simplilearn | PURDUE UNIVERSITY

Natural Language Processing

NLP Libraries

Learning Objectives

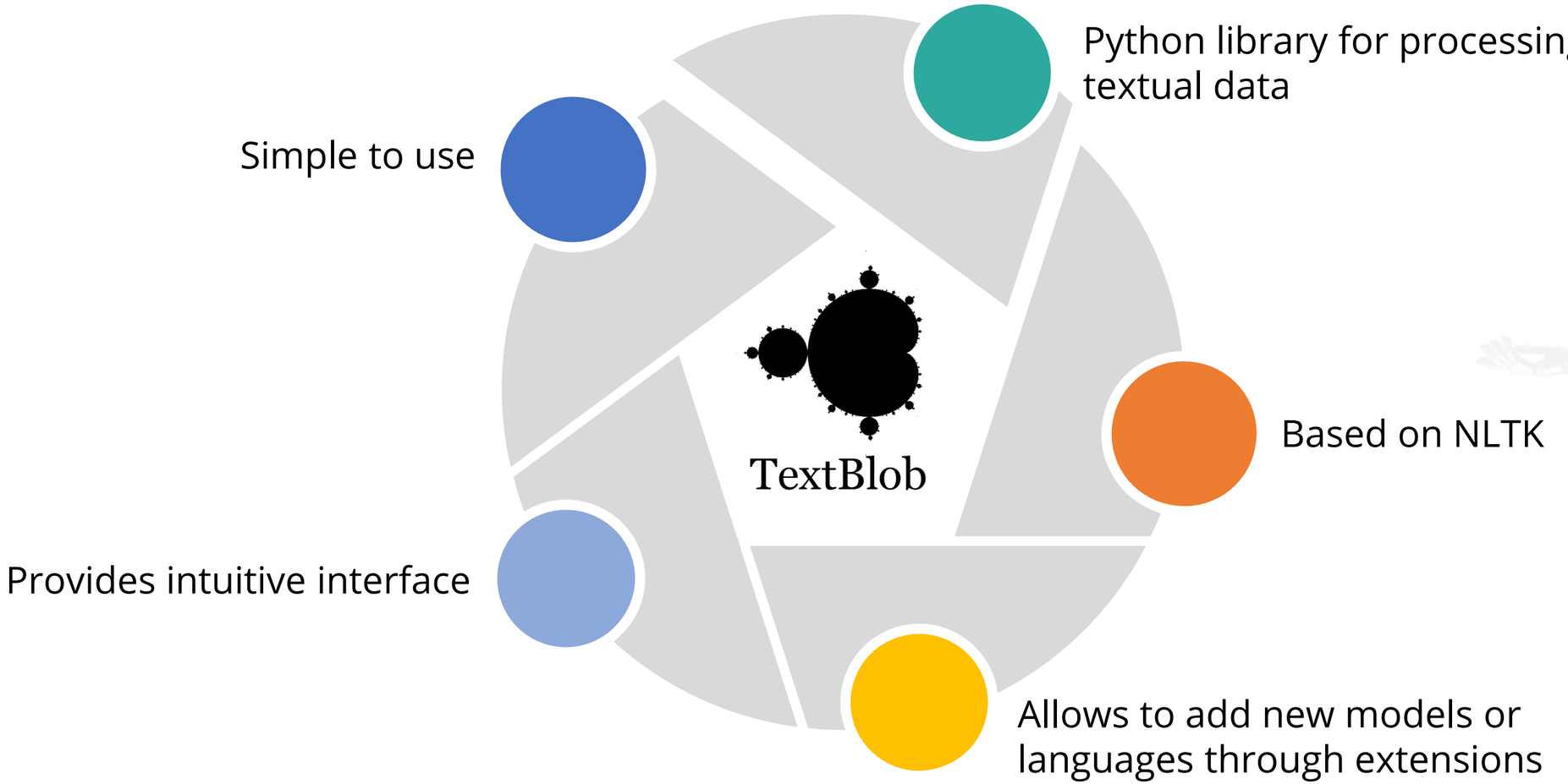
By the end of this lesson, you will be able to:

- Define and work with TextBlob, Vocabulary, and Polyglot
- Describe NLTK corpora
- Compare and understand the use of different libraries in NLP
- Demonstrate the spacy-based feature extraction from data

DATA AND
ARTIFICIAL INTELLIGENCE

Introduction to TextBlob

What Is TextBlob?



TextBlob APIs

- Part-of-Speech Tagging
- Noun Phrase Extension
- Classification
- Translation
- Language Translation and Detection
- Tokenization
- Parsing
- Word and Phrase Frequencies
- N-gram
- Word Inflection and Lemmatization
- Word Net Integration
- New Model Addition

TextBlob: Requirements

System Requirement:
- Operating system: macOS / OS X, Linux, and Windows
- Python version: Python >=2.7

Dependency: NLTK 3

```
import textblob
```

TextBlob: Word and Phrase Frequencies

```
#Textb lob
#Loading library

from textblob import TextBlob

#Textblob object
textblob_obj = TextBlob(""" Simplilearn is one of the world's
leading certification training providers.""")

#Counting word 'the'
word_count = textblob_obj.word_counts['the']
print("Word count of 'the' is: ", word_count)

#phrase count
phrase_count = textblob_obj.noun_phrases.count('world's leading')
print("Phrase count of 'artificial intelligence' is:
",phrase_count)
```

Output:

```
Word count of 'the' is:  2
Phrase count of 'world's leading' is:  1
```

`textblob_obj.word_counts[]`, returns the count of the specific word.

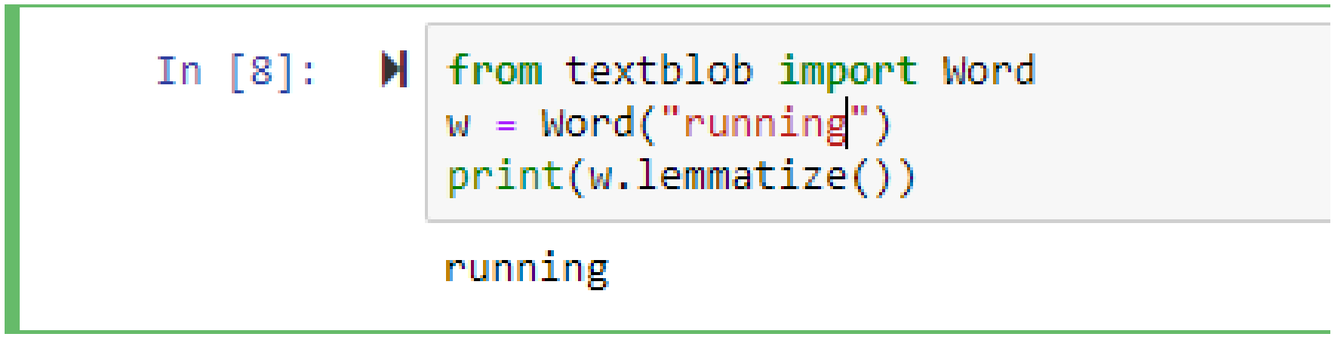
`textblob_obj.noun_phrases.count()method` returns the count of the specific phrase

TextBlob: Lemmatization

```
from textblob import Word
w = Word("running")
print(w.lemmatize())
```

Output:
running

.lemmatize() method, returns the word with all morphology (suffixes, etc.) removed.



