|  |  |
| --- | --- |
|  | **Department of Computer Engineering Class: B.E. (Computer) (Div- B) (Sem-VIII)** |
|  | **Subject: Computational Lab-II (NLP)(CSL804)** |
| **Sr. No.** | **Title of Experiment** |
| **1.** | **Study of Python with reference to Natural Language Processing.** |

|  |
| --- |
| **Course Name:** Computational Lab-II(NLP) |
| **Course Code:**  CSL804 |
| **Experiment No.:** 01 |
| **Lab outcome:** Acquire practical knowledge within the chosen area of technology for project development. |
| **Name of Student:** Moin Memon |
| **Student Roll No.:** 275 |
| **Year/Semester/Div:** B.E./VIII/B |

**Experiment No. 01**

**Aim:** Study of Python with reference to Natural Language Processing.

**Theory:**

Getting Started with NLTK.

We will be using Python library NLTK (Natural Language Toolkit) for doing text analysis in English Language The Natural language toolkit (NLTK) is a collection of Python libraries designed especially for identifying and tag parts of speech found in the text of natural language like English.

Installing NLTK

Before starting to use NLTK we need to install it with the help of following command, we can install it in our Python environment. **pip install nltk**

If we are using Anaconda, then a Conda package for NI TK can be built by using the following command.

**conda install - anaconda nltk**

**Downloading NLTK's Data-**

I After installing NLTK, another important task is to download its preset text repositories so that it can be carefully used.However, before that we need to import NLTK the way we import any other Python module. The following command will help us in importing NLTK. **import nltk.**

Now download NLTK data with the help of the following command, **nltk.download()**

It will take some time to install all available packages of NLTK.

**Other Necessary Packages:**

Some other Python packages like gemum and pattern are also very necessary for text analysis as well as building natural language processing applications by using NLTK. the packages can be installed as shown below-

**1.Gensim:**

gensim is a robust semantic modeling library which can be used for many applications. We can install by following command.

**pip install gensim**

**2.Pattern:**

It can be used to make gensim package work properly. The following command helps in installing pattern- **pip install pattern**

**3.Tokenization:**

Tokenization may be defined as the Process of breaking the given text, into smaller units called tokens Words, numbers or punctuation marks can be tokens It may also be called word segmentation.

Example:

**Input -** Bed and chair are types of furniture.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bed |  | and |  | Chair |  | are |  | types |  | of |  | furniture |

**Output -**

We have different packages for tokenization provided by NLTK. We can use these packages based on our requirements. The packages and the details of their installation are as follows-

**Sent\_tokenize package**

This package can be used to divide the input text into sentences. We can import it by the following command- **from nltk\_tokenize import sent\_tokenize word\_tokenize package**

This package can be used to divide the input text into words. We can import it by the following command- **from nltk\_tokenize import sent\_tokenize**

**WordPunctTokenizer package**

This package can be used to divide the input text into words and punctuation marks. We can import it by the following command-

**from nltk\_tokenize import WordPunctTokenizer**

**4.Stemming:**

Due to grammatical reasons, language includes lots of variations in the sense that the language English as well as other languages too, have different forms of a word For example, the words like democracy democratic, and democratization. For machine learning projects, it is very important for machines to understand that these different words, like above, have the same base form That is why it is very useful to extract the base forms of the words while analyzing the text.

Stemming is a heuristic process that helps in extracting the base forms of the words by chopping of their ends. The different packages for stemming provided by NLTK module are as follows – **i). PorterStemmer package**

Porter's algorithm is used by this gemming package to extract the base form of the words. With the help of the following command, we can import this package- **from nltk stem porter import PorterStemmer**

For example, "write” would be the output of the word "writing” given as the input to this stemmer.

