Hinkle_LexNLP_Example_Code

June 6, 2022

[1]:
```
### How to Extract Legal Citations using Python (for the complete beginner)
### Law and Courts Newsletter
### Rachael K. Hinkle
### June 6, 2022
### Run on macOS 10.15.7

### The only difference in this code for Windows is the way filepaths are set up.
 ↪ They use a different file seperator.
## Macs use a single foward slash; if you are using a mac run the follwing code:
fileSeperator = "/"
## Windows uses a double back slash: if you are using Windows use the following␣
 ↪code,
# Note, to do this delete the "#" at the beginning of the next line.
#fileSeperator = "\\"
```

[2]:
```
# The following code turns off the autosave feature to avoid inadvertently␣
 ↪saving over a previous file.
# As always, remember to save your work as you go along.
%autosave 0
```

```
Autosave disabled
```

[3]:
```
## Preliminary Step 1: It helps to verify we are running Python 3.6 before␣
 ↪proceeding

# Python tip: Any text preceeded by a "#" is ignored by Python.

import sys
print(sys.version)
```

```
3.6.13 |Anaconda, Inc.| (default, Feb 23 2021, 12:58:59)
[GCC Clang 10.0.0 ]
```

[4]:
```
## Preliminary Step 2: Install the lexNLP package (only need to do this the␣
 ↪first time)
# Note: This step will take a bit of time.  When an asterisk appears in the␣
 ↪square brackets
```

```
# to the left of a cell (e.g. "In [*]"), that means the code is running.  A␣
 ↪number will
# appear in the square brackets when the code is done running.
```