TextBlob: Tokenization

```
from textblob import TextBlob
blob = TextBlob("Simplilearn is one of the world’s leading certification training
providers."
           "We partner with companies and individuals to address their unique needs."
           "We provide training and coaching that helps working professionals achieve
their career goals.")

blob.words
```

Attribute of the TextBlob object used here for Tokenization for words is .words

Output:

```
WordList(['Simplilearn', 'is', 'one', 'of', 'the', 'world', '’', 's', 'leading',
'certification', 'training', 'providers.We', 'partner', 'with', 'companies', 'and',
'individuals', 'to', 'address', 'their', 'unique', 'needs.We', 'provide', 'training',
'and', 'coaching', 'that', 'helps', 'working', 'professionals', 'achieve', 'their',
'career', 'goals’])

blob.sentences
```

Attribute of the TextBlob object used here for Tokenization for sentence is .sentences

Output:

```
[Sentence("Simplilearn is one of the world’s leading certification training providers.We
partner with companies and individuals to address their unique needs.We provide training
and coaching that helps working professionals achieve their career goals.")]
```

TextBlob: Tokenization

In [5]:
```
from textblob import TextBlob

blob = TextBlob("Simplilearn is one of the world's leading certification training providers."
           "We partner with companies and individuals to address their unique needs."
           "We provide training and coaching that helps working professionals achieve their career goals.")
blob.words
```

Out[5]:
```
[Sentence("Simplilearn is one of the world's leading certification training providers.We partner with companies and individu
als to address their unique needs.We provide training and coaching that helps working professionals achieve their career goa
ls.")]
```

In [6]:
```
blob.sentences
```

Out[6]:
```
[Sentence("Simplilearn is one of the world's leading certification training providers.We partner with companies and individu
als to address their unique needs.We provide training and coaching that helps working professionals achieve their career goa
ls.")]
```

TextBlob: Word Inflection

```
sentence = TextBlob('Get Certified Get Ahead Digital economy training
Driving innovation and accelerating career')
print (sentence.words)

print ("Singularize form of 2nd word: ",
sentence.words[1].singularize())
print ("Pulralize form of last word: ", sentence.words[-
1].pluralize())
['Get', 'Certified', 'Get', 'Ahead', 'Digital', 'economy', 'training',
'Driving', 'innovation', 'and', 'accelerating', 'career’]
```

Output:

```
Singularize form of 2nd word:  Certified
Pulralize form of last word:  careers
```

Singularize() method is check the singular form of a word and pluralize() method in vice versa

TextBlob: Word Inflection

In [35]:

```
#Textblob
#Loading library
from textblob import TextBlob

sentence = TextBlob('Get Certified Get Ahead Digital economy training Driving innovation and accelerating career')
print (sentence.words)

print ("Singularize form of 2nd word: ", sentence.words[1].singularize())

print ("Pulralize form of last word: ", sentence.words[-1].pluralize())
```

```
['Get', 'Certified', 'Get', 'Ahead', 'Digital', 'economy', 'training', 'Driving', 'innovation', 'and', 'accelerating', 'care
er']
Singularize form of 2nd word:  Certified
Pulralize form of last word:  careers
```

TextBlob: Part-of-Speech Tagging

```
from textblob import TextBlob
blob = TextBlob("Simplilearn is one of the world's leading
certification training providers.")
for word, pos in blob.tags:
    print (word, pos)
```

Output:

```
Simplilearn NNP
is VBZ
one CD
of IN
the DT
world NN
' NN
s NN
leading VBG
certification NN
training NN
providers NNS
```

Attribute of the TextBlob object used here for PoS tagging is .tags

.tags evaluates to a list of two-item tuples. Word and pos in this example are 2 variables which contain word and part of speech respectively.

TextBlob: Pluralization

```
from textblob import Word
w = Word("company")
print (w.pluralize())
```

Output:

```
companies
```

.pluralize() method to get the plural form of that word.

TextBlob: Sentiment Analysis

```
#Textblob
#Loading library
from textblob import TextBlob

textblob_obj = TextBlob("Simplilearn is one of the
world's leading certification training
providers.")

#Find sentiment score
sentiment_score = textblob_obj.sentiment

#Find sentiment polarity
sentiment_polarity =
textblob_obj.sentiment.polarity

print ("Score: ", sentiment_score)
print ("Polarity: ", sentiment_polarity)
```

Output:

```
Score:  Sentiment(polarity=0.0, subjectivity=0.0)
Polarity:  0.0
```

`textblob_obj.sentiment` is used to get the sentiment score of a sentence.

`textblob_obj.sentiment.polarity` is used to get the sentiment polarity score of a sentence.

TextBlob: Sentiment Analysis

In [45]:

```
#Textblob
#Loading library
from textblob import TextBlob

textblob_obj = TextBlob("Simplilearn is one of the world's leading certification training providers.")

#Find sentiment score
sentiment_score = textblob_obj.sentiment

#Find sentiment polarity
sentiment_polarity = textblob_obj.sentiment.polarity

print ("Score: ", sentiment_score)
print ("Polarity: ", sentiment_polarity)
```

```
Score:  Sentiment(polarity=0.0, subjectivity=0.0)
Polarity:  0.0
```

TextBlob: Spelling Correction

```
In [48]: b = TextBlob("Simplilearn isi one of the world's leading certification training providers.")
         print(b.correct())

         Simplilearn is one of the world's leading mortification training provides.
```

```
b = TextBlob("Simplilearn isi one of the world's leading
certification training providers.")
print(b.correct())
```

.correct() method to get the correct spelling

Output:

```
Simplilearn is one of the world's leading mortification
training provides.
```

```
In [51]: from textblob import Word
         w = Word('proviler')
         w.spellcheck()

Out[51]: [('provider', 1.0)]
```

```
from textblob import Word
w = Word('proviler')
w.spellcheck()
```

Word.spellcheck() method returns:
a list of (word, confidence) tuples

Output:

```
[('provider', 1.0)]
```

TextBlob: Translation and Language Detection

```
en_blob = TextBlob('Simplilearn is one of the
world’s leading certification training providers.')
en_blob.detect_language()
en_blob.translate(to='es’)

Output:
TextBlob("Simplilearn es uno de los principales
proveedores de capacitación en certificación del
mundo.")
```

New in version 0.5.0.

If no source language is specified, TextBlob will attempt to detect the language

TextBlob: N-Gram

```
blob = TextBlob("Simplilearn is one of the
world’s leading certification training
providers.")
blob.ngrams(n=3)
```

Output:

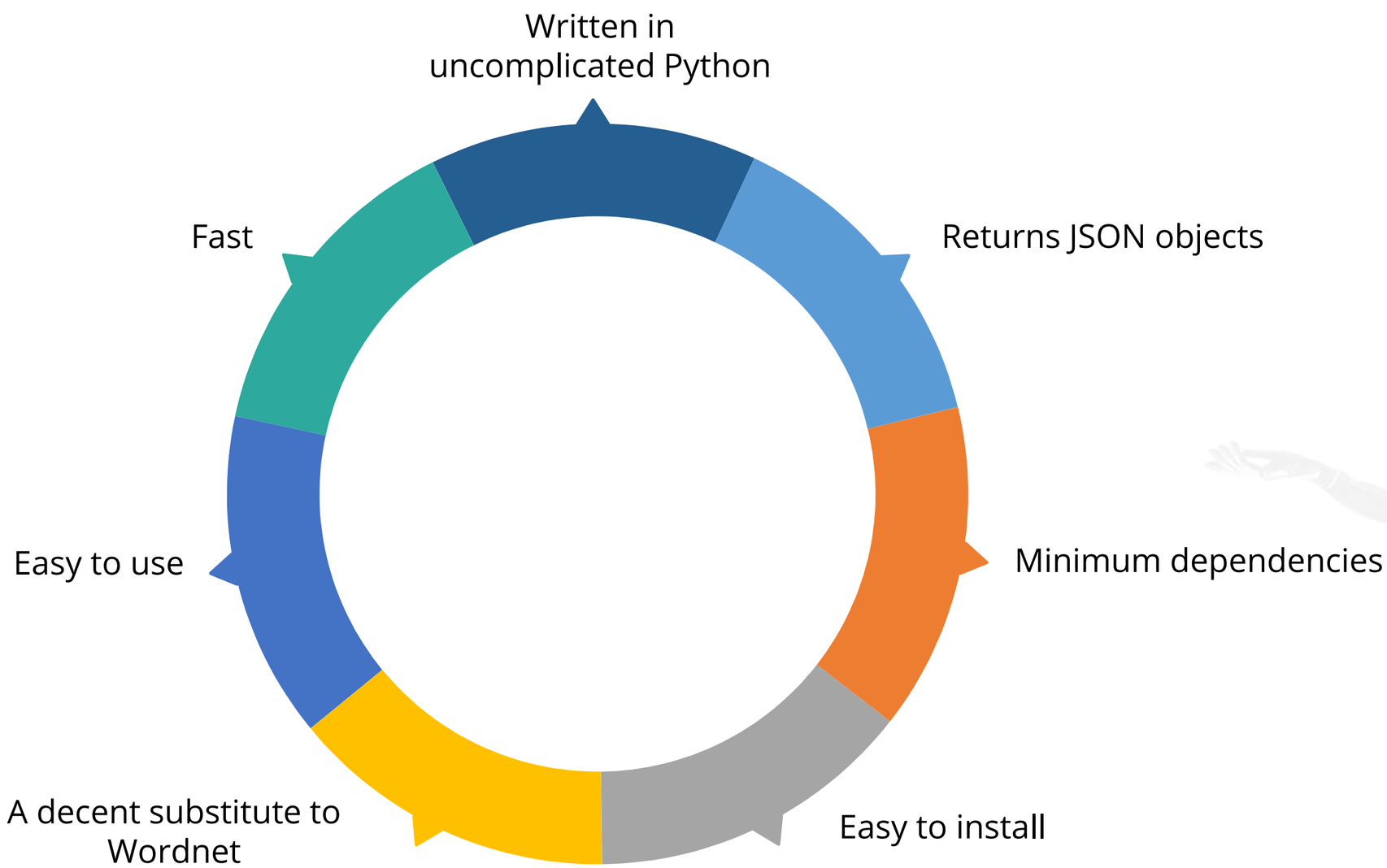
```
[WordList(['Simplilearn', 'is', 'one']),
 WordList(['is', 'one', 'of']),
 WordList(['one', 'of', 'the']),
 WordList(['of', 'the', 'world']),
 WordList(['the', 'world', '’']),
 WordList(['world', '’', 's']),
 WordList(['’', 's', 'leading']),
 WordList(['s', 'leading', 'certification']),
 WordList(['leading', 'certification',
'training']),
 WordList(['certification', 'training',
'providers'])]
```

TextBlob.ngrams() method returns a list of tuples of n successive words.

DATA AND
ARTIFICIAL INTELLIGENCE

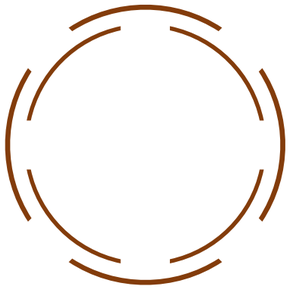
Introduction to Vocabulary

Vocabulary: Features

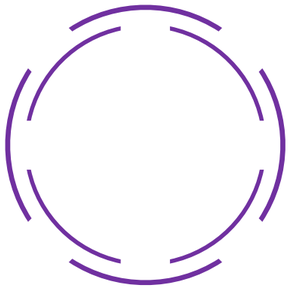


Vocabulary: Outputs

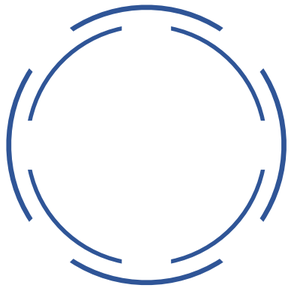
Using Vocabulary for a given word, you can get any of these outputs:



Meaning

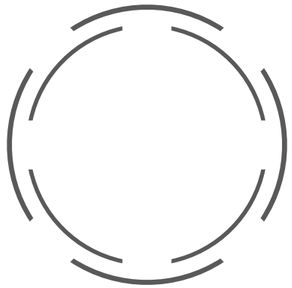


Antonyms

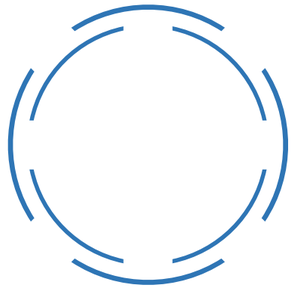


Synonyms

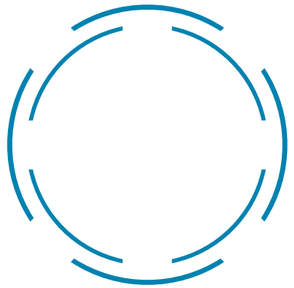
Parts-of-Speech



Translate



Pronunciation



Hyphenation

Vocabulary: Requirements

System Requirement:

- Operating system: macOS / OS X, Linux, and Windows
- Python Version: Python 3.6.8/3.7.2

```
import vocabulary
```

Vocabulary: Antonym and Synonym

```
In [11]: from vocabulary.vocabulary import Vocabulary as vb
         vb.antonym("boy")

Out[11]: '[{"seq": 0, "text": "daughter"}, {"seq": 1, "text": "female child"}, {"seq": 2, "text": "girl"}]'
```

```
from vocabulary.vocabulary import Vocabulary as vb
vb.antonym("boy")
```

Output:

```
'[{"seq": 0, "text": "daughter"}, {"seq": 1, "text": "female
child"}, {"seq": 2, "text": "girl"}]'
```

`vb.antonym()` method returns the antonym of a specific words.

```
from vocabulary.vocabulary import Vocabulary as vb
vb.synonym("girl")
```

Output:

```
'[{"seq": 0, "text": "daughter"}, {"seq": 1, "text":
"female child"}]'
```

`vb.synonym()` method returns the synonym of a specific words.

Vocabulary: Part-of-Speech

```
from vocabulary.vocabulary import Vocabulary as vb
vb.part_of_speech("boy")
```

`vb.part_of_speech()` method returns the part of speech of a specific word that it belongs to.

