

Business Analytics and Text Mining Modeling Using python
Prof. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology Roorkee

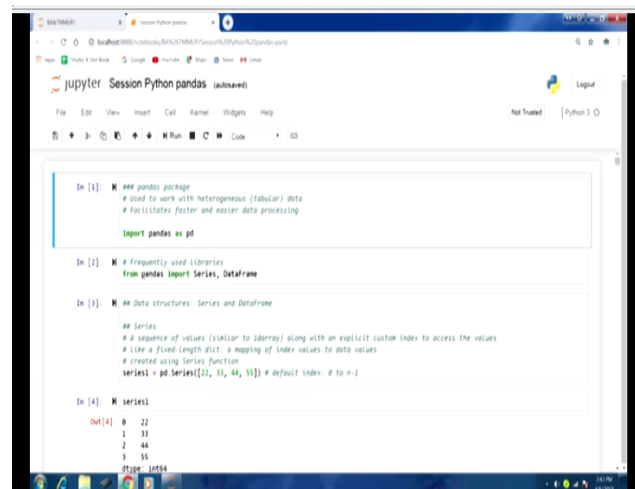
Lecture-26
Database Using python-Pandas-Part I

Welcome to the course business analytics and text mining modeling using python. So, in the previous lecture we started our discussion on the new package that is pandas. So, we will start we will do a slight you know recap of what we discussed you know briefly in the previous lecture and then we will pick up from there. So, as we have discussed you know in the previous lecture as well that NumPy package mainly for generic numerical processing.

However if we want to work with tabular kind of data structure kind of data that we typically do in the analytics area, then pandas package actually provides all those facility. However panda is based on NumPy and you know various other packages also, so that is why we have covered them before. So, that the foundation is there when we discuss the pandas package and many things that we would be learning in this package.

They would be directly you know relevant for the analytics the data processing part, the data transformation part, the summary statistics that we typically generate. So, all those things you know they you know we would be using lot more of you know pandas functionality to directly perform some of these analytics related task.

(Refer Slide Time: 01:49)



```
In [1]: # pandas package
# used to work with heterogeneous (tabular) data
# Facilitates faster and easier data processing

import pandas as pd

In [2]: # Frequently used libraries
from pandas import Series, DataFrame

In [3]: # Data structures: Series and DataFrame

# Series
# A sequence of values (similar to ndarray) along with an explicit custom index to access the values
# like a fixed-length dict: a mapping of index values to data values
# created using Series function
series = pd.Series([22, 33, 44, 55]) # default index: 0 to n-1

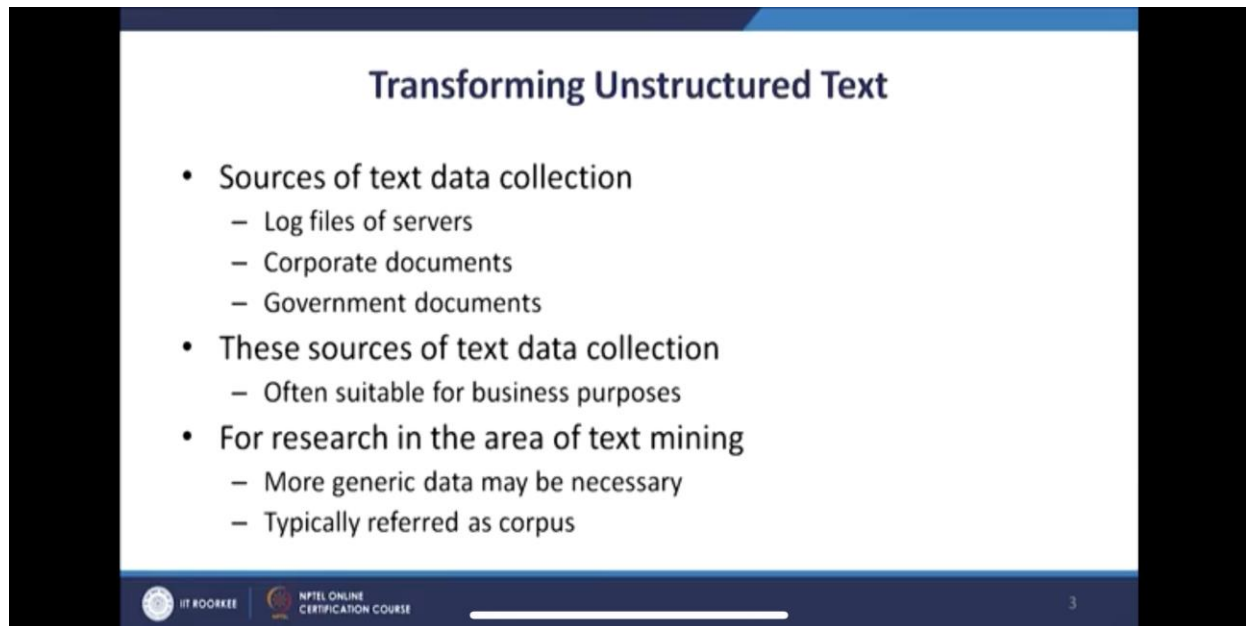
In [4]: # series

Out[4]:
0    22
1    33
2    44
3    55
dtype: object
```

So, let us start, so as we discussed pandas package this is typically used to work with heterogeneous data that is tabular data. That means all the columns they would be representing different variables and therefore they could be different data types as well, one could be you know continuous another could be a categorical, so that kind of differences. So, in terms of python context one would be one could be having floating or integer data point another would be having you know a string or you know or boolean kind of data type.

So, that is going to be one difference, so that is what upon pandas provide us, facilitates faster and easier data processing. So, because many functionality they are for they are happy in develop to work with heterogeneous data. So, therefore directly they are going to be relevant for data processing and these functionalities make this processing easier and faster. So, as we discussed you know this we first typically import the libraries import pandas as pd, so this is first thing that we do, so let me perform this.

(Refer Slide Time: 05:30)



The slide is titled "Transforming Unstructured Text" and contains the following content:

- Sources of text data collection
 - Log files of servers
 - Corporate documents
 - Government documents
- These sources of text data collection
 - Often suitable for business purposes
- For research in the area of text mining
 - More generic data may be necessary
 - Typically referred as corpus

At the bottom of the slide, there are logos for IIT ROORKEE and NPTEL ONLINE CERTIFICATION COURSE, and the number 3 in the bottom right corner.

