Business Analytics and Text Mining Modeling Using python

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

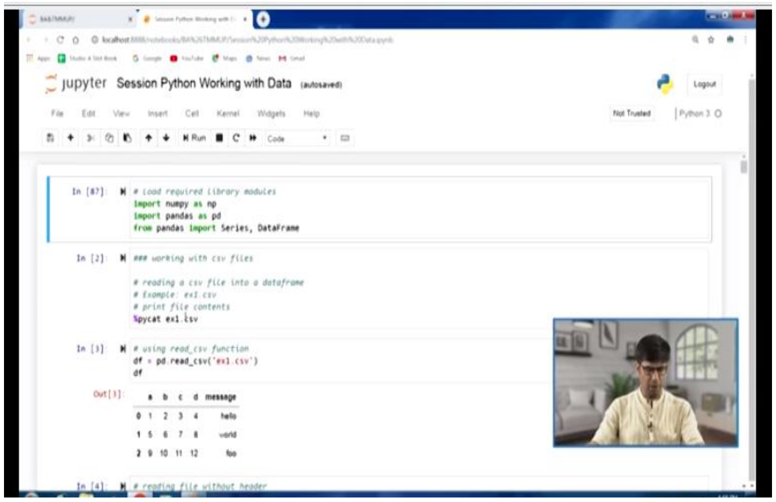
Lecture-30
Python Working with Data-Part I

Welcome to the course business analytics and text mining modeling using python. So, in previous lecture we were able to finish another module that was on python pandas package. So, in this particular course we have been able to cover the introductory part of the text mining and then the python for the analytics which is the you know covering the major number of lectures in this course.

Because the python is the platform which we would be using ex10sively for text mining. So, we were able to cover the basics for python, the building capabilities, the numerical python package and the pandas. Now we are coming to the part where we would be talking about how we can use python to work with data. So, we would be starting those aspects in this particular lecture, so let us start.

So, as you would expect that in this part we would be using some of the packages and libraries that we have discussed in the previous lectures. So, we would be in this first thing will load required library modules.

(Refer Slide Time: 01:35)


jupyter Session Python Working with Data (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run
Code
Not Trusted
Python 3
Logout

```
In [87]:  # load required library modules
          import numpy as np
          import pandas as pd
          from pandas import Series, DataFrame

In [2]:   ### working with csv files

          # reading a csv file into a dataframe
          # Example: ex1.csv
          # print file contents
          %pycat ex1.csv

In [3]:   # using read_csv function
          df = pd.read_csv('ex1.csv')
          df

Out[3]:
             a   b   c   d  message
          0  1   2   3   4    hello
          1  5   6   7   8    world
          2  9  10  11  12      foo

In [4]:   # reading file without header
```

So, first thing is NumPy as np pandas as pd and certain library within pandas series and rid of data frame that we would be using quite of10.

(Video Starts: 01:44)

So, let me run this, so all these are required, then first thing that we typically do is that loading required library modules and you know. So, first thing is NumPy and pandas and then certain library modules there, so let me run this. And the first thing while discussing about working with data first thing will talk about the csv files many databases they are stored in csv file and excel files.

So, in this starting lecture on working with data will focus on csv files and excel files, so let us start with the csv. So, first thing reading a csv file into a data frame, so data frame is the particular data structure python object where we can actually you know import the csv data. So, let us take example of this file ex1.csv before we go ahead and import the data stored in this particular file into a data frame in this python environment.

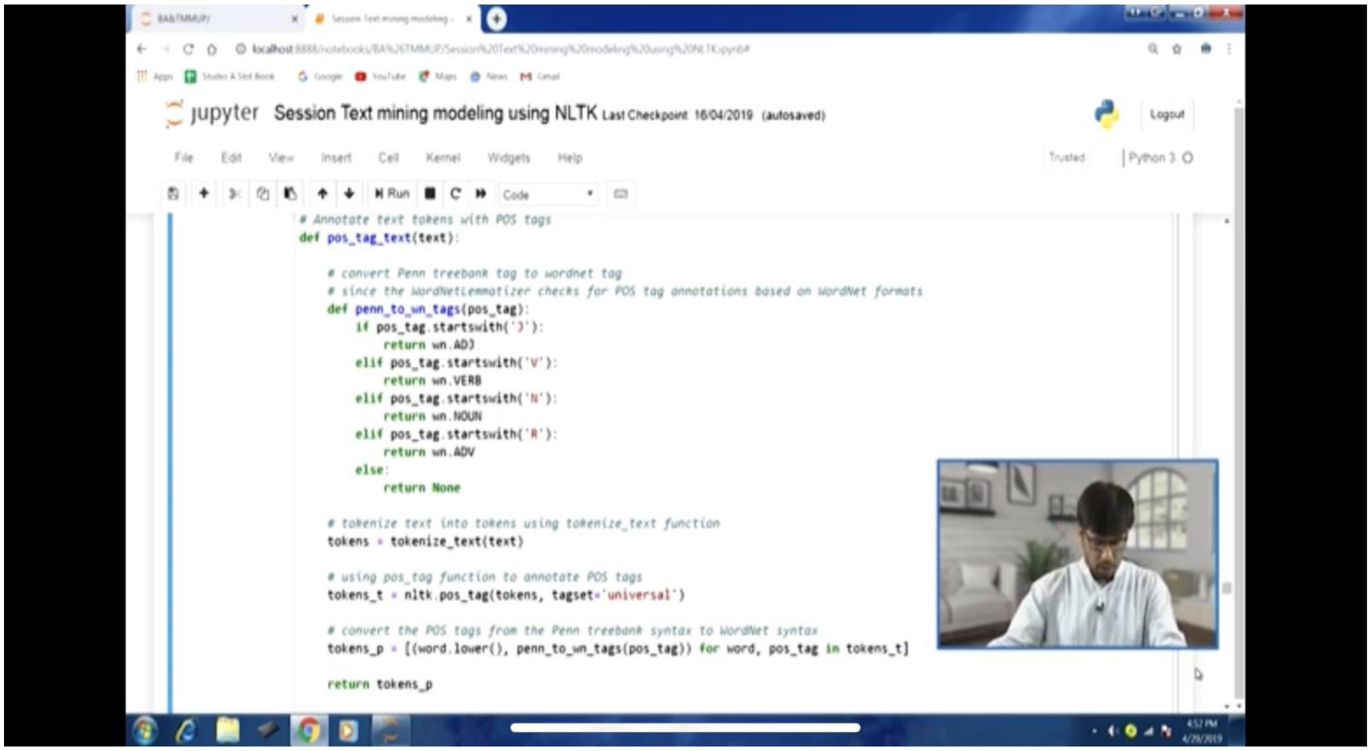
Let us have a look at the con10ts of this particular file, so as we discuss in the python basics lecture we can use certain magic commands for these purposes. So, in this case we are using this %pycat you know command here, so %pycat and the name of the files in this ex1.csv. So, if I run this you would be able to see the con10ts of this particular file as you can see in the popped up window at the bottom of this page that first we have ABCD message.

So, these are the headers then we have the you know 1, 2, 3, 4 hello and 5, 6, 7, 8, so these are the values. So, small you know data said that we have in this file for a demonstration purpose, so that you were able to see. Now looking at this file you could see that the values were separated by commas, so what the csv related function was for example read_csv, they could be used to actually you know import that data into a read that particular data into a data frame.

