NLP – NLP is a subfield of computer science and artificial intelligence concerned with interactions between computers and human (natural) languages. It is used to apply machine learning algorithms to text and speech.

1.NLU – Understanding (avoid ambiguity)

Lexical (words(document) level)

Syntactical Parsing (Sentence level) "Call me a CAB" or "Book a cab for me" which one is better??

Referential A->B I and my friend are going to food street as he is hungry.

2.NLG – Generations

Text planning

Sentence planning

Text realization

Steps: -

Lexical -> Syntactical -> Semantic -> Disclosure Integration (context analysis) -> Pragmatic Analysis (Reinterpret and meaning)

I am eating mango

Jhon eats mango

Mango eats jhon

Set alarm for tomorrow morning at 6, assume alarm is being set at 6PM.

Set alarm for 6 AM, (assume he is setting his on the same day at 12:05 AM)

**Introduction to the NLTK library for Python -** NLTK (**Natural Language Toolkit**)

import nltk

nltk.download()

Basic example demo (I will show during class)

The Basics of NLP for Text

We’ll cover the following topics:

1. Sentence Tokenization

Break sentences wherever you see punctuation

1. Word Tokenization

Break sentences wherever you see “”

1. Text Lemmatization and Stemming

Stemming is the process of reducing inflection in words to their root forms such as mapping a group of words to the same stem even if the stem itself is not a valid word in the Language."

Lemmatization, unlike Stemming, reduces the inflected words properly ensuring that the root word belongs to the language. In Lemmatization root word is called ***Lemma***. A lemma (plural lemmas or lemmata) is the canonical form, dictionary form, or citation form of a set of words.

Reducing the words to their base form

Remove derivationally words drive driving, play playing and

Use full resource for concepts: <https://www.datacamp.com/community/tutorials/stemming-lemmatization-python>

1. Stop Words

In computing, **stop words** are **words** which are filtered out before or after processing of natural language data (text). For **SEO** purposes, these are extremely common **words** that most search engines skip over in order to save space in their databases, and to speed up the process of crawling/indexing.

The, a, an, etc.

1. Regex

Regular expression

* . - match any character except newline
* \w - match word
* \d - match digit
* \s - match whitespace
* \W - match not word
* \D - match not digit
* \S - match not whitespace
* [abc] - match any of a, b, or c
* [^abc] - not match a, b, or c
* [a-g] - match a character between a & g

1. Bag-of-Words

Machine learning algorithms cannot work with raw text directly; we need to convert the text into vectors of numbers. It describes the occurrence of each word within a document.

A very common feature extraction procedure for sentences and documents is the bag-of-words approach (BOW). In this approach, we look at the histogram of the words within the text, i.e. considering each word count as a feature.

**It was the best of times,**

**it was the worst of times,**

**it was the age of wisdom,**

**it was the age of foolishness.**

**Techniques: -**

**Ignoring the case of the words**

**Ignoring punctuation**

**Removing the stop words from our documents**

**Reducing the words to their base form (Text Lemmatization and Stemming)**

**Fixing misspelled words**

**Limitations:**

**Vocabulary: The vocabulary requires careful design, most specifically in order to manage the size, which impacts the sparsity of the document representations.**

**Sparsity: Sparse representations are harder to model both for computational reasons (space and time complexity) and also for information reasons, where the challenge is for the models to harness so little information in such a large representational space.**

**Meaning: Discarding word order ignores the context, and in turn meaning of words in the document (semantics). Context and meaning can offer a lot to the model, that if modeled could tell the difference between the same words differently arranged (“this is interesting” vs “is this interesting”), synonyms (“old bike” vs “used bike”), and much more.**

**Group words: n-grams:** An n-gram is a **sequence of**a number of **items**(words, letter, numbers, digits, etc.).

A unigram is one word, a bigram is a sequence of two words, a trigram is a sequence of three words etc. The “n” in the “n-gram” refers to the number of the grouped words.

The office building is open today

the office

office building

building is

is open

open today

Scoring Words: Once, we have created our vocabulary of known words, we need to score the occurrence of the words in our data. We saw one very simple approach - the binary approach (1 for presence, 0 for absence).

Counts. Count the number of times each word appears in a document.

Frequencies. Calculate the frequency that each word appears in document out of all the words in the document.

**document in a collection or corpus**

1. TF-IDF

Wx,y = tfx,y x log (N/dfx)

Consider a document containing 100 words wherein the word *cat* appears 3 times. The term frequency (i.e., tf) for *cat* is then (3 / 100) = 0.03. Now, assume we have 10 million documents and the word *cat* appears in one thousands of these. Then, the inverse document frequency (i.e., idf) is calculated as log (10,000,000 / 1,000) = 4. Thus, the Tf-idf weight is the product of these quantities: 0.03 \* 4 = 0.12.

Doc1: cat cat cat

Doc2: cat cat cat dog

Doc3: cat dog mouse

Doc4: cat cat dog dog dog

Doc5: mouse

**Doc 1:** Ben studies about computers in Computer Lab.  
**Doc 2:** Steve teaches at Brown University.  
**Doc 3:** Data Scientists work on large datasets.

POS Tagging:

It is a process of converting a sentence to forms – list of words, list of tuples (where each tuple is having a form (word, tag)). The tag in case of is a part-of-speech tag, and signifies whether the word is a noun, adjective, verb, and so on.

**Corpus:**Body of text, singular. Corpora is the plural of this.  
**Lexicon :**Words and their meanings.  
**Token :** Each “entity” that is a part of whatever was split up based on rules.

In corpus linguistics,**part-of-speech tagging**(**POS tagging** or**PoS tagging** or **POST**), also called **grammatical tagging** or **word-category disambiguation.**

Input: Everything is all about money.

Output: [('Everything', 'NN'), ('is', 'VBZ'),

('all', 'DT'), ('about', 'IN'),

('money', 'NN'), ('.', '.')]

Common list:

NN – Singular Noun

V – Verb

A – Adjective

R - Adverb

Possible list of POS:

*CC coordinating conjunction  
CD cardinal digit  
DT determiner  
EX existential there (like: “there is” … think of it like “there exists”)  
FW foreign word  
IN preposition/subordinating conjunction  
JJ adjective ‘big’  
JJR adjective, comparative ‘bigger’  
JJS adjective, superlative ‘biggest’  
LS list marker 1)  
MD modal could, will  
NN noun, singular ‘desk’  
NNS noun plural ‘desks’  
NNP proper noun, singular ‘Harrison’  
NNPS proper noun, plural ‘Americans’  
PDT predeterminer ‘all the kids’  
POS possessive ending parent‘s  
PRP personal pronoun I, he, she  
PRP$ possessive pronoun my, his, hers  
RB adverb very, silently,  
RBR adverb, comparative better  
RBS adverb, superlative best  
RP particle give up  
TO to go ‘to‘ the store.  
UH interjection errrrrrrrm  
VB verb, base form take  
VBD verb, past tense took  
VBG verb, gerund/present participle taking  
VBN verb, past participle taken  
VBP verb, sing. present, non-3d take  
VBZ verb, 3rd person sing. present takes  
WDT wh-determiner which  
WP wh-pronoun who, what  
WP$ possessive wh-pronoun whose  
WRB wh-abverb where, when*

What is Chunking?

Chunking is a process of extracting phrases from unstructured text. Instead of just simple tokens which may not represent the actual meaning of the text, it’s advisable to use phrases such as “**South Africa**” as a single word instead of ‘**South**’ and ‘**Africa**’ separate words.

Chunking works on top of POS tagging, it uses pos-tags as input and provides chunks as output. Similar to POS tags, there are a standard set of Chunk tags like Noun Phrase(NP), Verb Phrase (VP), etc. Chunking is very important when you want to extract information from text such as Locations, Person Names etc. In NLP called **Named Entity Extraction**.

Ref:- <https://medium.com/greyatom/learning-pos-tagging-chunking-in-nlp-85f7f811a8cb>

Sentiment, semantic and syntactic

Text: “The burger and oreo crunch at Mc’D is simply awesome and smooth on pockets! I would recommed it anyday ahead of Burger King.”

Sentiment Analysis would simply say, ‘Hey look this review is positive. ’

Semantic Analysis on the other hand would bring up many different information on the plate: ‘The review is about food.’, ‘The review is about the quality as well as price.’, ‘Two products: burger and oreo crunch have been mentioned.’ and so on.

**Regular Expression**

###### BASIC SYNTAX

. One character except new line

\. A period. \ escapes a special character.

\d One digit

\D One non-digit

\w One word character including digits

\W One non-word character

\s One whitespace

\S One non-whitespace

\b Word boundary

\n Newline

\t Tab

###### MODIFIERS

$ End of string

^ Start of string

ab|cd Matches ab or de.

[ab-d] One character of: a, b, c, d

[^ab-d] One character except: a, b, c, d

() Items within parenthesis are retrieved

(a(bc)) Items within the sub-parenthesis are retrieved

###### REPETITIONS

[ab]{2} Exactly 2 continuous occurrences of a or b

[ab]{2,5} 2 to 5 continuous occurrences of a or b

[ab]{2,} 2 or more continuous occurrences of a or b

+ One or more

\* Zero or more

? 0 or 1

Reference:

<https://medium.com/factory-mind/regex-tutorial-a-simple-cheatsheet-by-examples-649dc1c3f285>

Read first for compile, findall, search, match, split and substitute

<https://sites.google.com/site/gothnlp/links/regular-expressions> for functions

Practical1: - Write regular expressions to fulfil the following requirements: 15-20 Minutes Max

* Any character except for a new line
* A period
* Any digit
* Anything but a digit
* Any character, including digits
* Anything but a character
* Collection of characters
* Match something up to ‘n’ times
* Match 1 or more occurrences
* Match any number of occurrences (0 or more times)
* Match exactly zero or one occurrence
* Match word boundaries
* Extract the user id, domain name and suffix from the following email addresses.

emails = """awais @facebook.com,

yasir @google.com,

mazhar@amazon.com"""

desired\_output:

[(awais, 'facebook', 'com'),

(yasir, 'google', 'com'),

(mazhar, 'amazon', 'com')]

Practical2: - **Sentiment Analysis 40-50 Minutes**

**Goal: Find most informative features (words) in the reviews**

1. Go through from provided dataset
2. Load basic libraries
3. Load dataset in Dataframe
4. Apply basic descriptive analytics \*if possible
5. Apply text pre-processing techniques
6. Apply basic descriptive analytics
7. Create BoW
8. Compute TF-IDF
9. Apply Naive Bayes/Random Forest Model (I will help out)

Additional Recommended reading:

EDA: Visually representing the content of a text document is one of the most important tasks in the field of [text mining](https://en.wikipedia.org/wiki/Text_mining).

<https://towardsdatascience.com/a-complete-exploratory-data-analysis-and-visualization-for-text-data-29fb1b96fb6a>

Assignment: -

1. Build Document Classifier & Spell Checker with Python
2. Create corpus of Urdu/Sindhi/Panjapi
3. Sentiment analysis of Roman Urdu