Vocabulary: Part-of-Speech

In [39]:
```
from vocabulary.vocabulary import Vocabulary as vb
vb.part_of_speech("boy")
```

Out[39]:

```
'[{"seq": 0, "text": "noun", "example": "A male child."}, {"seq": 1, "text": "noun", "example": "A son."}, {"seq": 2, "tex
t": "noun", "example": "A man, especially a young man."}, {"seq": 3, "text": "noun", "example": "A man socializing in a grou
p of men."}, {"seq": 4, "text": "noun", "example": "A male servant or employee."}, {"seq": 5, "text": "interjection", "examp
le": "Used to express mild astonishment, elation, or disgust."}, {"seq": 6, "text": null, "example": "To treat as a boy, or
as something belonging to or befitting a boy."}, {"seq": 7, "text": null, "example": "To act or represent in the manner of a
boy: in allusion to the acting by boys of women\'s parts on the stage."}, {"seq": 8, "text": "noun", "example": "In India, a
s far north as the Nerbudda river, a palankin-bearer. <em>Yule and Burnell</em>, Anglo-Ind. Glossary."}, {"seq": 9, "text":
"noun", "example": "A male child, from birth to full growth, but especially from the end of infancy to the beginning of yout
h: also applied to a young man, implying immaturity, want of vigor or judgment, etc."}, {"seq": 10, "text": "noun", "exampl
e": "In familiar or playful use (usually in the plural), a grown man regarded as one of the younger members of a family, as
an intimate friend or associate, or as having in any respect a boyish relation or character."}, {"seq": 11, "text": "noun",
"example": "Specifically, in the United States\\u2014 In the South, especially before the abolition of slavery, a negro ma
n."}, {"seq": 12, "text": "noun", "example": "An unscrupulous local politician, especially in a large city; one of the manag
ers or subordinates of the \\u201cmachine\\u201d of a party in local politics and elections: as, a ticket not acceptable to
the <em>boys.</em>"}, {"seq": 13, "text": "noun", "example": "A young servant; a page: as, \\u201c<em>boys</em>, grooms, and
lackeys,\\u201d"}, {"seq": 14, "text": "noun", "example": "[Supposed by some to be \\u201ca corruption of Hind. <em>bhaiee</
em>, a servant\\u201d; but the Hind. word, prop. <em>bh\\u0101\\u012b</em>, means \\u2018brother,\\u2019 and <em>boy</em> in
this use is merely the E. word. Cf. <internalXref urlencoded=\\"boy\\">boy</internalXref>.] In India and the treaty-ports of
China and Japan, etc., a native male servant, especially a personal servant; a butler or waiter, house-boy, office-boy, et
c., as distinguished from a coolie or porter: in common use among foreigners."}, {"seq": 15, "text": "noun", "example": "<st
rong>Old boy</strong>, a familiar name for the devil."}, {"seq": 16, "text": "noun", "example": "<strong>Roaring boys.</stro
ng> See <internalXref urlencoded=\\"roaring\\">roaring</internalXref>."}, {"seq": 17, "text": "transitive verb", "example":
"To act as a boy; -- in allusion to the former practice of boys acting women\'s parts on the stage."}, {"seq": 18, "text":
"noun", "example": "A male child, from birth to the age of puberty; a lad; hence, a son."}, {"seq": 19, "text": "noun", "exa
mple": "In various countries, a male servant, laborer, or slave of a native or inferior race; also, any man of such a race;
-- considered derogatory by those so called, and now seldom used."}, {"seq": 20, "text": "noun", "example": "a boy (usually
a chorister) elected bishop, in old Christian sports, and invested with robes and other insignia. He practiced a kind of mim
icry of the ceremonies in which the bishop usually officiated."}, {"seq": 21, "text": "noun", "example": "the Devil."}, {"se
q": 22, "text": "noun", "example": "guineas."}, {"seq": 23, "text": "noun", "example": "a popular English name of Southernwo
od (<spn>Artemisia abrotonum</spn>); -- called also <altname>lad\'s love</altname>."}, {"seq": 24, "text": "noun", "exampl
e": "childish amusements; anything trifling."}, {"seq": 25, "text": "noun", "example": "Male servant."}, {"seq": 26, "text":
"interjection", "example": "Exclamation of surprise, pleasure or longing."}, {"seq": 27, "text": "verb", "example": "To use
the word boy to refer to someone."}, {"seq": 28, "text": "verb", "example": "To act as a boy (in allusion to the former prac
tice of boys acting women\'s parts on the stage)."}, {"seq": 29, "text": "noun", "example": "a male human offspring"}, {"se
q": 30, "text": "noun", "example": "(ethnic slur) offensive and disparaging term for Black man"}, {"seq": 31, "text": "nou
n", "example": "a friendly informal reference to a grown man"}, {"seq": 32, "text": "noun", "example": "a youthful male pers
on"}]'
```

Vocabulary: Pronunciation

```
from vocabulary.vocabulary import Vocabulary as vb
vb.pronunciation("hippopotamus")
```

`vb.pronunciation()` method returns the pronunciation of a specific word.

Output:


```
'[{"seq": 0, "raw": "h\\u012dp\\u2033\\u0259-p\\u014ft\\u2032\\u0259-m\\u0259s",
"rawType": "ahd-5", "id": "H5222100", "attributionText": "from The American
Heritage\\u00ae Dictionary of the English Language, 5th Edition.",
"attributionUrl": "https://ahdictionary.com/"}, {"seq": 1, "raw": "HH IH2 P AH0 P
AA1 T AH0 M AH0 S", "rawType": "arpabet", "attributionText": "from The CMU
Pronouncing Dictionary.", "attributionUrl": "http://www.speech.cs.cmu.edu/cgi-
bin/cmudict"}, {"seq": 2, "raw":
"/\\u02cch\\u026ap.\\u0259\\u02c8p\\u0252t.\\u0259.m\\u0259s/", "rawType": "IPA",
"attributionText": "from Wiktionary, Creative Commons Attribution/Share-Alike
License.", "attributionUrl": "http://creativecommons.org/licenses/by-sa/3.0/"}]'
```

Vocabulary: Pronunciation

```
In [50]: from vocabulary.vocabulary import Vocabulary as vb
         vb.pronunciation("hippopotamus")
```

Out[50]:


```
'[{"seq": 0, "raw": "h\\u012dp\\u2033\\u0259-p\\u014ft\\u2032\\u0259-m\\u0259s", "rawType": "ahd-5", "id": "H5222100", "attr
ibutionText": "from The American Heritage\\u00ae Dictionary of the English Language, 5th Edition.", "attributionUrl": "http
s://ahdictionary.com/"}, {"seq": 1, "raw": "HH IH2 P AH0 P AA1 T AH0 M AH0 S", "rawType": "arpabet", "attributionText": "fro
m The CMU Pronouncing Dictionary.", "attributionUrl": "http://www.speech.cs.cmu.edu/cgi-bin/cmudict"}, {"seq": 2, "raw": "/
\\u02cch\\u026ap.\\u0259\\u02c8p\\u0252t.\\u0259.m\\u0259s/", "rawType": "IPA", "attributionText": "from Wiktionary, Creativ
e Commons Attribution/Share-Alike License.", "attributionUrl": "http://creativecommons.org/licenses/by-sa/3.0/"}]'
```

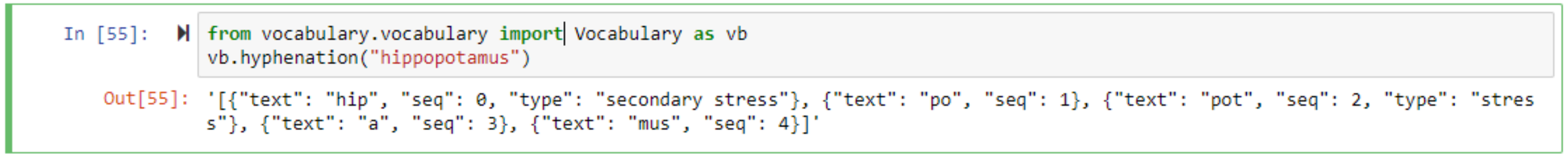
Vocabulary: Hyphenation

```
from vocabulary.vocabulary import Vocabulary as vb
vb.hyphenation("hippopotamus")
```

vb.hyphenation() method removes the hyphen from a specific word.