(Video Starts: 02:58)

So, before I do this like we have been in a previous few lecture we have started doing this aspects also we have started varying the output, so let me clear all the previous output and now let us start. So, first importing of this package, now then certain libraries we would be using quite often, so we would like to import you know these library series and data frame. These are the you know data structure that we would be discussing in this lecture.

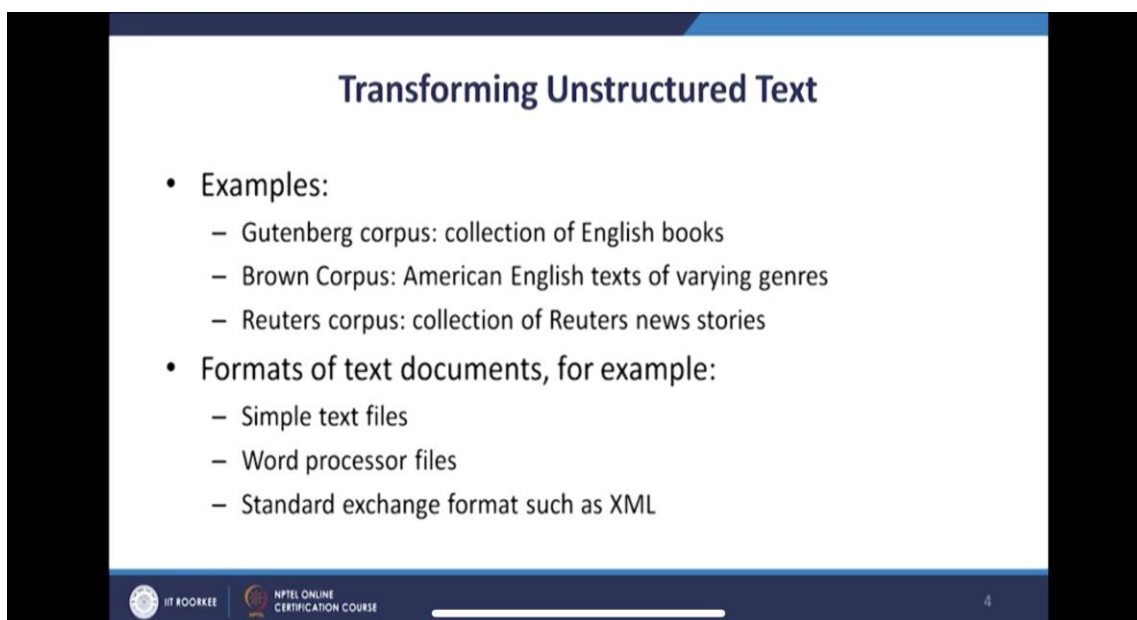
So, let me run this, now the series and data frame these are the 2 structure, so let us start with the series. So, as we discussed series is a sequence of values similar to what we have done in 1d array along with an explicit you know custom index to access the values. So, we will have the index here which would be explicit it would not be. So, whenever we are whenever we create a series object and whenever we are trying to print it we would see an explicit index you know in contrast to what we have in array where we just see the you know sequence of values.

Here in series we see the sequence of values and also an explicit index, now I have also mention custom index because we can always manipulate this index as per our requirement. Because as you would understand that in tabular data, structure data set that we typically use in analytics,

there the indexing we sometimes you would like to change. Because these are nothing but labels, so they could be represented using numeric codes or you know string value.

So, therefore sometimes you would like to you know modify these you know the explicit index that we have in series and data frame. So, you know that is why I have mention that explicit custom index to access the values. Now another way to understand the series is that it can be considered like a fixed length dict where you can understand dict whatever we have discussed in the previous lectures.

(Refer Slide Time: 06:30)



The slide is titled "Transforming Unstructured Text" and contains the following content:

- Examples:
 - Gutenberg corpus: collection of English books
 - Brown Corpus: American English texts of varying genres
 - Reuters corpus: collection of Reuters news stories
- Formats of text documents, for example:
 - Simple text files
 - Word processor files
 - Standard exchange format such as XML

At the bottom of the slide, there are logos for IIT ROORKEE and NPTEL ONLINE CERTIFICATION COURSE, and a small number '4' in the bottom right corner.

That key you know value appear that we have in dict, so as elements of dict. So, here also a kind of mapping is happening, the mapping is between index values to data values, so that kind of a mapping is being done in series. So, in a way we can consider series as a dict kind of object dict like object. Now the next the construction part, so how do we create a series object, so typically created using a series function so 1 code 1 example code example we have given here series 1.

And this is the series function `pd.series` and within the parenthesis you can see there that we are passing a list of you know 4 values here. In this first example we have not specified any index , so we have a default index in this which is 0 to n-1 where n is the total number of you know elements that are going to be present in the series. So, you know that is going to be the default

index.

So, in this case if I run this we will have created a series objects series1, so if I run this again you can see here in the output series1 with 4 values and indexing again just like in other data structure and indexing starts with 0, so this is also 0 indexed. So, the index as you can see is displayed explicitly here the first column is actually the index 0, 1, 2, 3 and then we have the values you know fall of the series, that is 4 values 22, 33, 44 and 55.



And then the data type of these values integer64 bit, so this is the series object, this is the simpler way of you know constructing or creating a series object here. So, let us move forward, so next thing is you know series properties or attributes, so let us first talk about the values attribute. So, this values attribute is like an array object, so if I run series1.values the only you know the 1 column of values that we had in series1 that is going to be or printed here, produced as output here.

So, series1.output you can see in the output it is clearly mentioning array and the values there, so you can see the column that we have it is actually. So, you can see how the data structures of pandas package series and data frame they are actually based on arrays. And arrays they are actually we have discuss in the NumPy you know package. So, this is the certain you know similarities and foundation that you should be able to notice here.

(Refer Slide Time: 11:31)

Transforming Unstructured Text

- Process of transforming unstructured text involves following steps
 - Cleaning Text
 - Tokenization
 - Stemming or lemmatization
 - Vector generation
 - Feature extraction and selection
- Order of execution
 - For few of the above steps
 - Depends on the analytics problem and data

 IIT ROORKEE |  NPTEL ONLINE CERTIFICATION COURSE

5

Now next attribute is index, so index we have talked about however we have an attribute and we can access using that. So, `series1.index`, so this is quite similar to you know when we call this function `range n`, so the output is quite similar to that, so this index is based on or similar to the functionality of this function. So, if I run `series1.index` you can see in the output we have range index you know start point, stop point and the step over there.