**ii). LancasterStemumer package**

Lancaster's algorithm is used by this stemming package to extract the base form of the words. With the help of following command, we can import this package – **from nltk stem lancaster import LancasterStemmer**

For example, “write” would be the output of the word “writing” given as the input to this stemmer **iii). Snow ballStemmer package**

Snowball's algorithm is used by this stemming package to extract the base form of the words. With the help of following command, we can import this package- **from stem.snowball impon SnowballStemmer**

For example, “write” would be the output of the word “writing” given as the input to this stemmer

**5.Lemmatization:**

It is another way to extract the base forms of words, normally aiming to remove inflectional endings by using vocabulary and morphological analysis. After Lemmatization, the base form of any word is lemma. NLTK module provides the following package for lemmatization- **i). Wordnet Lemmatizer package:**

**a) without pos tag**

Wordnet is an large, freely and publicly available lexical database for the English language aiming to establish structured semantic relationships between words. It offers lemmatization capabilities as well and is one of the earliest and most commonly used lemmatizes. With the help of following command, we can import this package – **from nltk.stem import WordNetLemmatizer lemmatizer.lemmatize('geese')**

For example, “geese” would be the output of the word "goose” given as the input to lemmatizer. **b) with appropriate POS tag**

It may not be possible manually provide the corrent POS tag for every word for large texts. So, instead, we will find out the correct POS tag for each word, map it to the right input character that the WordnetLemmatizer accepts and pass it as the second argument to lemmatize().The command to execute is- **from nltk.stem import WordNetLemmatizer lemmatizer.lemmatize(word, get\_wordnet\_pos(geese)))**

For example, “are” would be the output of the word "to be” given as the input to lemmatizer **ii). Gensim Lemmatize package:**

Gensim provide lemmatization facilities based on the `pattern` package. It can be implemented using the `lemmatize()` method in the `utils` module. By default, `lemmatize()` allows only the ‘JJ’, ‘VB’, ‘NN’ and ‘RB’ tags. Following is the package to execute it- **from gensim.utils import lemmatize**

The word “feet” would be lemmatized to word “foot” for the given lemmatizer.

**iii). TextBlob Lemmatizer:**

TexxtBlob is a powerful, fast and convenient NLP package as well. Using the Word and TextBlob objects, it’s quite straighforward to parse and lemmatize words and sentences respectively. The given package is as follows- **from textblob import TextBlob, Word**

For the given word “Stripes” as input the lemma generated is stripe.

**NLP Implementations:**

These are some of the successful implementations of Natural Language Processing (NLP):

**a).** **Search engines:** There are various search engines like Google, Yahoo, etc. Google search engine understands that you are a tech guy so it shows you results related to you.

**b). Social websites feed:** The social website like the Facebook news feed, community posts. The news feed algorithm understands your interests using natural language processing and shows you related Ads and posts more likely than other pasts.

**c). Speech engines:** The google assistant ,Alexa, Apple Siri, Cortana

**d). Spam filters:** The Google’s mail-based Spam filtering on Gmail Application. It's not just about the usual spam filtering, now spam filters understand what's inside the email content and see if it's a spam or not.

**NLP using Python:**

* **Natural Language Toolkit (NLTK):**

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, etc.

* **TextBlob:**

TextBlob is Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation. WordNet integration, parsing, word inflection, adds new models or languages through extensions and more.

* **Polyglot:**

Polyglot is a natural language pipeline which supports massive multilingual applications. The features include tokenisation, language detection, named entity recognition, part of speech tagging, sentiment analysis, word embeddings etc Polyglot depends on Numpy and libicu-dev.

* **Gensim:**

Gensim is a Python library for topic modelling document indexing and similarity retrieval with large corpora. Target audience is basically the natural language processing (NI P) and information retrieval (IR) community. The features of this library include such as all algorithms are memory-independent wrt the corpus size, intuitive interfaces, efficient multicore implementations of popular algorithms distributed computing, etc.

* **Core NLP:**

Stanford CoreNLP provides a set of human language technology tools Stanford CoreNLP's goal is to make it very easy to apply a bunch of linguistic analyst’s tools to a piece of text Stanford CoreNLP integrates many of Stanford's NLP tools, including the part-of-speech (POS) tagger, the named entity recognizer (NER), the parser the coreference resolution system, sentiment analysts, bootstrapped pattern learning, and the open information extraction tools.