So, next line of code you can see on the left hand side we have df and on the right hand side we have pd.read_csv. So, this is the function that we would be using within the parentheses we are passing the you know file path of this csv data set that we have. So, in this case this file is currently stored in the current working directory itself, so I just have to specify the file name, so that is the path itself in this case.

So, if I run this you would be importing the data we would be loading the data into a data frame, so you can see in the output 4 that ABCD message and the 3 rows 0, 1, 2 and the data has been loaded into the python environment. So, this is how data stored in a csv file can be easily imported into a data frame object in python environment. Now sometimes some of the csv files might not be carrying the header rows, so how to deal with those scenarios.

(Refer Slide Time: 04:14)


jupyter Session Text mining modeling using NLTK Last Checkpoint 16/04/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

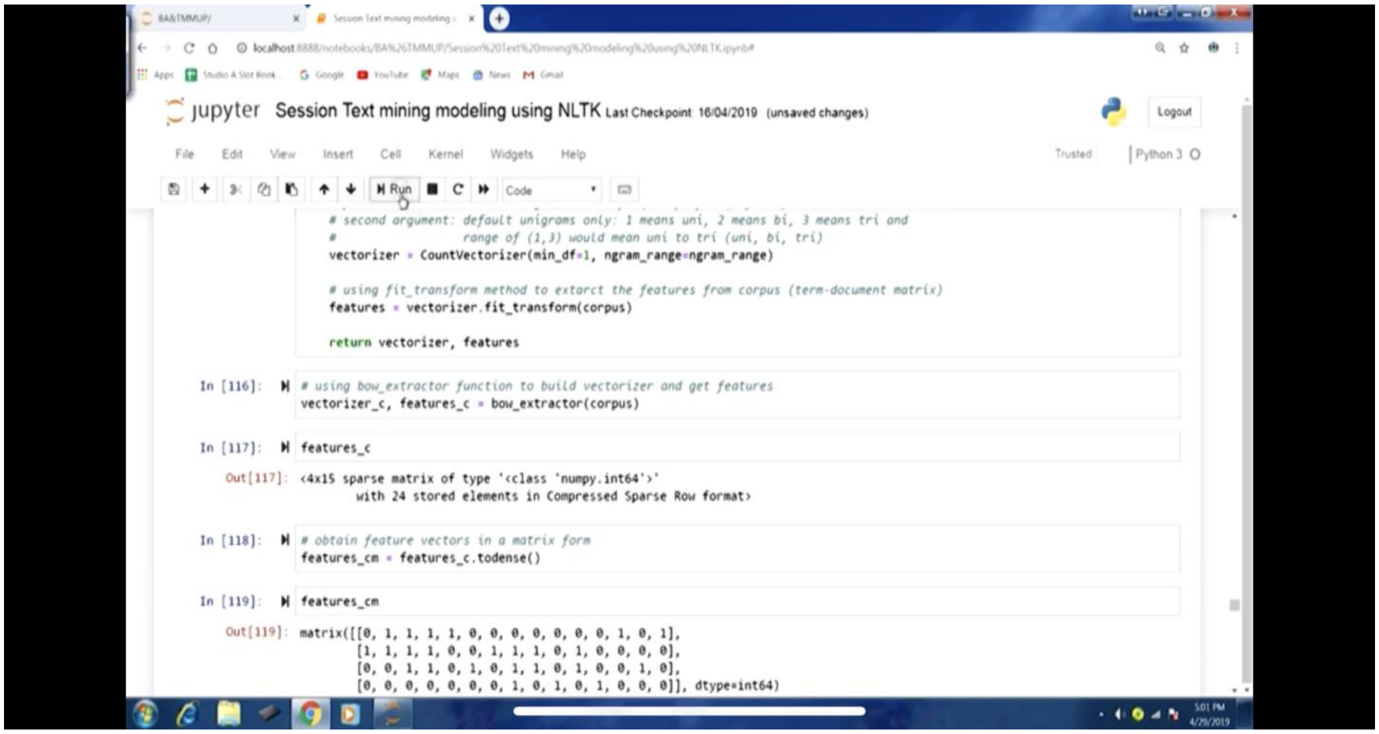
```
# Annotate text tokens with POS tags
def pos_tag_text(text):

    # convert Penn treebank tag to wordnet tag
    # since the WordNetLemmatizer checks for POS tag annotations based on WordNet formats
    def penn_to_wn_tags(pos_tag):
        if pos_tag.startswith('J'):
            return wn.ADJ
        elif pos_tag.startswith('V'):
            return wn.VERB
        elif pos_tag.startswith('N'):
            return wn.NOUN
        elif pos_tag.startswith('R'):
            return wn.ADV
        else:
            return None

    # tokenize text into tokens using tokenize_text function
    tokens = tokenize_text(text)

    # using pos_tag function to annotate POS tags
    tokens_t = nltk.pos_tag(tokens, tagset='universal')

    # convert the POS tags from the Penn treebank syntax to WordNet syntax
    tokens_p = [(word.lower(), penn_to_wn_tags(pos_tag)) for word, pos_tag in tokens_t]

    return tokens_p
```

So, let us take an example here again, so reading file without header, so we have this ex2.csv file. So, let us have a look at the con10ts of this file again we will use the %pycat magic command here. So, if I run this here and again you can see in the pop-up window that header is gone, it is the same data that we use in the previous example - the header row. So, let me close this and the next line of code they were we are calling again this read_csv function.

First argument is as usual that file name like we did in the previous command and then we are specifying a header argument here, the keyword argument header here as none because we do not have a header row here. Now default column names you know in case we do not have header they would be by default they would be you know integer numbers would be used, so 0 to nc-1 that means number of columns -1, so that would be used by default in case header is not present.

So, in this case if I run this file and you can see the output the column names the column index has changed and it has become the default one 0 to nc-1. Let us move forward, so in such scenarios where we do not have the header row in the data set we can also use another argument called names which will specify which will allow us to specify the column names and column index for such data set.

(Refer Slide Time:14:11)


jupyter Session Text mining modeling using NLTK Last Checkpoint 16/04/2019 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
    # second argument: default unigrams only: 1 means uni, 2 means bi, 3 means tri and
    #                  range of (1,3) would mean uni to tri (uni, bi, tri)
    vectorizer = CountVectorizer(min_df=1, ngram_range=ngram_range)

    # using fit_transform method to extarct the features from corpus (term-document matrix)
    features = vectorizer.fit_transform(corpus)

    return vectorizer, features

In [116]:  # using bow_extractor function to build vectorizer and get features
           vectorizer_c, features_c = bow_extractor(corpus)

In [117]:  features_c

Out[117]:  <4x15 sparse matrix of type '<class 'numpy.int64'>'
                   with 24 stored elements in Compressed Sparse Row format>

In [118]:  # obtain feature vectors in a matrix form
           features_cm = features_c.todense()