[5]:
```
pip install lexnlp
```

```
Requirement already satisfied: lexnlp in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (1.8.0)
Requirement already satisfied: num2words==0.5.10 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(0.5.10)
Requirement already satisfied: datefinder-lexpredict==0.6.2.1 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(0.6.2.1)
Requirement already satisfied: dateparser==0.7.2 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(0.7.2)
Requirement already satisfied: regex==2020.7.14 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(2020.7.14)
Requirement already satisfied: requests==2.24.0 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(2.24.0)
Collecting numpy==1.19.1
  Using cached numpy-1.19.1-cp36-cp36m-macosx_10_9_x86_64.whl (15.3 MB)
Requirement already satisfied: pycountry==20.7.3 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(20.7.3)
Requirement already satisfied: Unidecode==1.1.1 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(1.1.1)
Requirement already satisfied: nltk==3.5 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(3.5)
Requirement already satisfied: reporters-db==2.0.3 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(2.0.3)
Requirement already satisfied: gensim==3.8.3 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(3.8.3)
Collecting scipy==1.5.1
  Using cached scipy-1.5.1-cp36-cp36m-macosx_10_9_x86_64.whl (28.8 MB)
Requirement already satisfied: joblib==0.14.0 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(0.14.0)
Requirement already satisfied: us==2.0.2 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
```

```
(2.0.2)
Requirement already satisfied: scikit-learn==0.23.1 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from lexnlp)
(0.23.1)
Collecting pandas==0.24.2
  Using cached pandas-0.24.2-cp36-cp36m-macosx_10_6_intel.macosx_10_9_intel.maco
sx_10_9_x86_64.macosx_10_10_intel.macosx_10_10_x86_64.whl (16.3 MB)
Requirement already satisfied: pytz in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from datefinder-
lexpredict==0.6.2.1->lexnlp) (2021.3)
Requirement already satisfied: python-dateutil>=2.4.2 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from datefinder-
lexpredict==0.6.2.1->lexnlp) (2.8.2)
Requirement already satisfied: tzlocal in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
dateparser==0.7.2->lexnlp) (4.1)
Requirement already satisfied: smart-open>=1.8.1 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
gensim==3.8.3->lexnlp) (5.2.1)
Requirement already satisfied: six>=1.5.0 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
gensim==3.8.3->lexnlp) (1.16.0)
Requirement already satisfied: tqdm in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
nltk==3.5->lexnlp) (4.62.3)
Requirement already satisfied: click in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
nltk==3.5->lexnlp) (8.0.3)
Requirement already satisfied: docopt>=0.6.2 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
num2words==0.5.10->lexnlp) (0.6.2)
Requirement already satisfied: idna<3,>=2.5 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
requests==2.24.0->lexnlp) (2.10)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
requests==2.24.0->lexnlp) (1.25.11)
Requirement already satisfied: chardet<4,>=3.0.2 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
requests==2.24.0->lexnlp) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
requests==2.24.0->lexnlp) (2021.5.30)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from scikit-
learn==0.23.1->lexnlp) (3.1.0)
Requirement already satisfied: jellyfish==0.6.1 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
```

```
us==2.0.2->lexnlp) (0.6.1)
Requirement already satisfied: importlib-metadata in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
click->nltk==3.5->lexnlp) (4.8.3)
Requirement already satisfied: typing-extensions>=3.6.4 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from importlib-
metadata->click->nltk==3.5->lexnlp) (4.1.1)
Requirement already satisfied: zipp>=0.5 in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from importlib-
metadata->click->nltk==3.5->lexnlp) (3.6.0)
Requirement already satisfied: pytz-deprecation-shim in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
tzlocal->dateparser==0.7.2->lexnlp) (0.1.0.post0)
Requirement already satisfied: backports.zoneinfo in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
tzlocal->dateparser==0.7.2->lexnlp) (0.2.1)
Requirement already satisfied: importlib-resources in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from
backports.zoneinfo->tzlocal->dateparser==0.7.2->lexnlp) (5.4.0)
Requirement already satisfied: tzdata in
/opt/anaconda3/envs/Python3.6Test/lib/python3.6/site-packages (from pytz-
deprecation-shim->tzlocal->dateparser==0.7.2->lexnlp) (2021.5)
Installing collected packages: numpy, scipy, pandas
  Attempting uninstall: numpy
    Found existing installation: numpy 1.19.2
    Uninstalling numpy-1.19.2:
      Successfully uninstalled numpy-1.19.2
  Attempting uninstall: scipy
    Found existing installation: scipy 1.5.2
    Uninstalling scipy-1.5.2:
      Successfully uninstalled scipy-1.5.2
  Attempting uninstall: pandas
    Found existing installation: pandas 1.1.5
    Uninstalling pandas-1.1.5:
      Successfully uninstalled pandas-1.1.5
Successfully installed numpy-1.19.1 pandas-0.24.2 scipy-1.5.1
Note: you may need to restart the kernel to use updated packages.
```

[ ]:

[6]:
```
## Preliminary Step 3: Import the libraries you will need for this session␣
 ↪(analogous to R packages)

# 'os' provides useful funtions for dealing with the operating system
# 're' provides the ability to use regular expressions
# 'csv' provides the ability to write the results to a .csv file
# 'string' provides useful functions for processing the opinion content
```

```
# 'lexnlp' provides the ability to extract citations from natural language

import os, re, csv, string, lexnlp
import lexnlp.extract.en.citations
```

[ ]:

[7]:
```
## Preliminary Step 4: See the name of your working directory
os.getcwd()
```

[7]:
```
'/Users/rachaelhinkle/Dropbox/Law_and_Courts_Newsletter_2022/Hinkle_L&C_Newslett
er_Dataverse_Files2'
```

[8]:
```
## Preliminary Step 5: Setup a shortcut reference to your working directory

# Python tip: This just creates a variable with a string.
#             You can name the variable anything you want.

mydir = os.getcwd() + fileSeperator
```

[ ]:

[9]:
```
### Illustration 1: How to extract citations from a single file
## Note: If you are familiar with Python, you can proceed to Illustration 2
```