Output:

```
'[{"text": "hip", "seq": 0, "type": "secondary stress"}, {"text": "po", "seq": 1},
{"text": "pot", "seq": 2, "type": "stress"}, {"text": "a", "seq": 3}, {"text":
"mus", "seq": 4}]'
```



Vocabulary: Usage

```
from vocabulary.vocabulary import Vocabulary as vb
help(vb.translate)
```

To see the usage for any of the methods

```
In [60]: from vocabulary.vocabulary import Vocabulary as vb
         help(vb.translate)
```

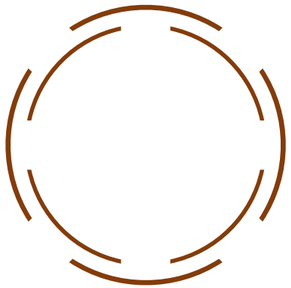
```
Help on function translate in module vocabulary.vocabulary:

translate(phrase, source_lang, dest_lang, format='json')
    Gets the translations for a given word, and returns possibilites as a list
    Calls the glosbe API for getting the translation

    <source_lang> and <dest_lang> languages should be specifed in 3-letter ISO 639-3 format,
    although many 2-letter codes (en, de, fr) will work.

    See http://en.wikipedia.org/wiki/List_of_ISO_639-3_codes for full list.

    :param phrase:  word for which translation is being found
    :param source_lang: Translation from language
    :param dest_lang: Translation to language
    :param format: response structure type. Defaults to: "json"
    :returns: returns a json object as str, False if invalid phrase
```

Polyglot

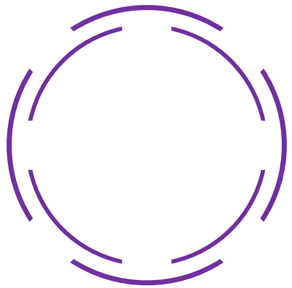
Polyglot: Introduction

Polyglot is used to support massive multilingual applications.

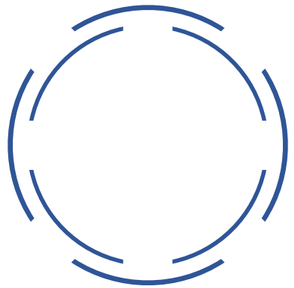
The following are the features of Polyglot:



Tokenization
(165 Languages)

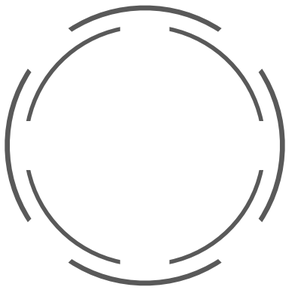


Language Detection
(196 Languages)

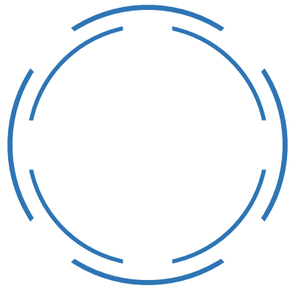


Name Entity Recognition
(40 Languages)

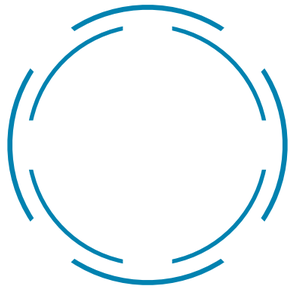
Parts-of-Speech Tagging
(16 Languages)



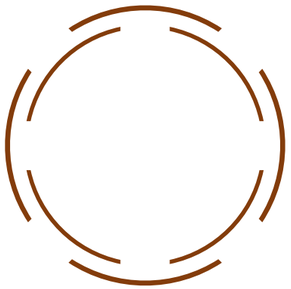
Sentiment Analysis
(136 Languages)



Word Embeddings
(137 Languages)



Morphological Analysis
(135 Languages)



Transliteration
(69 Languages)

Polyglot: Tokenization

```
import polyglot
from polyglot.text import Text, Word
zen = Text(" Simplilearn is one of the world’s leading
certification training providers.”
”We partner with companies and individuals to address
their unique needs, providing training and coaching that
helps working professionals achieve their career
goals.")
print(zen.words)
print (zen.sentences)
```

TextBlob.ngrams() method returns a list of tuples of n successive words.

```
['Beautiful', 'is', 'better', 'than', 'ugly', '.',
'Explicit', 'is', 'better', 'than', 'implicit', '.',
'Simple', 'is', 'better', 'than', 'complex', u'.']
```

```
[Sentence("Beautiful is better than ugly."),
Sentence("Explicit is better than implicit."),
Sentence("Simple is better than complex.")]
```

Polyglot: Part-of-Speech Tagging

```
import polyglot
from polyglot.text import Text, Word
text = Text(u"Depois de um pouco de educação indistinta foi
decidido que ele deveria ir para a Inglaterra para estudar
Direito na University College.")
print("{:<16}{}".format("Word", "POS Tag")+"\n"+"-"*30)
for word, tag in text.pos_tags:
print(u"{:<16}{:>2}".format(word, tag))
```

TextBlob.ngrams() method returns a list of tuples of n successive words.

Output:

```
Word             POS Tag
------------------------------
O                DET
primeiro         ADJ
uso              NOUN
de               ADP
desobediência    NOUN
civil            ADJ
em               ADP
massa            NOUN
ocorreu          ADJ
em               ADP
setembro         NOUN
de               ADP
1906             NUM
.                PUNCT
```

Polyglot: Named Entity Recognition

```
import polyglot
from polyglot.text import Text, Word
text = Text(u"In Großbritannien war Gandhi mit
dem westlichen Lebensstil vertraut geworden")
print(text.entities)
```

Output:

```
[I-LOC([u'Gro\xdfbritannien']), I-
PER([u'Gandhi'])]
```

TextBlob.ngrams() method returns a list of tuples of n successive words.