In [119]:  features_cm

Out[119]:  matrix([[0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1],
                   [1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0],
                   [0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0],
                   [0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0]], dtype=int64)
```

So, you can see here we are specifying names a, b, c, d message, so we have we have total 5 columns as you can see in the previous output. So, in this case we can specify the names for those 5 columns and again we can use the read_csv function to read the data and the data frame. So, if I run this you can see in the output 7 in that the header, the column names have been changed.

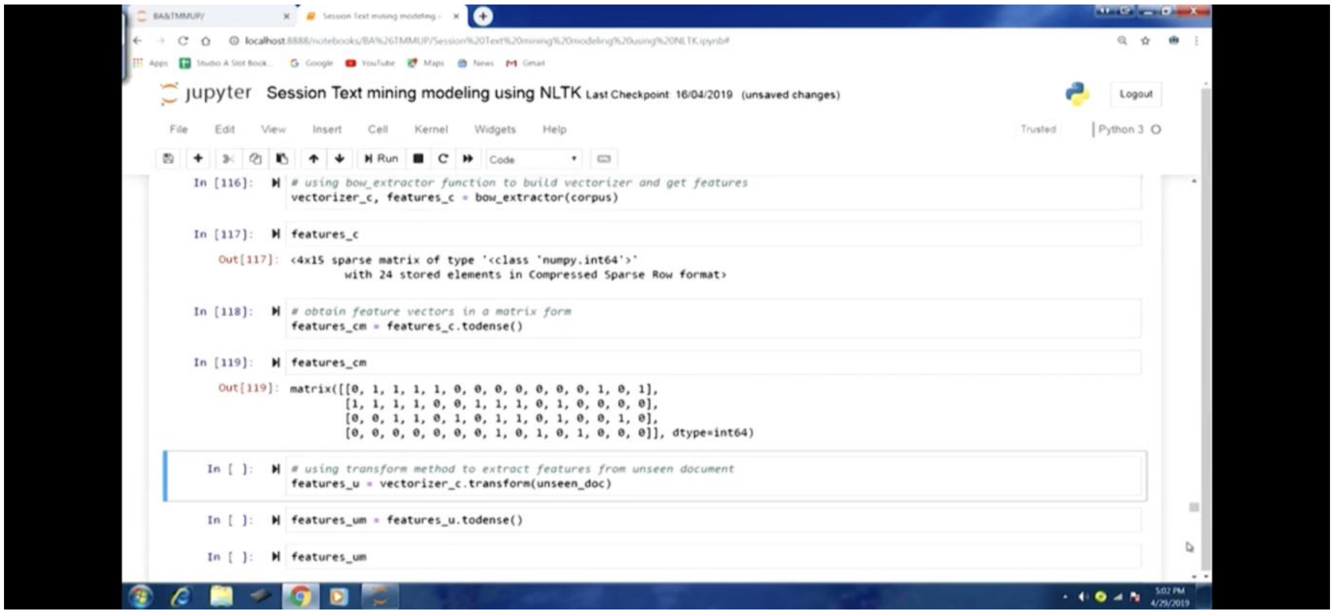
Now sometimes you know we would like to you know you skip certain rows because they have the unwanted data. Many data sets the you know people who have actually develop those data set they typically to explain the data and the variables and many other things that are part of a particular data set, they have comment section there. So, for our analysis purpose for the analytics purposes those comments might not be required.

So, therefore we can skip some of those rows where the comments have been written, so let us take example of this ex4.csv. So, if we look at the con10t of this file using the magic come command let us have a look let me run this. And you can see the popped up window you can see here the few comments you know are there and so we have the functionality here in the read_csv function.

So, that we would be able to escape you know those rows having comments, so in this case we can use this skipped rows argument here. And you can see that we have a specified those particular rows which we want to escape, so if I run this we would be having a data frame with just the data that we want and the comments have been skipped. Now let us move forward the next thing that we can talk about is you know handling missing values.

So, in this particular context if you look at the default behavior here you know the typically when we have missing values will have na or nan in the case of python. So, this is referred as a sentinel value, so the nan that is typically used in python is actually a sentinel value. So, sometimes you know we would like to you know change this default sentinel value and nan that is there.

(Refer Slide Time: 20:06)



Session Text mining modeling using NLTK Last Checkpoint: 16/04/2019 (unsaved changes)

```
In [116]:  # using bow_extractor function to build vectorizer and get features
           vectorizer_c, features_c = bow_extractor(corpus)

In [117]:  features_c

Out[117]:  <4x15 sparse matrix of type '<class 'numpy.int64'>'
                   with 24 stored elements in Compressed Sparse Row format>

In [118]:  # obtain feature vectors in a matrix form
           features_cm = features_c.todense()

In [119]:  features_cm

Out[119]:  matrix([[0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1],
                   [1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0],
                   [0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0],
                   [0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0]], dtype=int64)

In [ ]:    # using transform method to extract features from unseen document
           features_u = vectorizer_c.transform(unseen_doc)

In [ ]:    features_um = features_u.todense()

In [ ]:    features_um
```

And for that again we have certain mechanism we can use to change that, so let us take example of ex5 let us have a look at the con10ts of this file. So, you can see here if we have nan all those things here. So, if we want to change you know something like foo in this particular data frame. So, let me first read it, read this data frame here, so you can see.

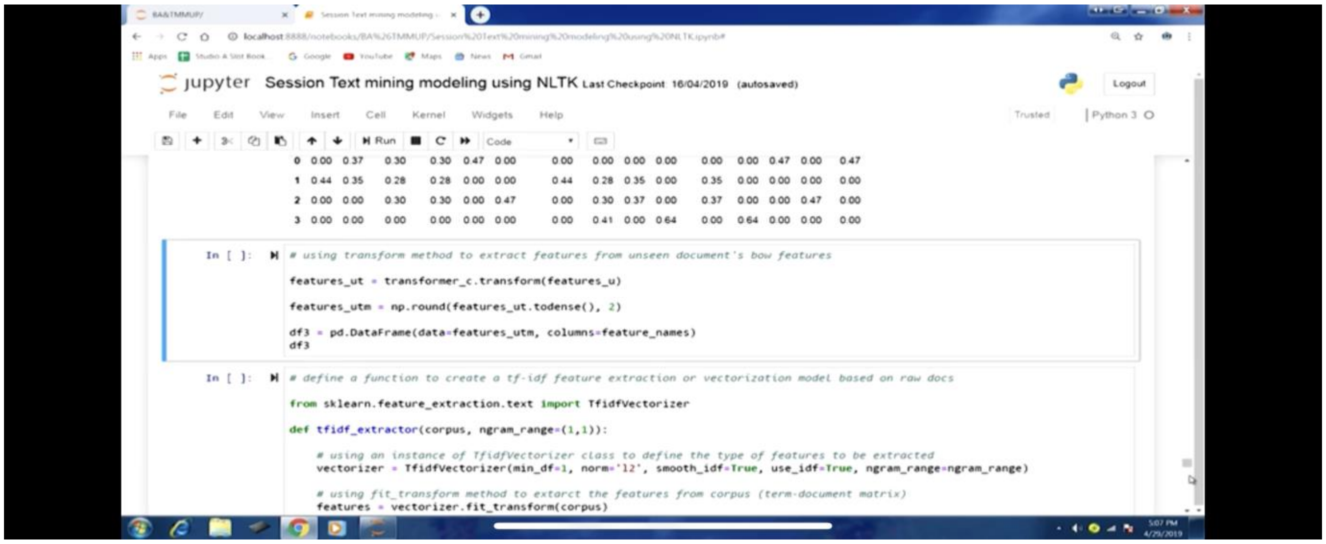
Now if you really focus here we have 2 nans in this data frame and you can see in the last column with the column name message and the third row foo is mentioned. So, this we consider this has to be a meaningless string you know we can change the sentinel and consider this also a kind of na value here. So, how do we you know perform this kind of you know transformation here.

So, we can use using we can use this particular argument nan_values to change this marking of sentinel values. So, making meaningless values such as foo using sentinels, so we can define the sentinels as a **as a** dict object here. So, you can see message column and within message column we have passing a list of you know such values, so in this case foo. So, we would like to change this as well to na.

So, if you look at the next line, let me first define the sentinels and the next line where we are calling read_csp function for this file ex5.csv. And you can see that na_values argument there and we are passing the sentinels be just defined and if I run this. In the output number 13 you can see that the place where we had the foo the last column with the column name message and the last row last cell, so nan is has been you know written over there.

So, in this fashion if there are other if there are missing values with others you know terms we can convert, we can change the sentinel marking and we can you know convert them into any nan as well which is the default term to denote, to indicate, a missing value. So, in this fashion we can you know transform our data set, now let us talk about another aspect of analytics where we would be dealing with larger data sets.

(Refer Slide Time: 25:03)



jupyter Session Text mining modeling using NLTK Last Checkpoint: 16/04/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
0  0.00  0.37   0.30   0.30  0.47  0.00   0.00   0.00  0.00  0.00   0.00  0.00  0.47  0.00   0.47
1  0.44  0.35   0.28   0.28  0.00  0.00   0.44   0.28  0.35  0.00   0.35  0.00  0.00  0.00   0.00
2  0.00  0.00   0.30   0.30  0.00  0.47   0.00   0.30  0.37  0.00   0.37  0.00  0.00  0.47   0.00
3  0.00  0.00   0.00   0.00  0.00  0.00   0.00   0.41  0.00  0.64   0.00  0.64  0.00  0.00   0.00
```

```
In [ ]:  # using transform method to extract features from unseen document's bow features

         features_ut = transformer_c.transform(features_u)

         features_utm = np.round(features_ut.todense(), 2)

         df3 = pd.DataFrame(data=features_utm, columns=feature_names)
         df3
```