So, this is how the values part and the index part can be accessed using these attributes, now let us move forward. Now as we said that the explicit index that we have for series object you know this can be customized. So, we can customize this you know as a list of labels, so let us look at this example `series2` and on the right hand side we have `pd.series`.

And you know within the parenthesis we are passing the list of value, sequence of values and then index we have mentioned 4 levels here a, b, c, d you can see 4 values and correspondingly 4

indices we have mentioned here. So, if I run this I would have created a you know customized index and you can see the first column is referring to the index the custom index that we have just created starting from a then b then c and d and then we have the you know a 1 column of values then 22, 33, 44, 55.

Let us move forward, so we can also check the presence or absence of labels in a series. So, like we discuss that you know the series can be considered as a dict kind of object, so here also key value kind of thing you know those kind of you know presence and absence, we can find out here. So, let us see whether this index c is present in series2, so if I run this line number 9 and you can see it is present as you can see in the output number 8.

So, therefore in the output 9 comes out to be true similarly another example let us see this label e whether it is present in series2 or not. So, as you can see in the output 8, it is not present so therefore we should be expecting false as the output, so that is the output that we have got. Similarly let us also retrieve these values and index attribute for this series2, so series2.values, so you can see this array with 4 values and index, you can see a, b, c, d.

So, this is how we can create kind of index and do few things check presence and absence of labels and access the attributes also just like the you know regular you know series object. So, accessing values using label indices, so how we can do that, so for this again brackets operator we can use. So, in this case series 2 within the brackets we can specify the label here, so for an example a, so series2 within brackets a.

(Refer Slide Time: 13:41)

The slide is titled "Transforming Unstructured Text" and contains a bulleted list of information about tokenization. The list includes a definition of tokenization, the nature of tokens, and common tokenization techniques. The slide footer includes the IIT Roorkee and NPTEL Online Certification Course logos, and the number 7.

Transforming Unstructured Text

- Tokenization
 - Process of breaking down or splitting textual data into smaller meaningful components called tokens
 - It will involve breaking down a text corpus into sentences, and each sentence into words
 - Tokens are
 - Independent and minimal textual components that have some definite syntax and semantics
 - Tokenization techniques include sentence and word tokenization

IIT ROORKEE NPTEL ONLINE CERTIFICATION COURSE 7

So, in this fashion we would be able to access the value corresponding to this index label a, so that comes out to be 22. Similarly if we want to check you know if you want to you know access value of our multiple indices, so that can also be done. So, within series2 and within brackets we will have to specify a list of labels. So, in that list of label you know in this next line of code have indicated the 2 labels a and c.

So, let me run this and you can see or output number 14, we have 2 levels a and c and corresponding values as well. However you know the order is not really important here when we are accessing values in this fashion. So, if I run something like this series2 and within brackets I am specifying a list of these labels. So, if I run this and you can see that whatever order we had indicated in the brackets operator the output is also according to that order.

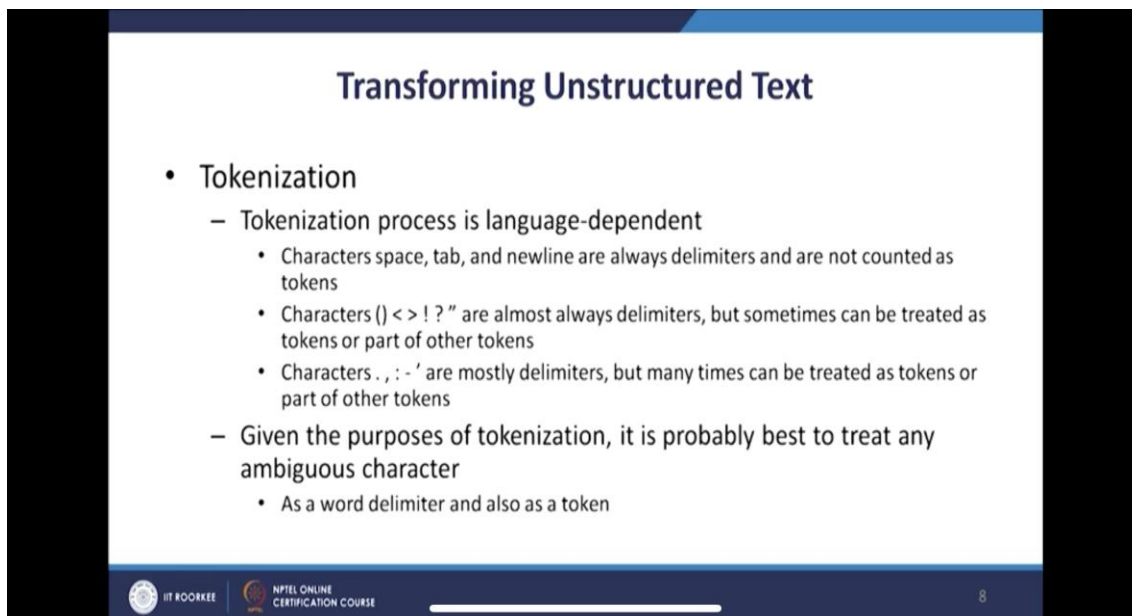
So, in any order, so order in that sense is not important that like order is fixed rather whatever we are indicating as per that the output is going to be retrieved. So, let us move forward, now here

because in this data structure that we just talked about series the link between in the index and value that is very important. And it is suppose to be a stable link it is suppose to be robust link, so whenever we are operating on the values of the series object any operation that we are performing.

So, this link would not be disturbed, so that is the case here also, so index value link is not disturb by execution of various operations. So, another aspect is that automatic alignment of index label with corresponding values and results just like in R platform. So, in previous courses that I have taken you know for the NPTEL platform business analytics and data mining modeling there in the R platform also you know you would always see whenever we are applying any operation the index this link between index and columns and values that is very consistent, the same thing is here.

So, automatic alignment is inbuilt in this functionality, so we will do a filtering example here, so we will take the series2 object and within the brackets we have put this conditional statement series2 greater than 0. So, all the values which are greater than 0 they are to be you know, so this will return a boolean you know output and only those indices which comes out to be true, only those are to be displayed.