[10]:
```
## Step 1.1: Read in an opinion (or other legal document) in a .txt file
# This code creates a list; each entry of the list is a string that contains the␣
 ↪text
# from one line in the file

# Python tip: The "print()" function is very useful for seeing the contents of␣
 ↪variable
#             to see or double check what the code is doing

# Note: If you get an error, check that the "mydir" variable correctly lists␣
 ↪your directory
# where the file is located, and/or make sure you have downloaded the "myExample.
 ↪txt" file from Dataverse.

tfile = open(mydir + "myExample.txt", "r", encoding="latin-1")
lines = tfile.readlines()
print(lines)
```

```
['\n', '\n', '             UNITED STATES OF AMERICA, Plaintiff-Appellee, v.
KORVELL\n', '                      DENNIS PITTMAN, Defendant-Appellant.\n',
'\n', '                                 No. 04-2546\n', '\n', '
UNITED STATES COURT OF APPEALS FOR THE SEVENTH CIRCUIT\n', '\n', '
```

```
411 F.3d 813; 2005 U.S. App. LEXIS 11067\n', '\n', '
April 5, 2005, Argued\n', '                                   June 13, 2005,
Decided\n', '\n', '\n', 'COUNSEL: For UNITED STATES OF AMERICA, Plaintiff-
Appellee: John K. Mehochko,\n', 'OFFICE OF THE UNITED STATES ATTORNEY, Rock
Island, IL USA.\n', '\n', 'For KORVELL D. PITTMAN, Defendant-Appellant: George
F. Taseff, OFFICE OF THE\n', 'FEDERAL PUBLIC DEFENDER, Peoria, IL USA.\n', '\n',
'OPINION BY: POSNER\n', '\n', 'OPINION\n', '\n', '     [*814]  POSNER, Circuit
Judge. The defendant pleaded guilty to being a felon\n', 'in possession of a
firearm and was sentenced under the federal sentencing\n', "guidelines (before
the Supreme Court's Booker decision) to 188 months in prison,\n", 'the bottom of
the applicable guideline range but only eight months above the\n', 'statutory
minimum. [HN1] The Armed Career Criminal Act, 18 U.S.C.  924(e),\n', 'imposes a
180-month minimum sentence on anyone  [*815]  who has at least three\n', "prior
convictions of specified offenses. The judge found that the defendant's\n",
'criminal record qualified under this provision, and [HN2] findings of prior\n',
"convictions are not within the scope of Booker's rule. United States v.
Booker,\n", '160 L. Ed. 2d 621, 125 S. Ct. 738, 756 (2005); United States v.
Ngo, 406 F.3d\n', '839, 2005 WL 1023034, at *2-4 (7th Cir. 2005)  [**2]  .\n',
'\n']
```

[ ]:

[11]:
```
## Step 1.2: Process the list from the previous step to create a single string␣
 ↪variable with
#           only the majority opinion content

#create an empty string variable (that we will later fill with the majority␣
 ↪opinion)
opinText = ""

#create a logical operator variable (think of this as a switch set to "off")
opin = False

# loop through each line in the document in turn, after each line it goes to the␣
 ↪next,
# when there are no longer any lines left, the loop will stop
for line in lines:
    # Check to see if the line starts with some or no white space followed by␣
 ↪"OPINION" and then the end of the line
    # If so, turn the switch "on" (by making the logical operator variable equal␣
 ↪True)
    # Once this switch is turned on, it will remain that way until we turn it off
    if re.search("^[\s]*OPINION$", line):
        opin = True

    # Check for various strings that would indicate the end of the majority␣
 ↪opinion
```

```
    # If any of these patters of text are found, turn the switch "off" by making␣
 ↪the logical operator variable equal False
    if line and (re.search("^[\s]*DISSENT", line)):
        opin = False
    if line and (re.search("^[\s]*CONCUR", line)):
        opin = False
    if line and (re.search("^[^A-Za-z]*APPENDI", line)):
        opin = False

    # For any line of text that occurs while the switch is "on", that is not␣
 ↪empty, add it
    # to the string variable where we are collecting the majority opinion text.
    if opin and not re.search("^[\s]*$", line):
        opinText = opinText + line

# Now that the loop is complete, and we have the entire opinion, we can process it
# This code finds any instances of one of more spaces and substitutes a single␣
 ↪space
opinText = re.sub("\s+", " ", opinText)

#Finally, we want to see our final product
print(opinText)

# Python tip: some elements of regular expressions used in the code above:
# ^: beginning of the line
# $: end of the line
# [\s]*: zero or more white spaces
# [\s]+: one or more white spaces
# [^A-Za-z]: Anything that is NOT a capital or lower case letter.
#  (Inside the square brackets "^" means NOT, outside the brackets it means␣
 ↪beginning of the line)
# [A-Z] means any capital letter
# [a-z] means any lower case letter
# [A-z] means any letter
# [0-9] means any numeral
```

```
OPINION [*814] POSNER, Circuit Judge. The defendant pleaded guilty to being a
felon in possession of a firearm and was sentenced under the federal sentencing
guidelines (before the Supreme Court's Booker decision) to 188 months in prison,
the bottom of the applicable guideline range but only eight months above the
statutory minimum. [HN1] The Armed Career Criminal Act, 18 U.S.C. 924(e),
imposes a 180-month minimum sentence on anyone [*815] who has at least three
prior convictions of specified offenses. The judge found that the defendant's
criminal record qualified under this provision, and [HN2] findings of prior
convictions are not within the scope of Booker's rule. United States v. Booker,
160 L. Ed. 2d 621, 125 S. Ct. 738, 756 (2005); United States v. Ngo, 406 F.3d
839, 2005 WL 1023034, at *2-4 (7th Cir. 2005) [**2] .
```

