axes you know axes object that we have and we can use the coordinates through the indices to select particular sub plot where we would like to generate you know our graphics.

So, for example axes 0, 1, so in this fashion we will be able to you know, we can, so this is like a 2d array this indexing is like a 2d array. So, 0, 1, we will be able to pick up that you know top left corner plot there and we can generate this again this scatter plot here. So, if I run this and we go back, so you can see here that 0th row that means first row and you know, in column B the index 1 that is second column.

So, there this particular scatter plot has been generated. So, in this fashion, we can use these you know these functions, these subplots, axes and will be able to generate plots in different you know subplot in the fig object. Now let us move forward. So now will talk about some of the you know plotting functionality that is available in the pandas package itself.

(Refer Slide Time: 25:07)


jupyter Session Python NumPy Last Checkpoint: Yesterday at 04:27 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

```
In [89]:  arr3d_0

Out[89]:  array([[1, 2, 3],
                 [4, 5, 6]])

In [90]:  # array assignment
          arr3d[0,1] = np.array([66, 66, 66])

In [91]:  arr3d[0,1]

Out[91]:  array([66, 66, 66])

In [92]:  arr3d[0,0]

Out[92]:  array([55, 55, 55])

In [93]:  arr3d[0] = arr3d_0

In [94]:  arr3d[0]

Out[94]:  array([[1, 2, 3],
                 [4, 5, 6]])

In [95]:  # dimensions of arr3d[0] and arr3d[1] (2-dimensional arrays) are two 1-dimensional arrays (having 3 elements)
          arr3d[1,0]
```

So what will do because the we are going to use pandas, we would like to pandas broad functionality which is internally based on some of the based on, you know many of the features of matplotlib itself. So, to avoid any conflict we will clear the output today there and will start rerunning of our code from here again. So first thing will clear outputs will go to cell tab and all output and will clear the previous things previous output that we have generated along with the variables and other things, objects other python objects, other things that are there.

Now, will talk about some of the pandas build-in methods that can be used to create visualizations and graphics and there we can use the data frame and series objects. So, typically as we have said, that we would be dealing with the structured data, so where we will be have the tabular format of data. So, there data fame and series objects if we can use these objects and directly create plots that would be more meaningful in the analytics context.

So, will use some of the you know pandas build-in methods for that, so let us import pandas to pd and then will generate a you know a series 1 object as well here. So, here again we are drawing 10 observation, 10 values from normal distribution cumulative sum and you know, so in this fashion let me run this code. And will have the series 1 objects, so you can see and you can see you can have a look at the indices also.

So, the in dices are the way we have generated there you can see that different indexes you have generated . Now we can go ahead and create a plot, so in pandas built in functionality we have the plot methods, so any series or data frame object we can call this plot methods. So, we can call series1.plot and the data and the values that we have those are going to be plotted here.

So, if I run this, you can see again we have a fig kind of object here that has been created and plot has been generated. So, you can see the functionality is essentially you know built on top of matplotlib library and you can see the indices they have been plot along the x-axis. So, that is why we had that kind of you know, particular indices those values to clearly display you know the kind of plot that we are going to generate.

So, you can see indices are among the x-axis and the values that we have you can have a look at the values and output 16. So, those have been plotted along you know y axis you can have a look at the range, so range is start from -0.23 and upward up to 6. We can have a look at the plot the plot is also you know is starting from a value 0 and then going up to 6 and on x axis as you can see the indices. So, this is how the python building methods that can used to generate plot here.

Now let us take a data fame example, so for that will you know create this define this data frame object df1. So, will call pd.data frame and again will draw will create a 2d to 2 dimensional this is a 2 dimensional arrays will take 10, 4 will draw those values there. And will take cumulative sum there and sum along all the columns there and we have these columns ABCD and indexes also .

So, let me define this data frame, let us have a look, so you can see that we have these 4 columns A B, C, D and we have been able to generate indices also you can have a look at 0, 10, 20. So, these are the index values that we have row index values, and the column names ABCD and the values they have been drawn from normal distribution. So, you can see these values here, so we have 4 columns and 10 rows, so for t values here in total, this is the data frame object.

And you can see also you can have a look at the values also along each you know, along each column, you can see the cumulative sum kind of cumulative kind of values are being generated there. So, that is clearly visible from the numbers itself. So now, just like the series of object, here also, we can introduce this data frame object df1 and call the plot method here. So, we can call df1.plot.

And these 4 columns that we have, they are going to be, so we will have 4 plots, using these 4 columns and the same indices. So, we have 1 index here 1 index that is going to be plotted along x-axis and 4 columns we have which might be representing which could represent any variable. So, those are going to be plotted along the y-axis will have 4 plots there.

So, if I run this, you can see this kind of plotting is typically more useful in the analytics context that in 1 go, we have been able to generate a number of plots in the same fig objects same you know, sub plots you can see 4 columns A, B, C, D with different color coding and they have been plotted in this fashion. So, similarly you know this is the default thing, so a line you know plotting has been done.

We can further use the you know bar we can also generate bar plot as well. So, for that we can call this method plot.bar. So, df1.plot.bar, so if I call this will have this kind of you know bar plot that has been generated. So again for you know different columns A, B, C, D, so bars have been generated for all of these 4 columns for the same indices you can make comparisons also in this fashion .

(Video ends: 24:25)

You can see how easy it becomes to you know generate plots. So, with this we have been able to cover some of the visualization aspects in python environment. So, we used matplotlib library as well as the pandas building methods to generate different plots and you can see the underlying you know, functionality in panda is also built on top of matplotlib. We can see the figure object and many other functionality.