* **spaCy:**

spaCy is a library for advanced Natural Language Processing in Python and Cython which comes with a number of interesting features. spaCy comes with pre-trained statistical models and word vectors. and currently supports tokenization for 49- languages. It features state-of-the-art speed, convolutional neural network models for tagging, parsing and named entity recognition and easy deep learning integration.

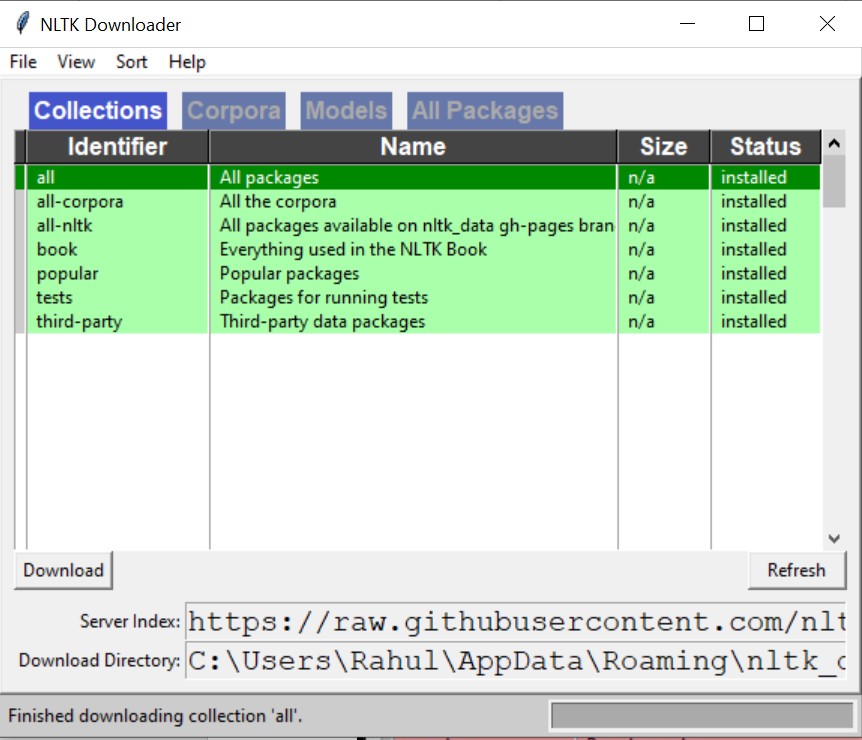
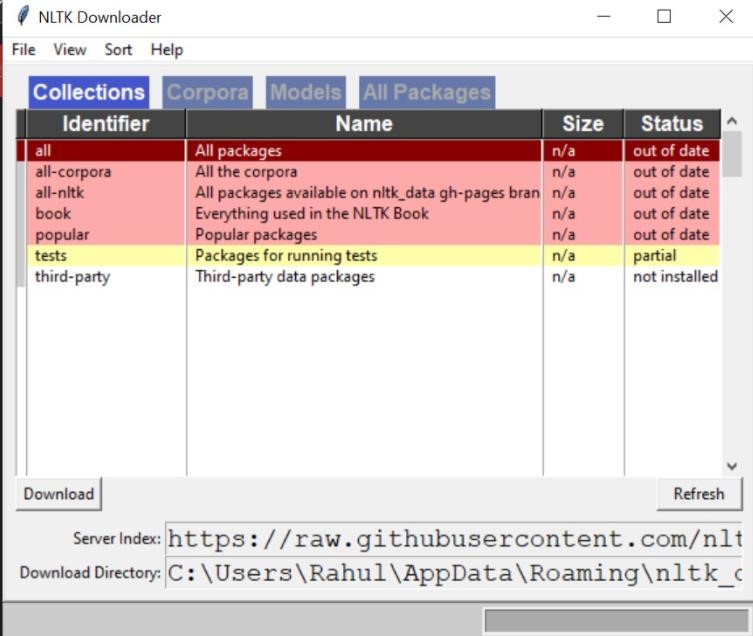
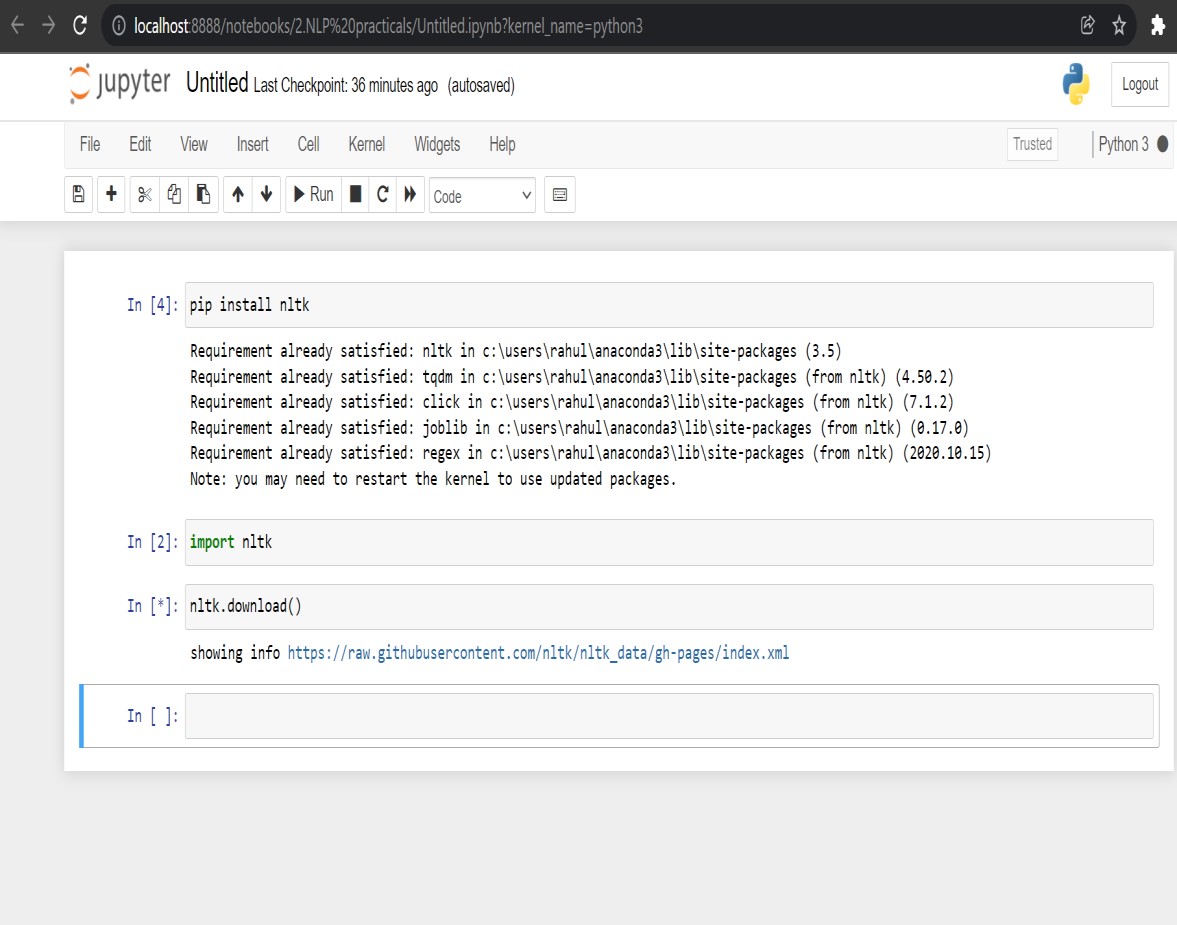
* **Pattern:**

Pattern is a web mining module for the Python programming language. It has tools for data mining. (Google, Twitter, and Wikipedia API, a web crawler, an HTML DOM parser), natural language processing (part-ofspeech taggers, n-gram search, sentiment analysis, WordNet), machine learning (vector space model, clustering SVM), network analysis by graph centrality and visualization

* **Vocabulary:**

Vocabulary is a Python library for natural language processing which is basically a dictionary in the form of Python module. Using this library, for a given word you can get its meaning, synonyms, antonyms, part of speech, translations and other such. This library is easy to install and is a substitute to Wordnet.

**Outputs:**



**Conclusion:** Hence, Python with reference to Natural Language Processing was studied.