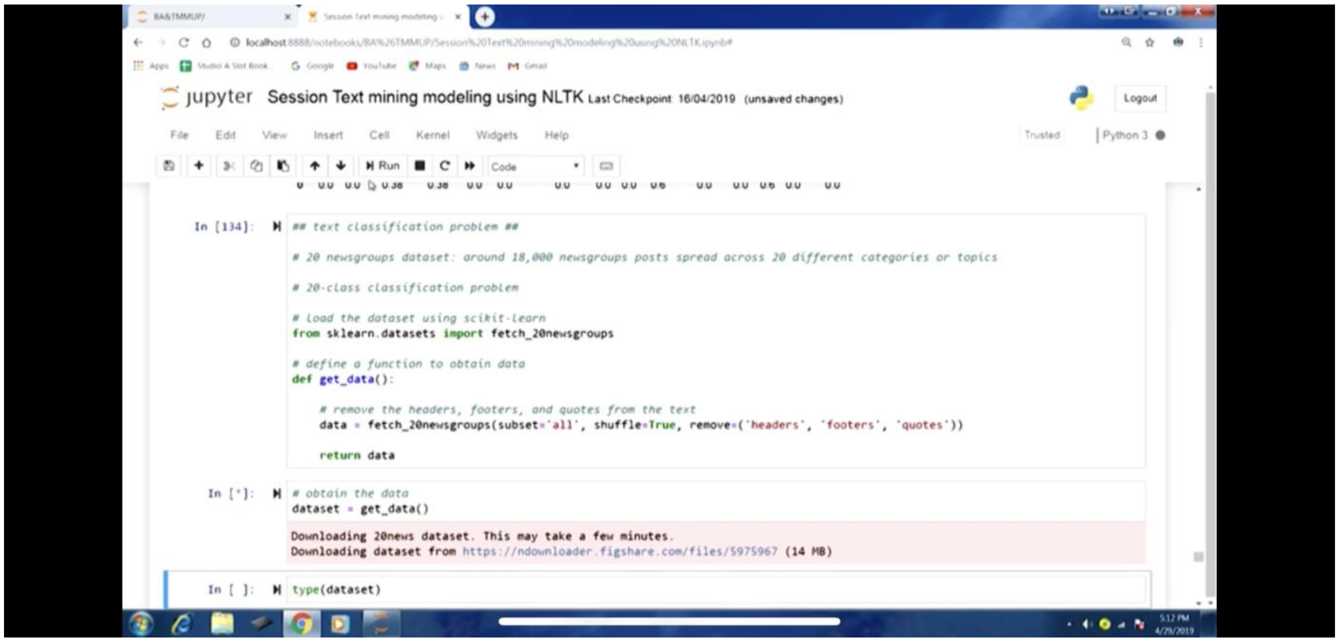
```
In [ ]:  # define a function to create a tf-idf feature extraction or vectorization model based on raw docs

         from sklearn.feature_extraction.text import TfidfVectorizer

         def tfidf_extractor(corpus, ngram_range=(1,1)):

             # using an instance of TfidfVectorizer class to define the type of features to be extracted
             vectorizer = TfidfVectorizer(min_df=1, norm='l2', smooth_idf=True, use_idf=True, ngram_range=ngram_range)

             # using fit_transform method to extarct the features from corpus (term-document matrix)
             features = vectorizer.fit_transform(corpus)
```

So, whenever we are dealing with larger data set and we just want to have a look at the data the kind of variables and other things that are there. So, loading a large data set will take a lot more time, so sometimes just to understand the data the kind of data that is there. It would be better if we are able to read a sample cases from a larger data set. So, how do we perform this, so let us talk about the so various ways we can various approaches we can take to execute this.

So, in the first approach that we are going to discuss will configure display setting, so we will have slightly compact display setting. But mean by this we would like to reduce the number of rows that are going to be displayed in our output here. So, this is the attribute here that we can use pd.options.display.max_rows. So, here we are specifying the maximum number of rows that can be displayed, so that is 10 and we can define it like this or any other number.

So, let me run this, now this is define, now will take an example of this data set ex6 which is a slightly larger file that we will see you know we will have a look at this safe attribute and we will find out how many rows and columns. So, I can you know first thing will load this will read this data into a data frame, so df1 reads csv and let us have a look at the shape of this.