But you can see a more easy to use in case of pandas. So, with this we would like to complete our discussion on most of the python packages. So, in this particular course we have focused a lot on different learning and gaining more expertise and more experience on various python modules, various python library packages, that is because in our previous courses we have used R, now python is slightly you know more general purpose programming language.

So, slightly more time is required you know to learn this, so that it could be use effectively with all it is power in the analytics context. So, with this you know slightly comprehensive coverage of python for you know analytics it could be really useful to write code for some of the techniques, some of the analysis that we have done in our previous courses.

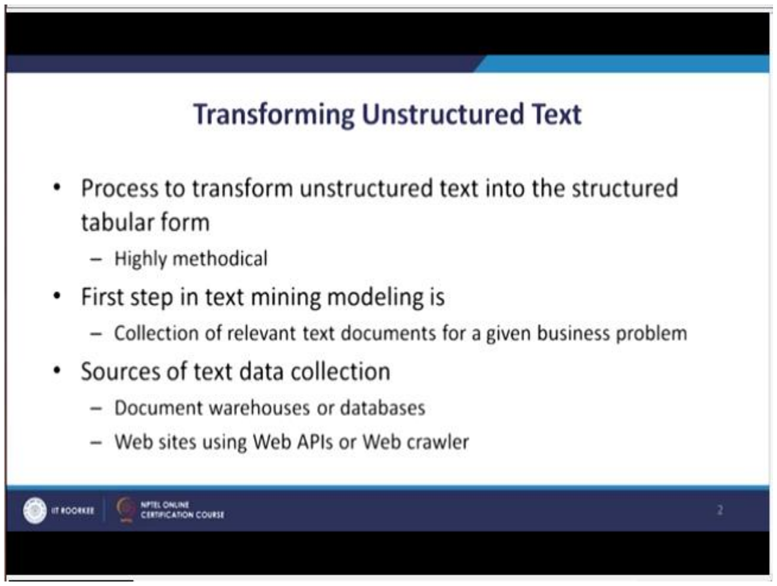
Now what we will do as we discussing some of the initial lectures of this particular course, we have talked about the text mining, modeling and we cover the introductory part there. We compare the text mining with the data mining modeling, so that comparison we perform we talked about different text mining problems that are useful in the analytics that are typically used to solve different you know analytics problem coming from different domains.

Now will take the discussion forward, now be the additional you know knowledge and expertise that we have gained in the python environment that is going to be really useful for us to understand and learn the text mining modeling aspect. So now will start this the next topic that is transforming unstructured text. So, some of the you know codes and functionality and libraries that we have learned about string and text processing in python that is going to be helpful in this .

So, let us start so transforming unstructured text, so as we have talked about that typically in data mining modeling we deal with the structured text and we deal with the numbers which are presented in a structure format. We might be require to process a lot more data to create that structure format but essentially when we have the tabular layout, essentially we are dealing with the numbers there in case of data mining modeling. However in case of you know text mining modeling.

We start with a number of text files, a number of text documents and then we process all those you know text documents text data in a structure layout in a tabular layout. So, that we can further apply some of the techniques that we have learned in data mining modeling in the text mining context as well. So, essentially text is naturally unstructured data and we will have to put in a lot of effort to actually convert into a structured structure tablet form.

(Refer Slide Time: 27:39)



Transforming Unstructured Text

- Process to transform unstructured text into the structured tabular form
  - Highly methodical
- First step in text mining modeling is
  - Collection of relevant text documents for a given business problem
- Sources of text data collection
  - Document warehouses or databases
  - Web sites using Web APIs or Web crawler

So, process for this is highly methodical, so this highly methodical process is required actually, so that we are able to learn both steps which are required to process, which are required to transform the unstructured text into a structure form. We you know some of those things be typically skip in the data mining modeling, however in this case of text mining modeling those steps are really important to learn.

Because you know, it takes a lot of time and expertise to convert some of the unstructured text into a usable form for analytics. So first step typically in text mining modeling is collection of relevant text documents for a given business problem. So, given your business problem you will have to you know collect all the documents that you would be using for a particular text mining modeling problem.

So, first thing is to collect all those documents, so what are some of the common sources of this text data collection so typically document warehouses or databases, organizations, public or private they will have they might have their own is a document warehouses and databases. For example, you know rules regulations, guidelines or technical documents you know they might have their knowledge systems where they are and learning system where there might be so many documents which might be lying around.

So, some of those documents can actually be used for text mining purposes, text mining modeling purposes. So, those could sources for data collection and rather easy sources for text data collection. Then we could have the websites you know, so many websites they provide you know web APIs publicly. So, any one any research analyst can actually use those Web APIs to get data from those popular websites.

For example, twitter they provide a number of you know web APIs to actually extract tweets . So, similarly other websites also then we have a web crawler software programmers also which many people use to crawl through the websites and get the text files that are there. and use that to develop either, you know text you know collection of text documents, which we typically refer as text corpus.

So, the this is these are websites or another sources to collect data, then there could be log files of servers, so many servers are deeply required to run our day to day business operations. So, logs file a number of log files are generated, which keep on recording various operational and task related details in those log files. So, sometimes those log files could be really useful to mine and you know to actually gain certain insights to help in certain witness problems.

So, a log files could also be a sources of text data collection, corporate documents, government documents, as we talked about. So, even if we do not have the data warehouses and databases is still we might be having the documents in our personal computers and mails and other places and hardcopy and many other, you know formats, these corporate and government documents could be available in our organization.

So, they could also be a source of you know text data collection. So, this is about different sources of text data collection. Now will stop here in this lecture, and will continue our discussion on transforming unstructured text in the next lecture, thank you.

Keyword: Text mining modelling, structured and unstructured data, range and pandas function, data warehouse.