LUIS

LUIS: Language Understanding

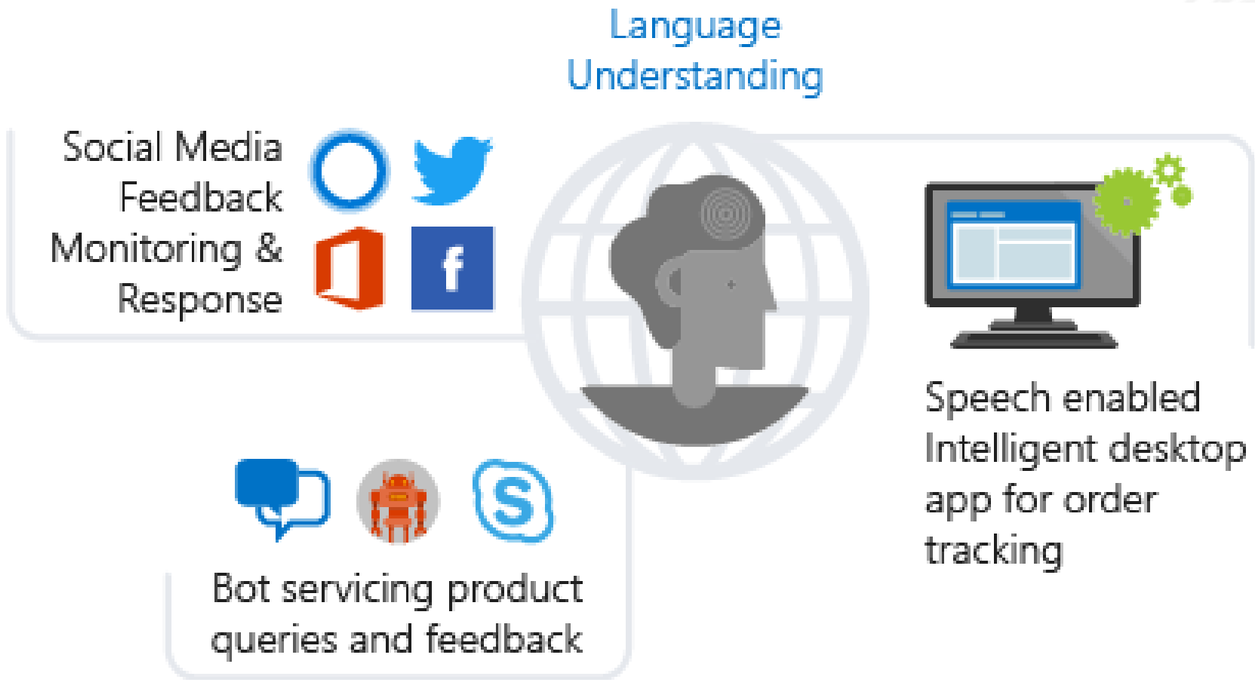
1. Cloud based API service
2. A machine learning-based service to build natural language into apps and bots
3. Integrates seamlessly with the Azure Bot Service
4. The service meets international compliance standards and supports 13 languages
5. Applies custom machine learning intelligence to a user's conversation and pulls out relevant or detailed information.

LUIS: Language Understanding

A client application for LUIS is any conversational application that communicates with a user in natural language to complete a task.

Examples:

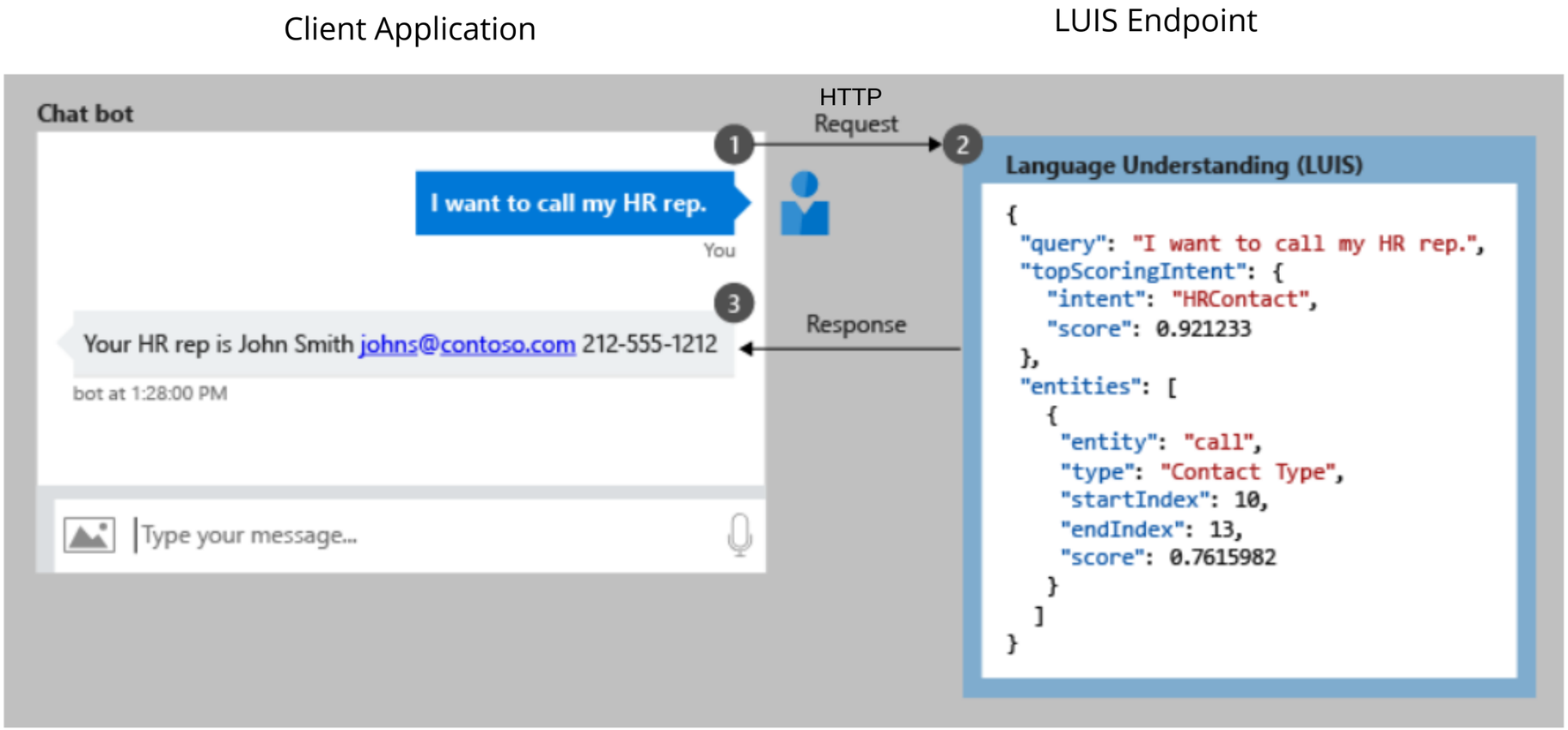
- Social media apps
- Chat bots
- Speech-enabled desktop applications



Source

LUIS Chatbot Application

The LUIS app provides intelligence, so that the client applications can make smart choices. But it does not provide those choices.



Source

Build LUIS Model

User Utterance

Intents

Intent represents a task or action the user wants to perform

Travel app intents	Example utterances
BookFlight	“Book me a flight to California next month” “Fly me to Rio on 10th Sep” “I require a flight ticket next Saturday to Los Angeles”
Greeting	“Hello“ “Welcome” “Good Afternoon”
CheckWeather	“What’s the weather like in Los Angeles” “Show me the forecast for next 5 days”
None	“Get me a nearby medical store location” “Did England win the ICC cricket world cup?”

Build LUIS Model

Intent represents a task or action the user wants to perform.

Entity represents words or phrases contained inside the utterance. The entities are intent agnostic

Create an intent when the user's intention would trigger an action in your client application. Example: Like a call to the checkweather() function. Then create an entity to represent parameters required to execute the action.