(Refer Slide Time: 16:24)



The slide is titled "Transforming Unstructured Text" and contains a list of bullet points under the heading "Tokenization". The bullet points describe the tokenization process as language-dependent and list various characters that act as delimiters or tokens. The slide also features logos for IIT ROORKEE and NPTEL ONLINE CERTIFICATION COURSE at the bottom, along with the number 8.

Transforming Unstructured Text

- Tokenization
 - Tokenization process is language-dependent
 - Characters space, tab, and newline are always delimiters and are not counted as tokens
 - Characters () < > ! ? " are almost always delimiters, but sometimes can be treated as tokens or part of other tokens
 - Characters . , : - ' are mostly delimiters, but many times can be treated as tokens or part of other tokens
 - Given the purposes of tokenization, it is probably best to treat any ambiguous character
 - As a word delimiter and also as a token

IIT ROORKEE NPTEL ONLINE CERTIFICATION COURSE 8

So, if I run this you can see since all the values are greater than 0 therefore we have got you know this kind of output, consisting of all the values of this series object. So, let us move forward scalar multiplication, so we can multiply this series you know with some scalar value like 9. And in the output you would see that all the elements of this series object they have been multiplied.

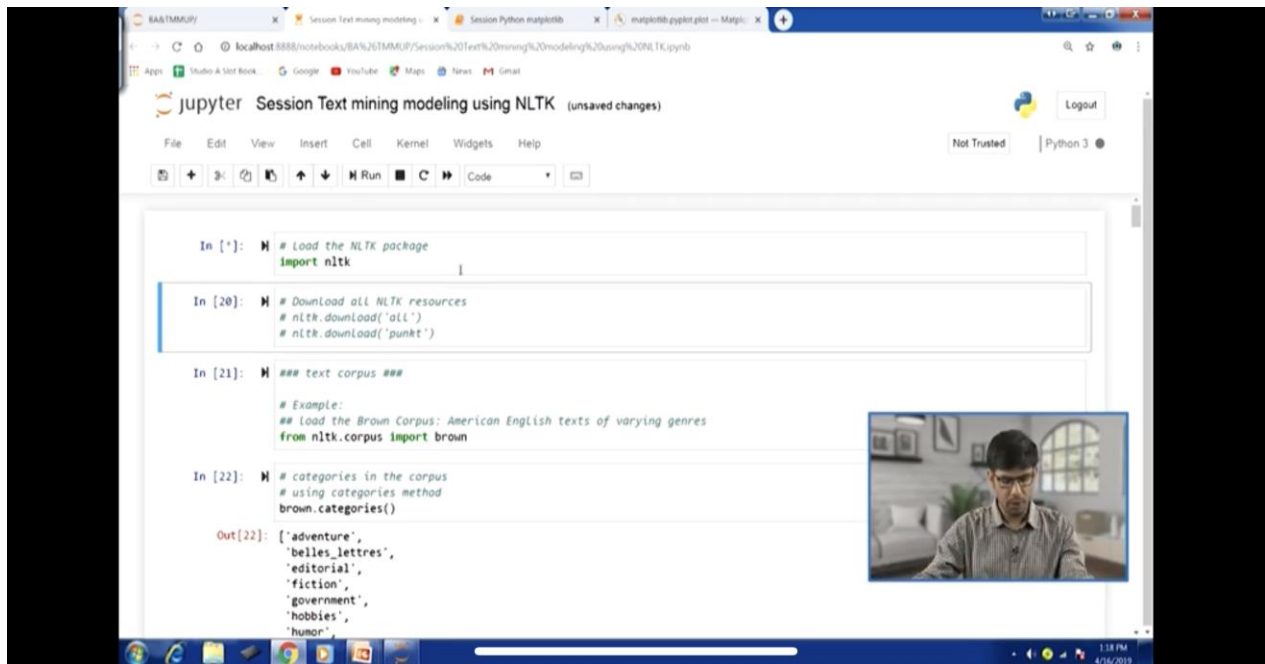
So, in a sense broadcasting has happened you know over the rows, so for all the rows this scalar you know value has been multiplied and broadcasting has been done. So, broadcasting is an also an important concept in the python platform, so through a few examples we would be you know covering this however we would not be going to for more details of that aspect.

Now let us take another example let us take this math operations will import NumPy here also. So, we will take exponential of the series2 so `np.exp` is the function that we are calling and we are passing on this you know data structure series2, this object here. So, let us see how this function is going to work because now NumPy, all NumPy functions are going to work here because essentially as we have discuss these 2 data structures series and data frame, they are based on arrays.

So, let me run this, so you can see in the output all the you know series values an exponential have been taken and that output has been displayed here in the output number 18. Now let us take another example, so in this case we will take you know another type of data structure and convert you take that you know data and convert it into a series kind of object.

So, we will convert python dict object into a series of object because as we discuss that we said that series can also be considered like a dict kind of you know some behavior is like you know that kind of thing. Because mapping between index and values is also there in series just like in the big dict object where we have the mapping between keys and value pairs. So, in this case you can see we have taken this example pin code and a few towns or cities in the Uttarakhand state of India they have been you know given as an example here.

(Refer Slide Time: 21:48)



```
In [*]: # load the NLTK package
import nltk

In [20]: # Download all NLTK resources
# nltk.download('all')
# nltk.download('punkt')

In [21]: ### text corpus ###

# Example:
## Load the Brown Corpus: American English texts of varying genres
from nltk.corpus import brown

In [22]: # categories in the corpus
# using categories method
brown.categories()

Out[22]: ['adventure',
          'belles_lettres',
          'editorial',
          'fiction',
          'government',
          'hobbies',
          'humor']
```

So, pin codes we have for these towns root Roorkee and 247, Dehradun, Haridwar and Nainital. So, if I run this we will have this dict and we can pass on this dict object to our pd.series function and this function will handle you know take this data and convert it into a series object. So, if I run in the next line and let me run and access the value, so you can see that the key part of the dict object that we had that has been taken as the you know index in the series object.

So, in code dict object that we had city names, town names, Roorkee, Dehradun, Haridwar and Nainital. Those have been taken as you know indices in the series object. And the pin codes that we had in the dict object the value part of the dict object we had the pin codes, they have been taken as the value part in the series object. So, you can see how this you know translation is happening here, so we can easily convert dict object into a series kind of thing.