(Refer Slide Time:29:05)



Session Text mining modeling using NLTK Last Checkpoint: 16/04/2019 (unsaved changes)

```
In [134]:  ## text classification problem ##

           # 20 newsgroups dataset: around 18,000 newsgroups posts spread across 20 different categories or topics

           # 20-class classification problem

           # load the dataset using scikit-learn
           from sklearn.datasets import fetch_20newsgroups

           # define a function to obtain data
           def get_data():

               # remove the headers, footers, and quotes from the text
               data = fetch_20newsgroups(subset='all', shuffle=True, remove=('headers', 'footers', 'quotes'))

               return data

In [*]:    # obtain the data
           dataset = get_data()

           Downloading 20news dataset. This may take a few minutes.
           Downloading dataset from https://ndownloader.figshare.com/files/5975967 (14 MB)

In [ ]:    type(dataset)
```

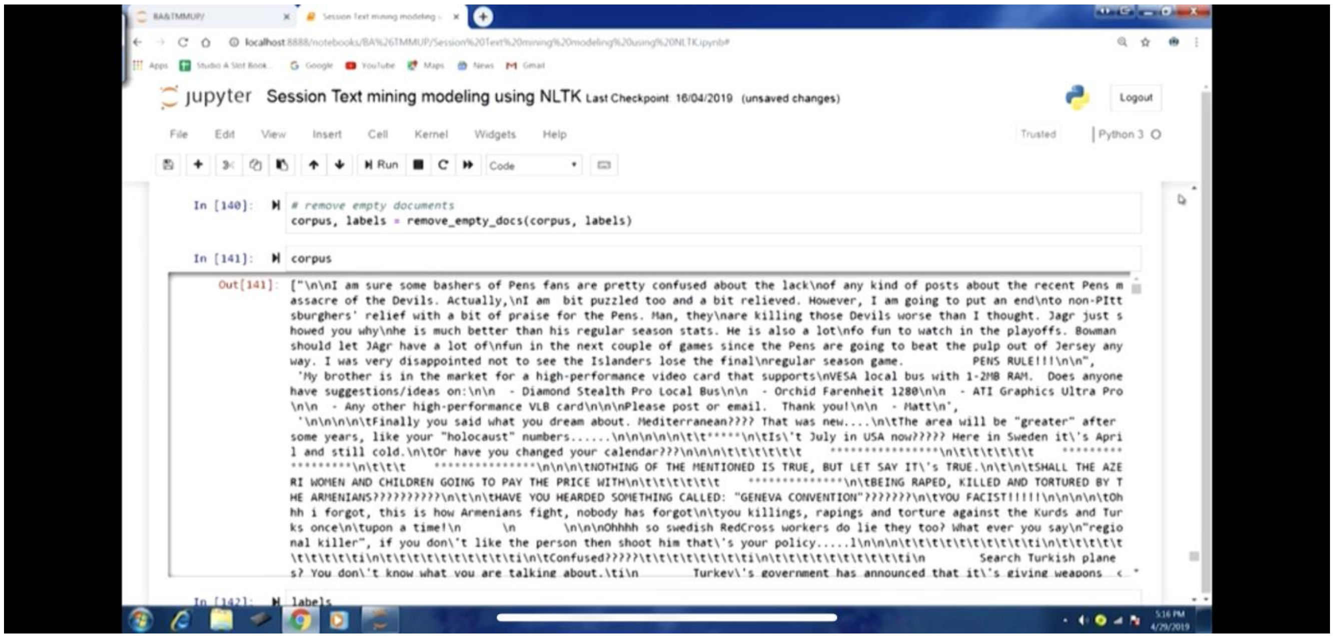
So, you can see that df1.7 output 16, 100005 so in the first dimension we have 10,000 elements that instant 1000 rows and 5 columns. So, this is 1000*5, so it is a large data set and we would prefer to have a look at a smaller sample of this and not load all the 10,000 you know observation which would be you know which will take time.

So, because we have configured the display parameters and the maximum number of rows that we can display is 10. So, if I try to print df1 here, if I run this you can see in the output 70 how many rows have been actually printed. So, you can see first few rows from the start of the you know dataset and last few rows from the you know end of the data sets, you can see index you can focus on the row indices here starting from 0, 1, 2, 3, 4 and then **..** and then those observation are not been displayed.

And the last you know 5 observation from the end of the data set 9995 and all those things. So, you can see 10,000 rows*5 columns that is the kind of dimension we are talking about and only 10 observations have been displayed in this fashion. So, this is quite useful when we are dealing with larger data set. Now we can achieve the same thing following a second approach, so in this case we can use n rows argument and we can specify the number of rows that we would like to display.

So, again while we are reading the data from ex6.csv into a data frame, we can specify n rows in this case let us say 5. So, if I run this particular code here, so you can see only 5 rows starting from 0 to 4 row indices those have been printed here. Now to there is another way but for slightly different context that can also be adopted. So, in this approach what we typically do is we use chunk size argument where we can specify a chunk in terms of number of rows.

(Refer Slide Time: 36:36)


jupyter
Session Text mining modeling using NLTK Last Checkpoint 16/04/2019 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