Build LUIS Model

Example Intent	Entity	Entity in Example Utterances
CheckWeather	{“type”:”location”,”entity”: ”Los Angeles”} {“type”:”builtin.datetimeV2.date”,”entity”: ”Monday”,”resolution”:”2019-08-26”}	What’s the weather like in Los Angeles on Monday?
CheckWeather	{{“type”:”date_range”,”entity”: ”next 5 days”}	Show me the forecast for next 5 days

Build LUIS Model


```
{
  "query":"where is the nearest club?",
  "topScoringIntent":{
     "intent":"LocationFinder",
     "score":0.251945645
  },
  "intents":[
     {
        "intent":"LocationFinder",
        "score":0.251945645
     },
     {
        "intent":"None",
        "score":0.08387748
     },
     {
        "intent":"BookFlight",
        "score":0.00309361285
     },
     {
        "intent":"Reminder",
        "score":0.002991894
     },
     {
        "intent":"FoodOrder",
        "score":1.4773135E-06
     }],
  "entities":[]
}
```

DATA AND
ARTIFICIAL INTELLIGENCE

NLTK Corpora

NLTK Corpora

- Corpus is a collection of written texts
- Corpora is the plural of corpus.
- nltk.corpus package defines a collection of corpus reader classes
- Each corpus reader class is specialized to handle a specific corpus format
- Most corpora consist of a set of files
- A list of identifiers for these files is accessed via the fileids() method of the corpus reader

NLTK Corpora: Accessing Objects

```
import nltk.corpus
# The Brown corpus:
print(str(nltk.corpus.brown).replace('\\\\','/'))

# The Penn Treebank Corpus:
print(str(nltk.corpus.treebank).replace('\\\\','/'))

# The Name Genders Corpus:
print(str(nltk.corpus.names).replace('\\\\','/'))

# The Inaugural Address Corpus:
print(str(nltk.corpus.inaugural).replace('\\\\','/’))
```

NLTK Corpora: Accessing Objects

Output:

<CategorizedTaggedCorpusReader in '.../corpora/brown' (not loaded yet)>

<BracketParseCorpusReader in '.../corpora/treebank/combined' (not loaded yet)>

<WordListCorpusReader in '.../corpora/names' (not loaded yet)>

<PlaintextCorpusReader in '.../corpora/inaugural' (not loaded yet)>

In [1]:

```
import nltk.corpus
# The Brown corpus:
print(str(nltk.corpus.brown).replace('\\\\','/'))

# The Penn Treebank Corpus:
print(str(nltk.corpus.treebank).replace('\\\\','/'))

# The Name Genders Corpus:
print(str(nltk.corpus.names).replace('\\\\','/'))

# The Inaugural Address Corpus:
print(str(nltk.corpus.inaugural).replace('\\\\','/'))
```

```
<CategorizedTaggedCorpusReader in '.../corpora/brown' (not loaded yet)>
<BracketParseCorpusReader in '.../corpora/treebank/combined' (not loaded yet)>
<WordListCorpusReader in '.../corpora/names' (not loaded yet)>
<PlaintextCorpusReader in '.../corpora/inaugural' (not loaded yet)>
```

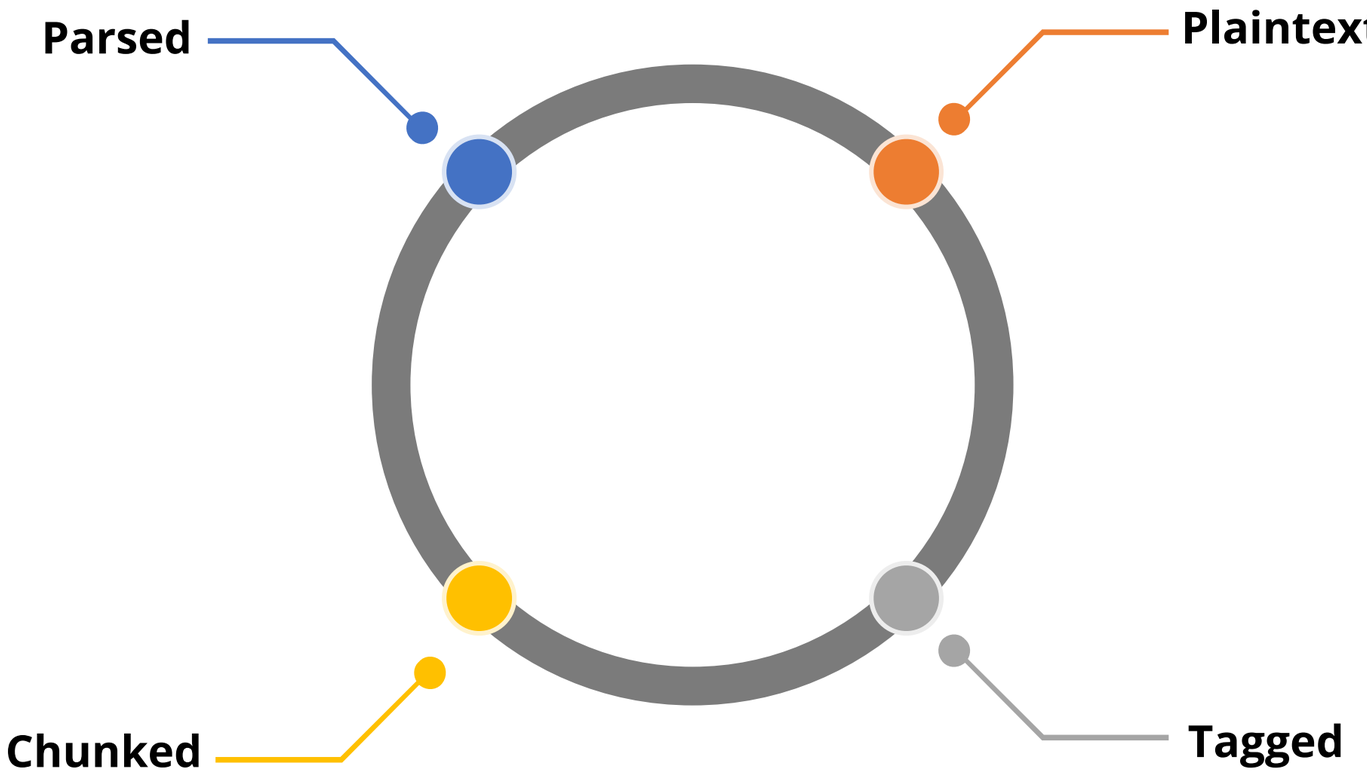
NLTK Corpora: Accessing Files and Fields

```
nltk.corpus.treebank.fileids) # doctest: +ELLIPSIS
nltk.corpus.inaugural.fileids) # doctest: +ELLIPSIS


Output:
['1789-Washington.txt', '1793-Washington.txt', '1797-Adams.txt', '1801-Jefferson.txt', '1805-Jefferson.txt', '1809-Madison.txt', '1813-Madison.txt', '1817-Monroe.txt', '1821-Monroe.txt', '1825-Adams.txt', '1829-Jackson.txt', '1833-Jackson.txt', '1837-VanBuren.txt', '1841-Harrison.txt', '1845-Polk.txt', '1849-Taylor.txt', '1853-Pierce.txt', '1857-Buchanan.txt', '1861-Lincoln.txt', '1865-Lincoln.txt', '1869-Grant.txt', '1873-Grant.txt', '1877-Hayes.txt', '1881-Garfield.txt', '1885-Cleveland.txt', '1889-Harrison.txt', '1893-Cleveland.txt', '1897-McKinley.txt', '1901-McKinley.txt', '1905-Roosevelt.txt', '1909-Taft.txt', '1913-Wilson.txt', '1917-Wilson.txt', '1921-Harding.txt', '1925-Coolidge.txt', '1929-Hoover.txt', '1933-Roosevelt.txt', '1937-Roosevelt.txt', '1941-Roosevelt.txt', '1945-Roosevelt.txt', '1949-Truman.txt', '1953-Eisenhower.txt', '1957-Eisenhower.txt', '1961-Kennedy.txt', '1965-Johnson.txt', '1969-Nixon.txt', '1973-Nixon.txt', '1977-Carter.txt', '1981-Reagan.txt', '1985-Reagan.txt', '1989-Bush.txt', '1993-Clinton.txt', '1997-Clinton.txt', '2001-Bush.txt', '2005-Bush.txt', '2009-Obama.txt', '2013-Obama.txt', '2017-Trump.txt']
```