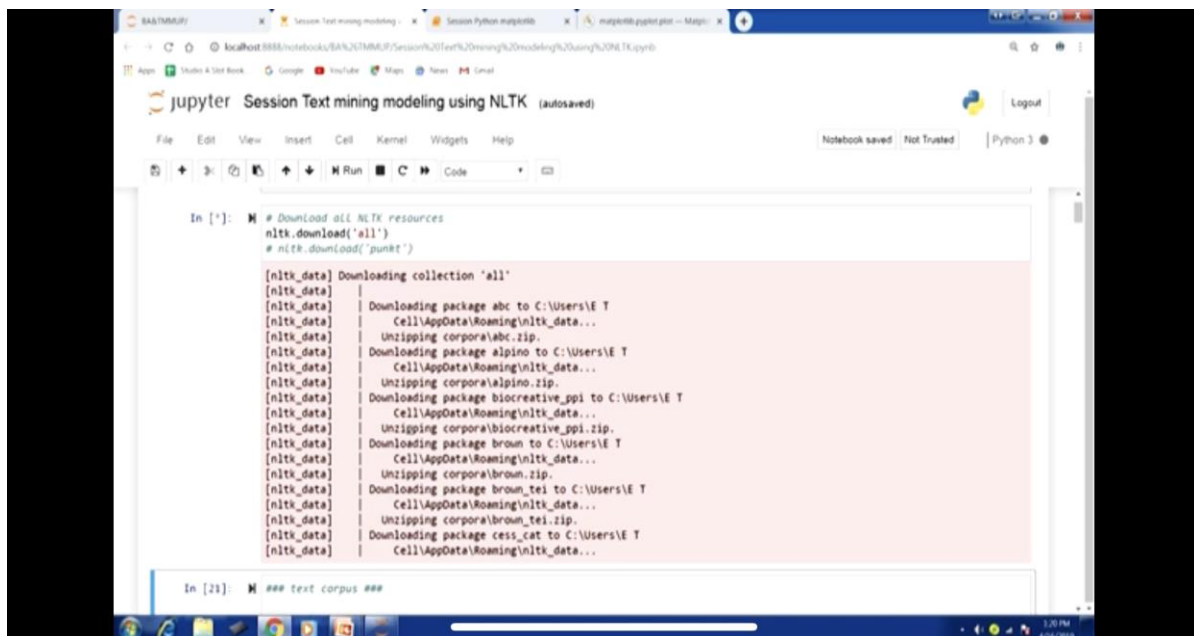
Now if we want to access the index part that we have just created for series3 object. So, let me run this line of code you can see index with 4 terms names of 4 terms here and in this fashion you can produce this output. So, now sometimes we might be require to change the index also, we might like to add few more rows or something in our data, so for that we will have to modify the index also.

So, you know that can be run using the index argument you know in the series function. So, let us take this example cities and this is a list of you know these town names, so in the cities we have added one more Dehradun, Haridwar, Nainital, Roorkee and Tehri Garhwal. So, here you would see we have change the order also here, so in the next line of code, so let me first create this object cities this list object.

And now we are passing on this list object cities into the second argument in the series functions. So, first one is pin code the dict object that we had created, so and the next one is the index, so now this because we are clearly specifying we are using this keyword argument here and clearly is specifying our index here. So, this index would be taken while constructing the series object number 4 series4, so data value part would be taken from pin code.

And then the index part would be taken from this index argument the cities and this series4 you know object is going to be constructed. So, let me run this and you can see in the output that first is as per the order as specified in the cities index Dehradun, Haridwar, Nainital and Roorkee. So, order is as per that and then Tehri Garhwal but we did not have the pincode value for Tehri Garhwal in the dict object, pin code.

(Refer Slide Time: 24:09)



```
In [1]: # Download all NLTK resources
nltk.download('all')
# nltk.download('punkt')
```

```
[nltk_data] Downloading collection 'all'
[nltk_data] |
[nltk_data] | Downloading package abc to C:\Users\E T
[nltk_data] | Cell\AppData\Roaming\nltk_data...
[nltk_data] | Unzipping corpora\abc.zip.
[nltk_data] | Downloading package alpine to C:\Users\E T
[nltk_data] | Cell\AppData\Roaming\nltk_data...
[nltk_data] | Unzipping corpora\alpine.zip.
[nltk_data] | Downloading package biocreative_ppi to C:\Users\E T
[nltk_data] | Cell\AppData\Roaming\nltk_data...
[nltk_data] | Unzipping corpora\biocreative_ppi.zip.
[nltk_data] | Downloading package brown to C:\Users\E T
[nltk_data] | Cell\AppData\Roaming\nltk_data...
[nltk_data] | Unzipping corpora\brown.zip.
[nltk_data] | Downloading package brown_tei to C:\Users\E T
[nltk_data] | Cell\AppData\Roaming\nltk_data...
[nltk_data] | Unzipping corpora\brown_tei.zip.
[nltk_data] | Downloading package cess_cat to C:\Users\E T
[nltk_data] | Cell\AppData\Roaming\nltk_data...
```

```
In [21]: ### text corpus ###
```

So, therefore that is missing and the missing values in this package python and in the python platform and in this package as well, they are specified as nan. So, you can see the same has happened, so while we are changing index if there are any unmatched indices. So, for them you know those indices are going to be created, so a sort of a union is going to be formed.

So, here you would see as per the pin code the dict object we have a number of indices and as per the index you know argument index keyword argument we had specified cities. So, while we are taking values from there and indices from here the matching will happen and for the matched values we have the indices, correct alignment, automatic alignment, indices and values wherever we do not have you know we have you know one thing missing, a new row is going to be created.

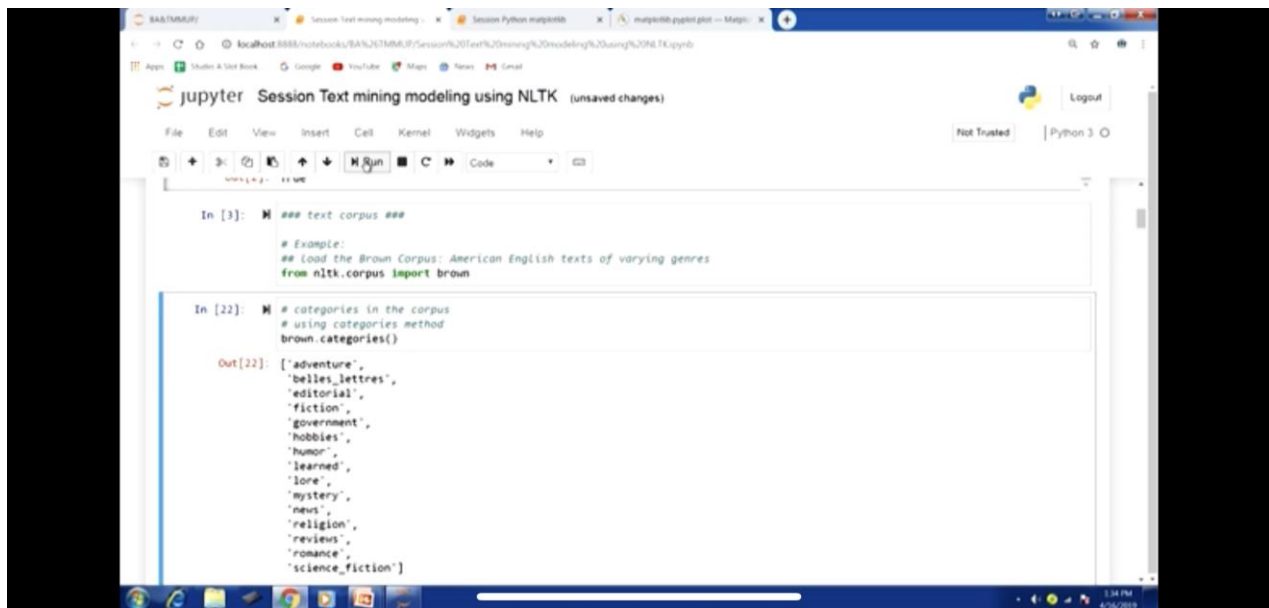
And you know if the value is not there then nan is going to be displayed, so that kind of output has been produced here. Now if you want to detect missing data in these objects, so for that we have these function is null and not null function, so we can use them. So, for example series4 object we can call this pd.is null this function and you can within the parenthesis, we can pass on this series4 object here.

So, if I run this and you can see in the output number 25 which you know data is missing, so in this case you can see is null. So, you can see Tehri Garhwal for that value part is missing, that comes out to be true others are false . So, you can see also data type d type is also indicated as bool similarly not null. So, this is another function to perform the similar kind of processing, so here you can see that you know only a first 4 values Dehradun, Haridwar, Nainital, Roorkee for them it comes out to be true, the last one it is false.

Because you know it is actually null, so we can in this fashion we can detect any missing data in our series object. Now these were the is null and not null these are the function that we had use, we also have the corresponding methods is null and not null we have these corresponding methods. So, we can also call them, so series4.is null, so in this fashion we can call these methods, so let me run this and you can see the output.

Similarly series4.notnull method also we can call this one also, so in this fashion we would be able to produce this output. Now let us look at the alignment feature, so once again, so series3 this 3+series 4 if we perform this addition operation here between these 2 series objects. So, let me run this and you can see that the values in the output 29, so the pin codes you know though pin code is a categorical variable here.

(Refer Slide Time: 27:47)



```
In [3]: ### text corpus ###  
  
# Example:  
## load the Brown Corpus: American English texts of varying genres  
from nltk.corpus import brown
```

```
In [22]: # categories in the corpus  
# using categories method  
brown.categories()
```

```
Out[22]: ['adventure',  
          'belles_lettres',  
          'editorial',  
          'fiction',  
          'government',  
          'hobbies',  
          'humor',  
          'learned',  
          'lore',  
          'mystery',  
          'news',  
          'religion',  
          'reviews',  
          'romance',  
          'science_fiction']
```

We are still into our python part, so these are numeric values, so they have been added up, so you can see Dehradun the you know pin code series3 and series4 both had same values, added up Haridwar, Nainital, Roorkee added up Tehri Garhwal where we did not have value and it is nan and you can see everything is aligned. So, whenever we are performing these kind of mathematical arithmetic operations you can see the alignment between index and values is maintained and accordingly the operation is apply.

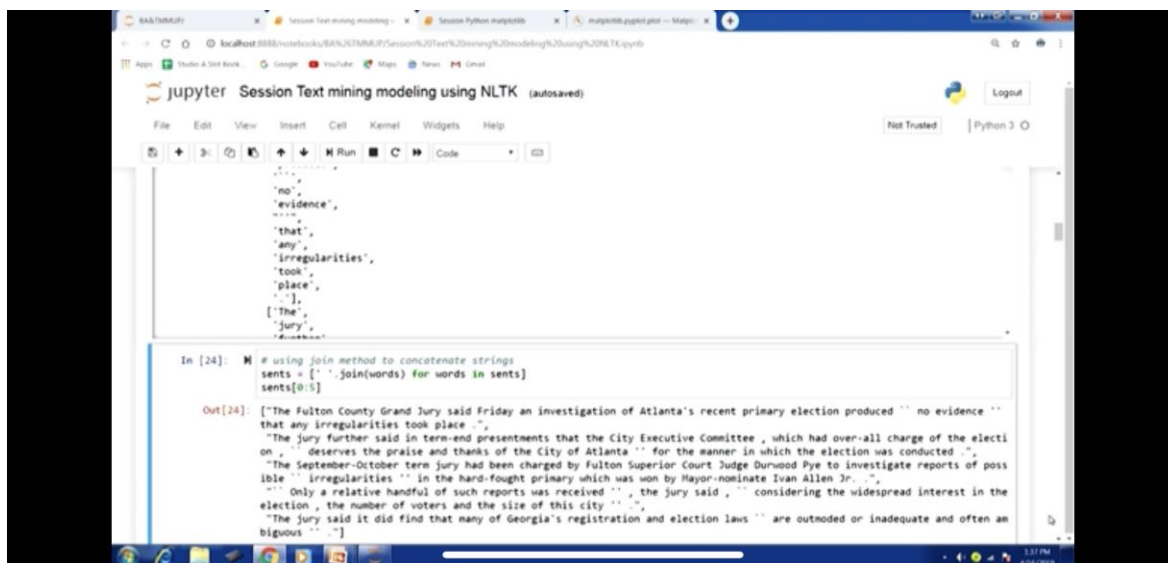
So, for the matched you know indices the operation will work for the unmatched of course you know nan is going to be there. Now let us move forward, now we have another you know aspect another attribute here name attribute we have for series object. So, series4.name, so in this case we can specify the name attribute for this series object as pin code, we have the name attribute for the index also.

So, for the series object as well as it is index, so we can use series4.name to specify the name attribute of the series object and we can type code series4.index.name to specify the name of the index. So, in this case it is about you know series about pin code you know values to pin code numbers. So, if I run this name would be you know defined and similarly for index those are actually city you know city in names.

So, now let us have a look at the series4 object, so you can see that city is you know appended like a header there on top of the list of towns in the first column that we have city you can clearly see. And for the name of the series you can see last line, last row in the output name pin_code, so for this series object this name is also indicated there. Let us move forward, next aspect about this is changing index, so we also have you know index attribute.

So, we can use that also to change index, so let us have a look at the you know index values here, this is our series. And in this if you look at the first column which are the index value 0, 1, 2, 3. So, for some people they might not prefer the 0 indexed thing, so for that they can customize their index and they can make it like 1, 2, 3, 4. So, in the next line of code we are doing exactly that series1.index and we have passing this list sequence of values 1, 2, 3, 4

(Refer Slide Time: 30:48)



The screenshot shows a Jupyter Notebook interface with the following content:

```
...
'no',
'evidence',
...
'that',
'any',
'irregularities',
'took',
'place',
...
['The',
'jury',
...]
```

```
In [24]: # using join method to concatenate strings
sents = [' '.join(words) for words in sents]
sents[0:5]
```

```
Out[24]: ["The Fulton County Grand Jury said Friday an investigation of Atlanta's recent primary election produced " no evidence "
that any irregularities took place .",
"The jury further said in term-end presentations that the City Executive Committee , which had over-all charge of the electi
on , " deserves the praise and thanks of the City of Atlanta " for the manner in which the election was conducted .",
"The September-October term jury had been charged by Fulton Superior Court Judge Durwood Pye to investigate reports of poss
ible irregularities " in the hard-fought primary which was won by Mayor-nominate Ivan Allen Jr. ",
" Only a relative handful of such reports was received " , the jury said , " considering the widespread interest in the
election , the number of voters and the size of this city " .",
"The jury said it did find that many of Georgia's registration and election laws " are outmoded or inadequate and often an
biguous " ."]
```

So, if I run this I would be able to change the index, so earlier when we change the index we use

the index argument in the series function. Now this time and we are changing index we are using the index attribute to assign it a new index. So, these are 2 different ways to perform the same thing, so now let us have a look at the output. So, you can see output number 35 that series1 and the first column index has been changed 22, 33, 44, 55 and the corresponding indices are 1, 2, 3, 4.

So, those were the main points about the series data structure, series object. Now let us move forward to another very important you know data structure that is data frame. So, in our previous courses **we** as we have seen that we have been typically importing the excel you know datasets you know into a data frame kind of you know a variable in R platform.

Similarly here in the python platform we have data frame object and this is also you know to built with similar kind of data set, excel you know, tabular format excel kind of data set. So, let us talk about this data structure, so data frame, so the name is also happens to be the same and represents physically 2-dimensional data in a tabular format of rows and columns. And it is like a dict of series elements having a common row index.

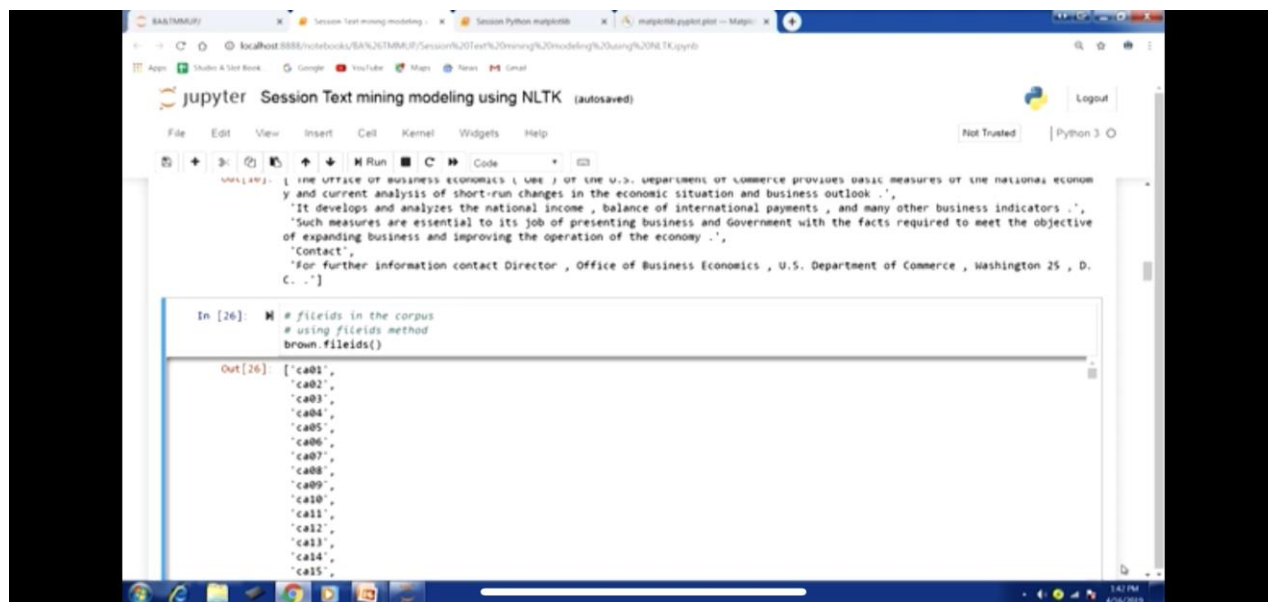
So, if you know if we can have a dict and all the you know there we have key value combinations. So, all the values they can be you know series elements and the keys that we have if the keys are common, so that is how we can consider this data frame. So, common key and multiple you know series elements, so that kind of would dict we can consider a data frame as that kind of dict object.

So, data frame have both row and column index, so in the series we have the explicit index but only the you know row index. In this case in the data frame will have the row and column index, so will have the indexing for the multiple columns that will have and also as usual we will have the indexing for the several rows that we are going to have in our data. So, row index is similar to series index that we have discussed and default is going to be 0 to n-1 and of course we can customize it as we have seen.