[ ]:

[12]:
```
## Step 1.3: Extract citations from the opinion

#Extract each ciation and put it in a variable
cite_holder = lexnlp.extract.en.citations.get_citations(opinText)

# See the citations extracted by using a loop
# Note: It doesn't work to just print the variable, but you can print (or␣
 ↪otherwise access)
#       each item in it by using a loop.
for cite in cite_holder:
    print(cite)

# From the output we can see that each citation has multiple pieces of␣
 ↪information
# Illustration 2 will show how to get key information into a spreadsheet for␣
 ↪further analysis
```

```
(160, 'L. Ed. 2d', "Lawyer's Edition", 621, '125', None, None)
(406, 'F.3d', 'Federal Reporter', 839, '2005', None, None)
```

[ ]:

[13]:
```
### Illustration 2: How to extract citations from all files in a subfolder
###                 and output the data to a spreadsheet with one row per citation
```

[14]:
```
## Step 2.1: Create and open a .csv file that we will write information into
fout = open(mydir + "myNewSpreadsheet.csv", "w", newline="")
outfilehandle = csv.writer(fout,
                           delimiter=",",
                           quotechar='"',
                           quoting=csv.QUOTE_NONNUMERIC)

# Create a row that contains the variables names and write that to the first row␣
 ↪of the .csv file
localrow = []
localrow.append("filename")
localrow.append("volume")
localrow.append("reporter")
localrow.append("page1")
localrow.append("page2")
localrow.append("court")
localrow.append("year")
localrow.append("raw citation")
#localrow.append("numCites")
#localrow.append("numIds")
```

```
outfilehandle.writerow(localrow)

print(localrow)
```

```
['filename', 'volume', 'reporter', 'page1', 'page2', 'court', 'year', 'raw
citation']
```

[15]:
```
## Step 2.2: Prepare information for where to read text files from

#create a filepath to the folder where all text files are located
dirname = mydir + "sampleOpinions" + fileSeperator
# Create a list of all files in a given folder
dirlist = os.listdir(dirname)
# See the list of all files to make sure things are working
print(dirlist)
```

```
['13638.842.F.2d.509.txt', '14374.967.F.2d.73.txt', '16894.436.F.3d.174.txt',
'2761.831.F.2d.538.txt', '7434.544.F.3d.449.txt']
```