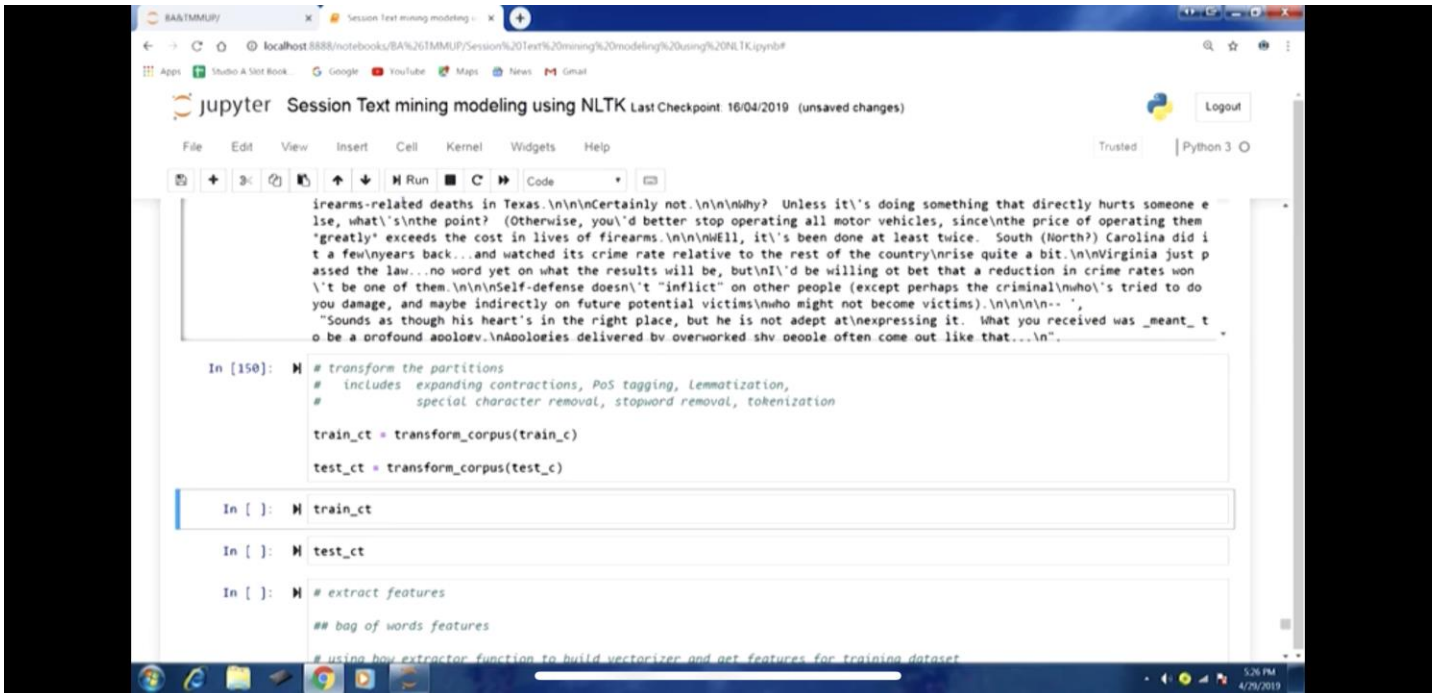
```
In [140]:  # remove empty documents
           corpus, labels = remove_empty_docs(corpus, labels)

In [141]:  corpus

Out[141]: ["\n\nI am sure some bashers of Pens fans are pretty confused about the lack\nof any kind of posts about the recent Pens m
          assacre of the Devils. Actually,\nI am  bit puzzled too and a bit relieved. However, I am going to put an end\nto non-PItt
          sburghers' relief with a bit of praise for the Pens. Man, they\nare killing those Devils worse than I thought. Jagr just s
          howed you why\nhe is much better than his regular season stats. He is also a lot\nfo fun to watch in the playoffs. Bowman
          should let JAgr have a lot of\nfun in the next couple of games since the Pens are going to beat the pulp out of Jersey any
          way. I was very disappointed not to see the Islanders lose the final\nregular season game.          PENS RULE!!!\n\n",
           'My brother is in the market for a high-performance video card that supports\nVESA local bus with 1-2MB RAM.  Does anyone
          have suggestions/ideas on:\n\n  - Diamond Stealth Pro Local Bus\n\n  - Orchid Farenheit 1280\n\n  - ATI Graphics Ultra Pro
          \n\n  - Any other high-performance VLB card\n\n\nPlease post or email.  Thank you!\n\n  - Matt\n',
           '\n\n\n\n\tFinally you said what you dream about. Mediterranean???? That was new....\n\tThe area will be "greater" after
          some years, like your "holocaust" numbers......\n\n\n\n\n\t\t*****\n\tIs\'t July in USA now????? Here in Sweden it\'s Apri
          l and still cold.\n\tOr have you changed your calendar???\n\n\n\t\t\t\t\t\t    ****************\n\t\t\t\t\t\t    *********
          *********\n\t\t\t    ***************\n\n\n\tNOTHING OF THE MENTIONED IS TRUE, BUT LET SAY IT\'s TRUE.\n\t\n\tSHALL THE AZE
          RI WOMEN AND CHILDREN GOING TO PAY THE PRICE WITH\n\t\t\t\t\t\t    **************\n\tBEING RAPED, KILLED AND TORTURED BY T
          HE ARMENIANS??????????\n\t\n\tHAVE YOU HEARDED SOMETHING CALLED: "GENEVA CONVENTION"???????\n\tYOU FACIST!!!!!\n\n\n\n\tOh
          hh i forgot, this is how Armenians fight, nobody has forgot\n\tyou killings, rapings and torture against the Kurds and Tur
          ks once\n\tupon a time!\n       \n        \n\n\nOhhhh so swedish RedCross workers do lie they too? What ever you say\n"regio
          nal killer", if you don\'t like the person then shoot him that\'s your policy.....l\n\n\n\t\t\t\t\t\t\t\t\t\t\ti\n\t\t\t\t\t
          \t\t\t\t\ti\n\t\t\t\t\t\t\t\t\t\ti\n\tConfused?????\t\t\t\t\t\t\t\ti\n\t\t\t\t\t\t\t\t\t\t\ti\n         Search Turkish plane
          s? You don\'t know what you are talking about.\ti\n        Turkey\'s government has announced that it\'s giving weapons

In [142]:  labels
```

So, a number of rows which we would like to sample actually, so that we can specify using chunk size. So, in this case a different object is written not data frame rather a text parser object is written. So, if we call this function pdand.read_csv and the file name ex6.csv which is the larger dataset and specify the chunk size let us say 500. So, we would like to sample 500 you know observation from the you know full data set.

Then this chunk is going to be returned as a tax parser object, so if I run this we will get this tp if you want to have a look at this what is this tp about. So, you can see in the output 20, you can see the type of this per particular python object there. Now we have a function for this kind of object get_chunk, so that will allow us to display the rows from this particular object.

So, we can call tp.get_chunk and if I run this you can see we are just displaying 10 observations here because the display settings if you remember we have already configured, so that is 10. So, from this chunk of 500 we have got these 10 observation, to confirm that you can look observe the row indices here starting from 0, 1, 2, 3, 4 and then we have 495, 496, 497 like that. So, you can see that the you know first 500 you know observations from the full data set have been taken as a chunk.

(Refer Slide Time: 40:26)


jupyter Session Text mining modeling using NLTK Last Checkpoint: 16/04/2019 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
irearms-related deaths in Texas.\n\n\nCertainly not.\n\n\nWhy?  Unless it\'s doing something that directly hurts someone e
lse, what\'s\nthe point?  (Otherwise, you\'d better stop operating all motor vehicles, since\nthe price of operating them
*greatly* exceeds the cost in lives of firearms.\n\n\nWEll, it\'s been done at least twice.  South (North?) Carolina did i
t a few\nyears back...and watched its crime rate relative to the rest of the country\nrise quite a bit.\n\nVirginia just p
assed the law...no word yet on what the results will be, but\nI\'d be willing ot bet that a reduction in crime rates won
\'t be one of them.\n\n\nSelf-defense doesn\'t "inflict" on other people (except perhaps the criminal\nwho\'s tried to do
you damage, and maybe indirectly on future potential victims\nwho might not become victims).\n\n\n\n-- ',
 "Sounds as though his heart's in the right place, but he is not adept at\nexpressing it.  What you received was _meant_ t
o be a profound apology.\nApologies delivered by overworked shy people often come out like that...\n".
```

```
In [150]:  # transform the partitions
           #   includes  expanding contractions, PoS tagging, lemmatization,
           #             special character removal, stopword removal, tokenization

           train_ct = transform_corpus(train_c)

           test_ct = transform_corpus(test_c)

In [ ]:    train_ct

In [ ]:    test_ct

In [ ]:    # extract features

           ## bag of words features

           # using bow extractor function to build vectorizer and get features for training dataset
```

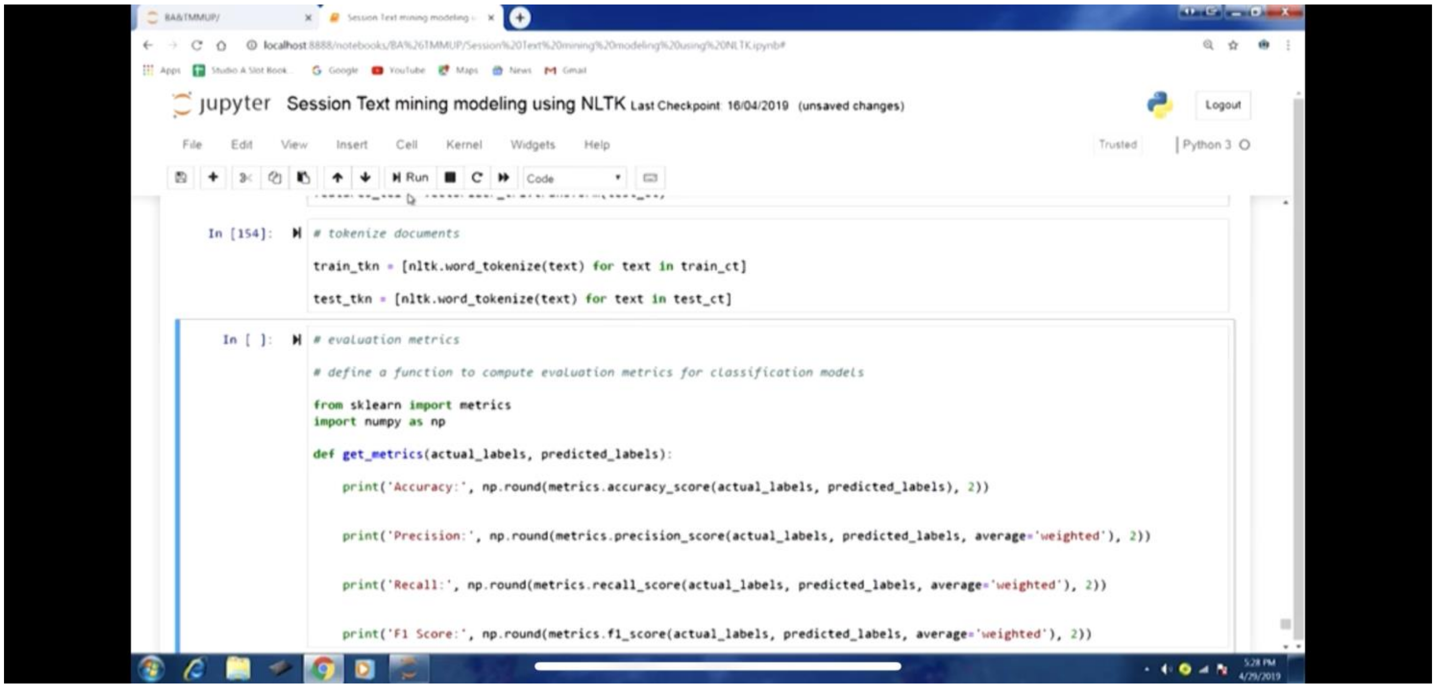
And from that because we had configure the display option for up to maximum 10 number of rows, so therefore 10 rows in this particular fashion have been displayed. Let us move forward, so till now we talked about writing till now we talked about reading data from csv file into a data frame. Now let us talk about writing a data frame into a csv file, so sometimes certain analysis certain output we would like to write into a csv file.

And so we can use 2_csv method to perform this, so if we have data frame let us say df and we can call this .2_csv function. And we can pass on the new file name that is let us say write1.csv. So, whatever data is there in this data frame df, that is going to be written in this particular you know new file. So, if I run this and because this is now written, so you know we can have a look at the you know contents of this file.

So, we can use this magic command %pycat here, so if I run this we will have a pop-up window and you can see this new file has been created and the data as well. Now let us move forward, now there could be other delimiters, so one is csv which is essentially comma separated for the comma separated kind of data. Now certain you know text files might have data **you** and they might be using other delimiters for example to tab, spaces.

So, how they can be handled here, so now we will use sys.standard output that means in the output here itself you would like to display and for this we will like to import this is module here. And the same function df.2_csv and you can see we are writing to sys.stdout which is also a file but a system file. So, that is the file which is responsible for printing the output in the standard you know in the console.

(Refer Slide Time: 40:56)


jupyter Session Text mining modeling using NLTK Last Checkpoint 16/04/2019 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
In [154]:  # tokenize documents

           train_tkn = [nltk.word_tokenize(text) for text in train_ct]

           test_tkn = [nltk.word_tokenize(text) for text in test_ct]

In [ ]:    # evaluation metrics

           # define a function to compute evaluation metrics for classification models

           from sklearn import metrics
           import numpy as np

           def get_metrics(actual_labels, predicted_labels):

               print('Accuracy:', np.round(metrics.accuracy_score(actual_labels, predicted_labels), 2))


               print('Precision:', np.round(metrics.precision_score(actual_labels, predicted_labels, average='weighted'), 2))


               print('Recall:', np.round(metrics.recall_score(actual_labels, predicted_labels, average='weighted'), 2))


               print('F1 Score:', np.round(metrics.f1_score(actual_labels, predicted_labels, average='weighted'), 2))
```

So, in this case it will be printed here itself and you can see here we are specifying the different separator. So, a different separator is there and so whatever you know we are going to write you know we will take data from df and will be writing that into the sys.standardout in the console output and different separator is going to be used. So, if I run this you can see in the output you know instead of comma we have different you know separator.

Now sometimes in the data we might have empty string as a sentinel you know, so that would typically indicate for missing values, sentinels that would be difficult to detect you know typically in a if you are using a simple text format. So, how do we change this, so again let us take the example of this ex5, so let me load this in a data frame df2.

So, here you can see that we have any nans and if I write this df2 into console output here, you can use 2_csv function here also. So, if I run this and if you look at the you know output that you know some of the nans for example if you look at the second row here you can see up after 4 and nan is gone that is replaced with the empty string.

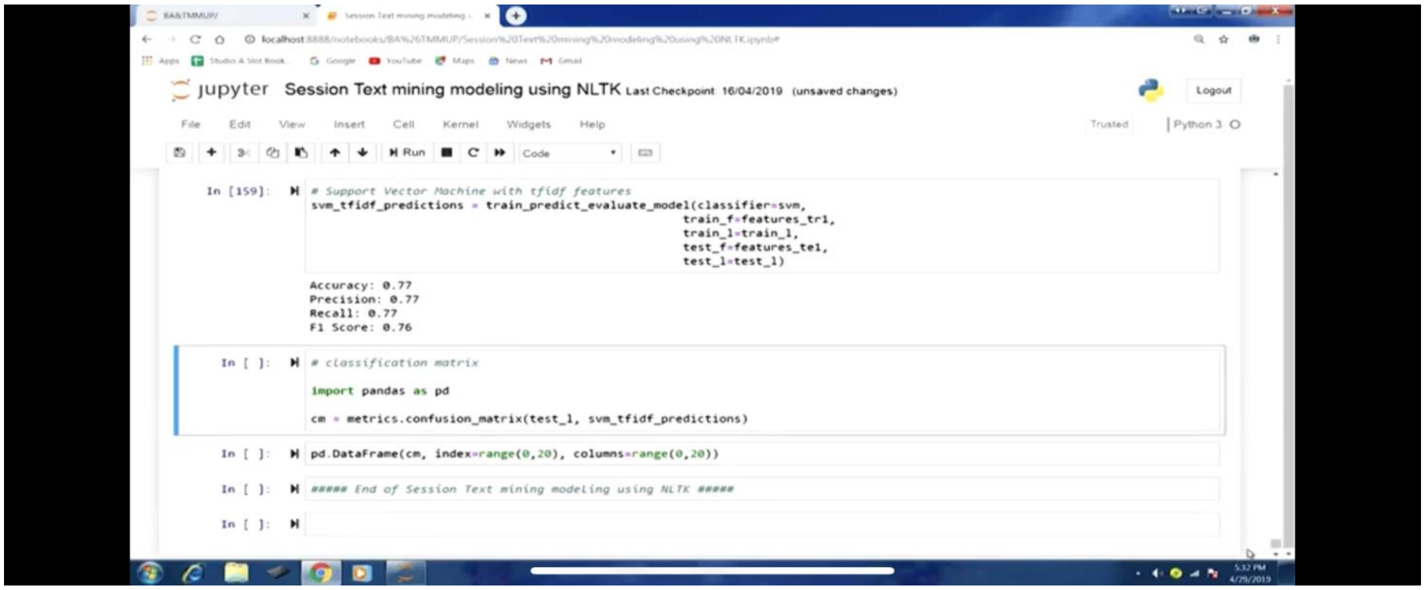
So, how do we overcome this, so for that we can use another argument na_rep, so it will replace the sentinel marker for na, so we can make it null. So, in the next line if you see that we can again call this function df2.2_csv and against sys.stdout and second argument na_rep we are replacing this you know sentinel marker here. If I run this now you can see after 4 in the second row we have got null to indicate the missing value.

Now let us move forward, now we will focus on excel files which are also typically use to store data sets. So, how do we work with excel file in python environment, so let us talk about reading an excel file into a data frame and because in an excel file we might have multiple sheet in csv file we have just 1 sheet. Wherein excel file we might have you know multiple sheets, so how do we read multiple sheets multiple sheet that could be present in a file.

So, for this we can use excel file function slightly faster way you know if we are reading multiple you know sheets from an excel file. So, using this function will get an excel file kind of object and that we could use to actually import the data from that sheet into a data frame. So, let me take example of an excel file in this case ex1.xlsx so I am using this excel file function let me run this code and this object is created, now let us have a look at this object, so you can see here that this is an excel file object.

Now we would be using this ef you know this particular object in read_excel function, so read_excel function can be used to read the data you know into a data frame. So, you can take an example you can have a look at this example pd.read_excel then ef this excel file object and c21. So, we are specifying in the second argument the sheet that we would like to you know read from the data set into a data frame.

(Refer Slide Time:38:26)



jupyter Session Text mining modeling using NLTK Last Checkpoint: 16/04/2019 (unsaved changes)

```
In [159]:  # Support Vector Machine with tfidf features
           svm_tfidf_predictions = train_predict_evaluate_model(classifier=svm,
                                                                train_f=features_tr1,
                                                                train_l=train_l,
                                                                test_f=features_te1,
                                                                test_l=test_l)

           Accuracy: 0.77
           Precision: 0.77
           Recall: 0.77
           F1 Score: 0.76

In [ ]:    # classification matrix

           import pandas as pd

           cm = metrics.confusion_matrix(test_l, svm_tfidf_predictions)

In [ ]:    pd.DataFrame(cm, index=range(0,20), columns=range(0,20))

In [ ]:    ##### End of Session Text mining modeling using NLTK #####

In [ ]:
```

If we have multiple sheet we can you know call this function many times and different data frames can be created. So, in this fashion we are dealing with multiple sheets this excel file object can really be helpful to you know read data from the file and from different sheets into different data frames. So, if I run this and again you can have a look at the output here output number 33 and this is the data frame that we have got.

However sometimes we would just be having 1 sheet in the data set and we would be all are you know we would just be focusing on 1 sheet we might not require to import the other sheets into the python environment. So, in those cases we can directly use the read_excel function where in the first argument instead of the excel file object we can specify the file path.

So, in this case x1.xlsx directly and the first sheet, sheet1 and we would be able to read the data into a data frame and again you can see. So, 2 ways depending on the scenario we are dealing with and you know we would have an easy way to import and read data into python environment. Now let us talk about writing a data frame into an excel file, so here again we have excel writer function.

So, again this is also suitable for writing you know multiple sheets whenever we are having multiple sheet, we can use the excel writer object. And for different sheets we would be able to write you know in the file. So, let us say let us let us call this excel writer for in you know this for writing this cwrite2.xlsx. And if I run this and you can have a look at the this object excel writer object and we can use this function to_excel function to actually write the data frame into the excel sheet.

So, here you can see we are calling we are using df3 data frame which would be written into this excel file and we the excel file object and sheet 1 where this data frame is going to be written. So, if we have multiple data frames to be written we can use those data frame and keep on changing the sheets and different sheets we can write as using this excel write object.

So, if I run this and now we can use the save method to actually you know save this file, so ew.save and we will have that file saved there in the disk. Now if you want to avoid excel writer or because we are just interested in 1 sheet so we can pass the file path directly to the you know to_excel function and we can write the data frame into that file. So, we can also say df3.to_excel on the file name write 3.xlsx, in this case again we would be able to write that file.

Let us move forward, now next thing is so till now we will we talked about you know working with csv files and working with excel files. And now next thing would be working with web APIs, **now** nowadays there are many websites which provide public APIs to provide data fields you know via JSON or other format for analytics purposes, for research purposes and many other fare you know fare purposes for that particular data.

So, to perform you know there are many packages that are available to achieve this to work with web APIs. So, one easy to use package is request package, so we will use that to **to** work with to demonstrate web API example here. So, first thing import this package in import request, so we will be importing this. Then let us take an example of github website, so here we are taking you know further we are taking last 30 github issues for pandas posted on github.

So, this is the URL for all these issues here, so we will be using this perk particular URL and first thing is to get a response object. Because as you could see that we talked about excel writer object, excel file object because whenever we are required to deal with a particular file, a particular object again and again**.** So, these kind of classes are really useful because we can you know a use object again and again as and when required.

So, in this case you can see we are calling request.get and URL, so we will get a response object for this URL and you can look at the type of the object here as well. Now we can use JSON method, so that is to obtain a list of so when we apply this method what we will get in the return is that list of dicts having all the data on a github issue page, so except for the comments.

So, except for the comments all the data that is going to be there this particular you know method is going to written that. So, let me store that in v1, so v1=rest.json and if I run this and have a look at the v1, so as I said v1 is a list of dict. So, you can see a list and then the list is starting and different columns and the associated value part, so key value pairs in the dict. And many dicts are there, so you can scroll down and see for yourself.

Now next thing that we are required to do is that we can create a data frame with fields of interest. That means you know whichever fields are require for our analytics purposes we can pick those fields and there will be columns and we can create you know a data frame. So, we can use this pd.data frame here and v1 is the data that we are passing the first argument and then we can specify in the columns you know argument in the fields that where we are interested in.

So, if I run this we will have data in df4, so if I run this you can see in the output 48 you can see number, title, labels, state. So, these are the columns and we have got the data as you can see the row index, so this is a 30*4 column. So, if you remember we wanted you know just 30 rows there, so now here other values you can see number, the title, title means the you know pandas issues you know and the title that has been given by the visitors.

And other labels and the store state whether that particular issue is still open or closed, so those details you can see in this particular data frame. So, this is how we can actually deal with you know we can use web APIs to download data which could be useful for analytics purposes.
(Video Ends: 30:56)

So, we will stop here and in the next lecture we will start our discussion on certain other aspects you know of you know data management fare you know dropping you know na values or replacing them. Certain some of those scenarios and many other things we would be discussing in the next lecture, thank you.

Keywords: Evaluation metrics, Numerical python, text mining modelling, arguments, exception handling etc.