Now column index is nothing but variables name because essentially these we are suppose to

work with the heterogeneous data different columns, representing different variables they could be of different they could be having different data types. So, therefore the column index is typically variables names and by default they are sorted alphabetically. So, columns can be variables of different value types as you can see in numeric string boolean anything.

(Refer Slide Time: 33:45)



So, these are so when we say numeric string boolean 3 this term logi is coming from the language perspective. That is the you know python platforms perspective that is different data type, different value types are defined you know one of these numeric string boolean. But from the analytics perspective we talk about you know categorical continuous variable however you can clearly see that like we have discuss in our previous courses how they can be easily mapped.

Now let us focus on the construction part of data frame, so how a data frame you know object is created. So, created using a data frame function, so let us take an example we will take year wise metro population data for few metros in India. So, here we are you know defining a dict object, dict1 here and you can see the first key is metro where we have a list of string values indicating the metro names and repetition is also there because this is a year wise data.

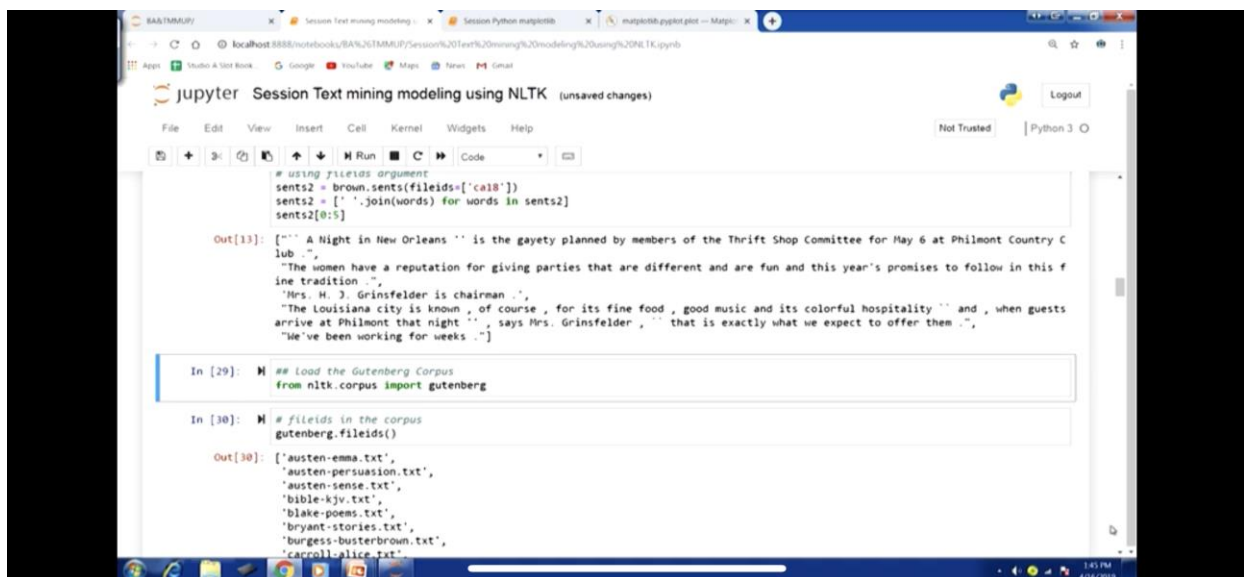
So, for different years and for these metros we will have the population in crores, so hypothetical example here. So, you can see matt metro 6 values Delhi, Delhi, Delhi and Mumbai, Mumbai,

Mumbai because in the second key value combination year we have data for 2011, 2012, 2013 and then you can see the repetition 2011, 2012 and 2013. So, from 2012 to 2011 to 2013 will have data for these 2 metros Delhi and Mumbai and you can see population in crores few numbers we have given for this example.

So, if I run this you will have the dict1 object and then we can pass this dict1 object to create a data frame. So, we can call this function pd.data frame and pass on this dict1 object and then we will have our data frame. So, let me run this and let us have a look at the output, so you can see output number 38 and you can see a table has been displayed there. So, you can see 3 columns are there metro, year and popcr, that is population in crores.

So, in metro you can see you know first row Delhi then Delhi then Delhi the next 3 rows Mumbai, Mumbai, Mumbai, years 3 years 2011 and 12 and 13 and same for Mumbai and population crores. And you can see the all this is 0 index, so very first column explicit index in column is there 0, 1, 2, 3, 4, 5, so this is the main structure of our data frame object. Now as you would expect that we would be dealing with you know larger data sets.

(Refer Slide Time: 36:42)



```
# using fileids argument
sents2 = brown.sents(fileids=['ca18'])
sents2 = [' '.join(words) for words in sents2]
sents2[0:5]

Out[13]: ["" A Night in New Orleans "" is the gayety planned by members of the Thrift Shop Committee for May 6 at Philmont Country Club .-
"The women have a reputation for giving parties that are different and are fun and this year's promises to follow in this fine tradition .-
"Mrs. H. J. Grinsfelder is chairman .-
"The Louisiana city is known , of course , for its fine food , good music and its colorful hospitality "" and , when guests arrive at Philmont that night "" , says Mrs. Grinsfelder , "" that is exactly what we expect to offer them .-
"Me've been working for weeks .-"]

In [29]: # Load the Gutenberg Corpus
from nltk.corpus import gutenberg

In [30]: # fileids in the corpus
gutenberg.fileids()

Out[30]: ['austen-emma.txt',
'austen-persuasion.txt',
'austen-sense.txt',
'bible-kjv.txt',
'blake-poems.txt',
'bryant-stories.txt',
'burgess-busterbrown.txt',
'carroll-alice.txt']
```

So therefore it will run into you know several 100s and 10000s of rows. So, if we just want to have a look at you know few values from the data set we can use this particular you know

method here `df.head`. So, it will just you know display first 5 rows of data, so if I run this you can see. In this case we just have you know 6 rows of data. So out of that first 5 rows are going to be displayed as you can see in this output number 31. So, this is how you know we can actually work around you know data frame and series object.

(Video Ends: 32:53)

Now we have lot more to discuss about a data frame as well as series and how we can you know apply different operations you know do certain processing as per different analytics scenarios. So, we will continue our discussion in this lecture, we will stop here, thank you.

Keywords: dataframes, data structure, pandas, prediction, exception, classification.