NLTK Corpora: Accessing Files and Fields

```
In [5]: nltk.corpus.treebank.fileids() # doctest: +ELLIPSIS
        nltk.corpus.inaugural.fileids() # doctest: +ELLIPSIS

Out[5]: ['1789-Washington.txt',
         '1793-Washington.txt',
         '1797-Adams.txt',
         '1801-Jefferson.txt',
         '1805-Jefferson.txt',
         '1809-Madison.txt',
         '1813-Madison.txt',
         '1817-Monroe.txt',
         '1821-Monroe.txt',
         '1825-Adams.txt',
         '1829-Jackson.txt',
         '1833-Jackson.txt',
         '1837-VanBuren.txt',
         '1841-Harrison.txt',
         '1845-Polk.txt',
         '1849-Taylor.txt',
         '1853-Pierce.txt',
         '1857-Buchanan.txt',
         '1861-Lincoln.txt',
         '1865-Lincoln.txt',
         '1869-Grant.txt',
         '1873-Grant.txt',
         '1877-Hayes.txt',
         '1881-Garfield.txt',
         '1885-Cleveland.txt',
         '1889-Harrison.txt',
         '1893-Cleveland.txt',
         '1897-McKinley.txt',
         '1901-McKinley.txt',
         '1905-Roosevelt.txt',
         '1909-Taft.txt',
         '1913-Wilson.txt',
         '1917-Wilson.txt',
         '1921-Harding.txt',
         '1925-Coolidge.txt',
         '1929-Hoover.txt',
         '1933-Roosevelt.txt',
         '1937-Roosevelt.txt',
         '1941-Roosevelt.txt',
         '1945-Roosevelt.txt',
         '1949-Truman.txt',
         '1953-Eisenhower.txt',
         '1957-Eisenhower.txt',
         '1961-Kennedy.txt',
         '1965-Johnson.txt',
         '1969-Nixon.txt',
         '1973-Nixon.txt',
         '1977-Carter.txt',
         '1981-Reagan.txt',
         '1985-Reagan.txt',
         '1989-Bush.txt',
         '1993-Clinton.txt',
         '1997-Clinton.txt',
         '2001-Bush.txt',
         '2005-Bush.txt',
         '2009-Obama.txt',
         '2013-Obama.txt',
         '2017-Trump.txt']
```

Types of NLTK Corpora



DATA AND
ARTIFICIAL INTELLIGENCE

Comparison of Libraries

Libraries Comparison

	PROS	CONS
NLTK	Most popular library	Slow
	More used for academic purpose	
	Multiple approach for single task	Complicated to use
	Does Tokenization Fast	Splits text into sentences without analyzing the semantic structure
	Lot of third-party extensions	Does not provide neural network model
Spacy	Fastest	Less flexible compared to NLTK
	Simple to use: One method for one problem	Sentence tokenization is slower compared to NLTK
	More object oriented	Only supports 7 languages
	Uses Neural network	
	Provides built-in vectors	
Gensim	Good for large datasets	
	Supports only TF_IDF vectorization, word2vec, doc2vec, :LDA	Does not support complete NLP pipeline so needs to be used along with NLTK/Spacy
	Supports deep learning	

Libraries Comparison

	NLTK	SpaCy	Gensim
Popularity	Most popular	Closest competition to NLTK	Specialized library
Ease of Use	Complicated (multiple methods to do same thing)	Easy	highly optimized for (unsupervised) semantic (topic) modelling.
Speed	Slow	Fastest	Fast
Flexibility	More	Less	More
Neural network model	No	Yes	Yes
Integrated Word Vectors	No	Yes	Yes
Language Support	Largest number of languages supported	Only 7	Only 4
User defined Deep Learning Support	No	No	Yes
Spell Checker	No	No	Yes
Sentiment Detector	No	No	Yes
Pretrained model	Yes	Yes	No
Training Models	Yes	Yes	Yes
Full Python API	Yes	Yes	Yes

Spacy-Based Feature Extraction from Data

Problem Statement: Demonstrate the spacy-based feature extraction from data.

Access: Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

Demonstrate the Use of Libraries in NLP

Problem Statement: Understand the use of different libraries in NLP.

Access: Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

Create Your Own NLP Module

Objective: Create your own NLP module for text data preprocessing and morphological processing.

Problem Statement: You saw that there are so many different packages available for NLP tasks, each having its own benefits. It is tiresome to keep track of the specifics and formats needed for each of them. You are a Data Scientist at Flipkart who works with user reviews data frequently. You have been assigned the role of standardizing and simplifying the code by creating your own NLP module for data cleanup and morphological processing. You will be using TextBlob, NLTK in the background, and providing the user with an option to choose the package for each task.

LESSON-END PROJECT

Key Takeaways

You are now able to:

- Define and work with TextBlob, Vocabulary, and Polyglot
- Describe NLTK corpora
- Compare and understand the use of different libraries in NLP
- Demonstrate the spacy-based feature extraction from data

Knowledge Check

Knowledge Check

1

Which of the following are the NLP libraries for text processing?

a. Spacy

b. Gensim

c. Tesseract

d. Both a and b

Knowledge Check

1

Which of the following are the NLP libraries for text processing?

a. Spacy

b. Gensim

c. Tesseract

d. Both a and b

The correct answer is **d.**

Tesseract is an OCR library.

Knowledge Check

2

Which of the following library is mostly used for topic modeling?

a. Gensim

b. Vocabulary

c. Spacy

d. NLTK

Knowledge Check

2

Which of the following library is mostly used for topic modeling?

a. Gensim

b. Vocabulary

c. Spacy

d. NLTK

The correct answer is **a.**

Gensim provides a good facility for topic modeling based on LDA and it is open source.

Knowledge Check

3

Which of the following library has user defined deep learning support?

a. Gensim

b. Vocabulary

c. Spacy

d. NLTK

Knowledge Check

3

Which of the following library has user defined deep learning support?

a. Gensim

b. Vocabulary

c. Spacy

d. NLTK

The correct answer is **a.**

Gensim has user defined deep learning support.

Knowledge Check

4

Which library has sentiment detector inbuilt?

a. Gensim

b. Vocabulary

c. Spacy

d. NLTK

Knowledge Check

4

Which library has sentiment detector inbuilt?

a. Gensim

b. Vocabulary

c. Spacy

d. NLTK

The correct answer is **a.**

Gensim has sentiment detector inbuilt.

Knowledge Check

5

Which of the following library provides spell checker?

a. Gensim

b. Vocabulary

c. Spacy

d. NLTK

Knowledge Check

5

Which of the following library provides spell checker?

a. Gensim

b. Vocabulary

c. Spacy

d. NLTK

The correct answer is **a.**

Gensim provides spell checker.