[16]:
```
## Step 2.3: Loop through all documents, process text, extract ciations, process␣
 ↪citations,
##           and export to spreadsheet

for entry in dirlist: ## each entry is a separate txt file with a single opinion
    infilehandle = open(dirname + entry, encoding='latin-1')
    txtlines = infilehandle.readlines()

    # This is useful for watching the progress of the code, espcially with many␣
 ↪documents
    print(entry)

    # Create empty variables (each time through the loop it is important to make␣
 ↪sure
    # all variables are empty so values from the previous document do not carry␣
 ↪over)
    opin_string = ""
    #opin_string2 = ""
    #opin_string3 = ""
    #opin_string4 = ""
    sent_holder = []
    cite_holder = []
    localrow = []
    sct = 0
    op_line = False

    # Loop through each line in the document
    for txtline in txtlines:
```

```
        # Find the beginning and end of the opinion
        if re.search("^[^A-Za-z]*OPINION[\s]*$", txtline):
            op_line = True
        if line and (re.search("^[\s]*DISSENT", line)):
            opin = False
        if line and (re.search("^[\s]*CONCUR", line)):
            opin = False
        if line and (re.search("^[^A-Za-z]*APPENDI", line)):
            opin = False

        # Create a string with the majority opinion and process the text
        if op_line:
            opin_string = opin_string + txtline

    ## After looping through all lines in the document, do the following steps␣
↪once per document

    ## Process text in opinion string

    # Get rid of newline characters
    opin_string = re.sub("\n", " ", opin_string)
    # Replace short citations with the placeholder "999 U.S. 999" that will be␣
↪recognized by lexnlp
    opin_string = re.sub("[\s](I|i)d\.[\s]", " 999 U.S. 999 ", opin_string)
    opin_string = re.sub("[\s](I|i)bid\.[\s]", " 999 U.S. 999 ", opin_string)

    # Extract citations
    cite_holder = lexnlp.extract.en.citations.get_citations(opin_string)

    # Change "cite_holder" to a list object
    cite_holder = list(cite_holder)

    # Create a second variable that replaces short references with the full␣
↪citation

    #Pyton tip: Python indexes begin with zero.  So to extract the first element␣
↪of the
    #           variable "cite_holder", we can use "cite_holder[0]"

    cite_holder2 = []
    # Since we need to refer back to the previous element of the loop for this␣
↪task,
    # we use a different kind of loop that uses an index, "i"
    for i in range(0, len(cite_holder)):
        if cite_holder[i][0] == 999: #and cite[1] == "U\.S\." and cite[3] == 999:
            cite_holder2.append(cite_holder2[i-1])
```

```
        else:
            cite_holder2.append(cite_holder[i])

    # Create empty variables that will be used for exporting information to␣
↪spreadsheet
    vol = ""
    rep = ""
    page1 = ""
    page2 = ""
    court = ""
    year = ""

    # Put content in each variable that will be written to the spreadsheet
    for cite in cite_holder2:
        vol =str(cite[0])
        rep = str(cite[1])
        page1 = str(cite[3])
        page2 = str(cite[4])
        court = str(cite[5])
        year = str(cite[6])

        #The lexnlp function does not perform perfectly, here is some code to␣
↪clean up
        # some of the issues by getting rid of info that is not correct
        # Note: the "raw citation" column still has the full information for␣
↪each cite
        if len(court) > 20:
            court = ""
        if court == "None":
            court = ""
        if page2 == "None":
            page2 = ""
        if year == "None":
            year = ""
        if re.search("U\.S\.", rep):
            court = "SCOTUS"
        if re.search("L\.[\s]*Ed\.", rep):
            court = "SCOTUS"
        if re.search("S\.[\s]*Ct\.", rep):
            court = "SCOTUS"

        # Write a row to the spreadsheet for each ciation
        if vol != "":
            localrow = []
            localrow.append(entry)
            localrow.append(vol)
            localrow.append(rep)
```

```
            localrow.append(page1)
            localrow.append(page2)
            localrow.append(court)
            localrow.append(year)
            localrow.append(cite)
            outfilehandle.writerow(localrow)

# Close the files you are reading from and to
infilehandle.close()
fout.close()
```

```
13638.842.F.2d.509.txt
14374.967.F.2d.73.txt
16894.436.F.3d.174.txt
2761.831.F.2d.538.txt
7434.544.F.3d.449.txt
```

[ ]: