1. What do you understand by Natural Language Processing?

Natural Language Processing is a field of computer science that deals with communication between computer systems and humans. It is a technique used in Artificial Intelligence and Machine Learning. It is used to create automated software that helps understand human spoken languages to extract useful information from the data it gets in the form of audio. Techniques in NLP allow computer systems to process and interpret data in the form of natural languages.

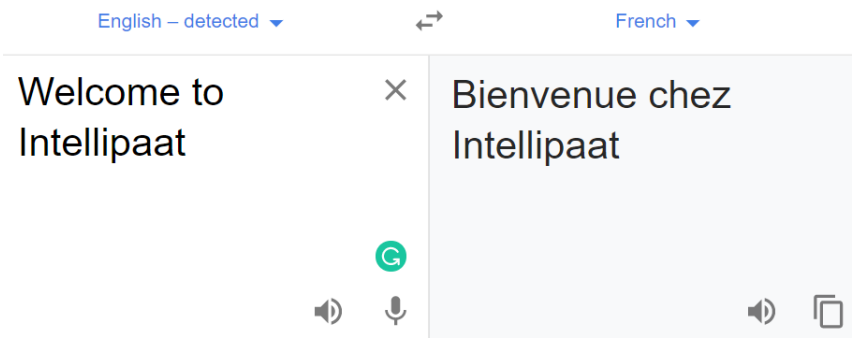
2. What are stop words?

Stop words are said to be useless data for a search engine. Words such as articles, prepositions, etc. are considered as stop words. There are stop words such as was, were, is, am, the, a, an, how, why, and many more. In Natural Language Processing, we eliminate the stop words to understand and analyze the meaning of a sentence. The removal of stop words is one of the most important tasks for search engines. Engineers design the algorithms of search engines in such a way that they ignore the use of stop words. This helps show the relevant search result for a query.

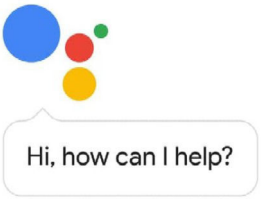
3. List any two real-life applications of Natural Language Processing.

Two real-life applications of Natural Language Processing are as follows:

1. **Google Translate**: Google Translate is one of the famous applications of Natural Language Processing. It helps convert written or spoken sentences into any language. Also, we can find the correct pronunciation and meaning of a word by using Google Translate. It uses advanced techniques of Natural Language Processing to achieve success in translating sentences into various languages.



2. **Chatbots**: To provide a better customer support service, companies have started using chatbots for 24/7 service. Chatbots helps resolve the basic queries of customers. If a chatbot is not able to resolve any query, then it forwards it to the support team, while still engaging the customer. It helps make customers feel that the customer support team is quickly attending them. With the help of chatbots, companies have become capable of building cordial relations with customers. It is only possible with the help of Natural Language Processing.



4. What is TF-IDF?

TFIDF or Term Frequency-Inverse Document Frequency indicates the importance of a word in a set. It helps in information retrieval with numerical statistics. For a specific document, TF-IDF shows a frequency that helps identify the keywords in a document. The major use of TF-IDF in NLP is the extraction of useful information from crucial documents by statistical data. It is ideally used to classify and summarize the text in documents and filter out stop words.

TF helps calculate the ratio of the frequency of a term in a document and the total number of terms. Whereas, **IDF** denotes the importance of the term in a document.

The formula for calculating TF-IDF:

TF(W) = (Frequency of W in a document)/(The total number of terms in the document)

IDF(W) = log_e(The total number of documents/The number of documents having the term W)

When **TF*IDF** is high, the frequency of the term is less and vice versa.

Google uses TF-IDF to decide the index of search results according to the relevancy of pages. The design of the TF-IDF algorithm helps optimize the search results in Google. It helps quality content rank up in search results.

5. What is Syntactic Analysis?

Syntactic analysis is a technique of analyzing sentences to extract meaning from it. Using syntactic analysis, a machine can analyze and understand the order of words arranged in a sentence. NLP employs grammar rules of a language that helps in the syntactic analysis of the combination and order of words in documents.

The techniques used for syntactic analysis are as follows:

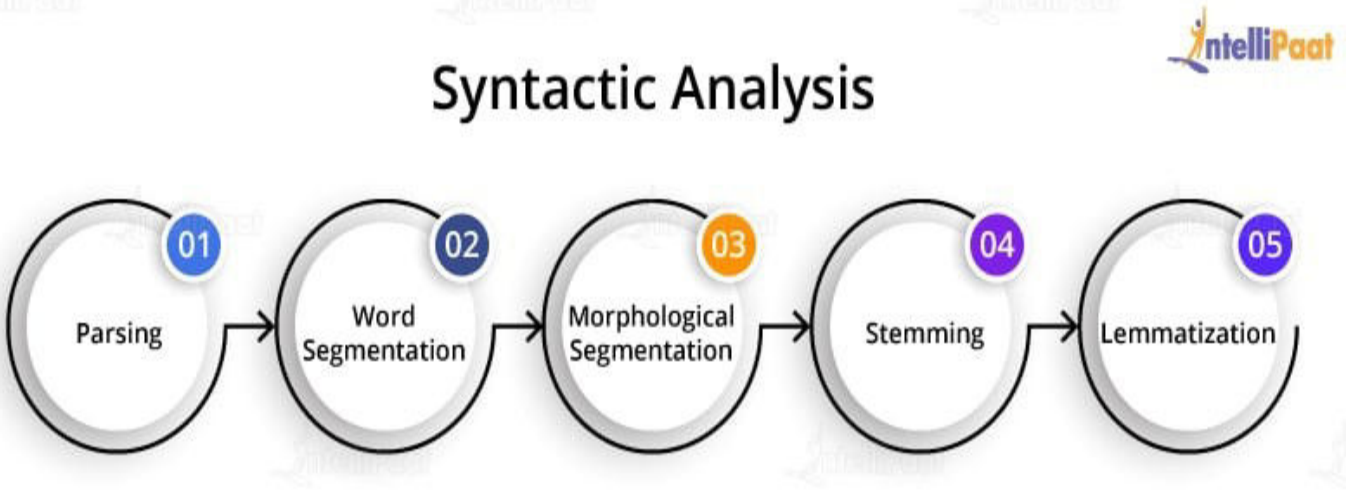
1. Parsing: It helps in deciding the structure of a sentence or text in a document. It helps analyze the words in the text based on the grammar of the language.

2. Word segmentation: The segmentation of words segregates the text into small significant units.

3. **Morphological segmentation**: The purpose of morphological segmentation is to break words into their base form.

4. Stemming: It is the process of removing the suffix from a word to obtain its root word.

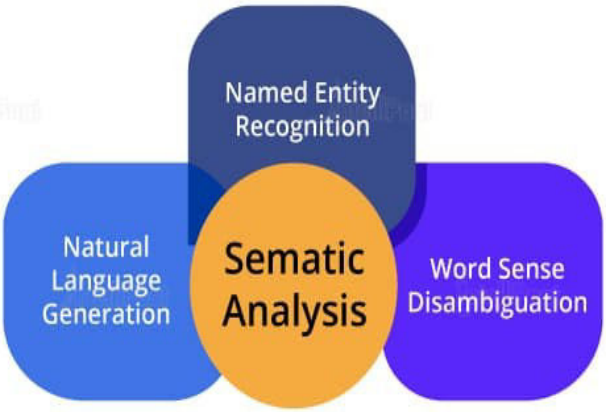
5. **Lemmatization**: It helps combine words using suffixes, without altering the meaning of the word.



6. What is Semantic Analysis?

Semantic analysis helps make a machine understand the meaning of a text. It uses various algorithms for the interpretation of words in sentences. It also helps understand the structure of a sentence.

Techniques used for semantic analysis are as given below:



1. **Named entity recognition:** This is the process of information retrieval that helps identify entities such as the name of a person, organization, place, time, emotion, etc.
2. **Word sense disambiguation:** It helps identify the sense of a word used in different sentences.
3. **Natural language generation**: It is a process used by the software to convert the structured data into human spoken languages. By using NLG, organizations can automate content for custom reports.

7. What is NLTK?

NLTK is a Python library, which stands for Natural Language Toolkit. We use NLTK to process data in human spoken languages. NLTK allows us to apply techniques such as parsing, tokenization,

lemmatization, stemming, and more to understand natural languages. It helps in categorizing text, parsing linguistic structure, analyzing documents, etc.

A few of the libraries of the NLTK package that we often use in NLP are:

1. SequentialBackoffTagger
2. DefaultTagger
3. UnigramTagger
4. treebank
5. wordnet
6. FreqDist
7. patterns
8. RegexpTagger
9. backoff_tagger
10. UnigramTagger, BigramTagger, and TrigramTagger

8. How to tokenize a sentence using the nltk package?

Tokenization is a process used in NLP to split a sentence into tokens. **Sentence tokenization** refers to splitting a text or paragraph into sentences.

For tokenizing, we will import **sent_tokenize** from the **nltk package**:

```
from nltk.tokenize import sent_tokenize<>
```

We will use the below paragraph for sentence tokenization:
Para = “Hi Guys. Welcome to Intellipaat. This is a blog on the NLP interview questions and answers.”

```
sent_tokenize(Para)
```

Output:

```
[ 'Hi Guys.' ,
'Welcome to Intellipaat. ',
'This is a blog on the NLP interview questions and answers. ' ]
```

Tokenizing a word refers to splitting a sentence into words.

Now, to tokenize a word, we will import **word_tokenize** from the nltk package.

```
from nltk.tokenize import word_tokenize
```

Para = “Hi Guys. Welcome to Intellipaat. This is a blog on the NLP interview questions and answers.”

```
word_tokenize(Para)
```

Output:

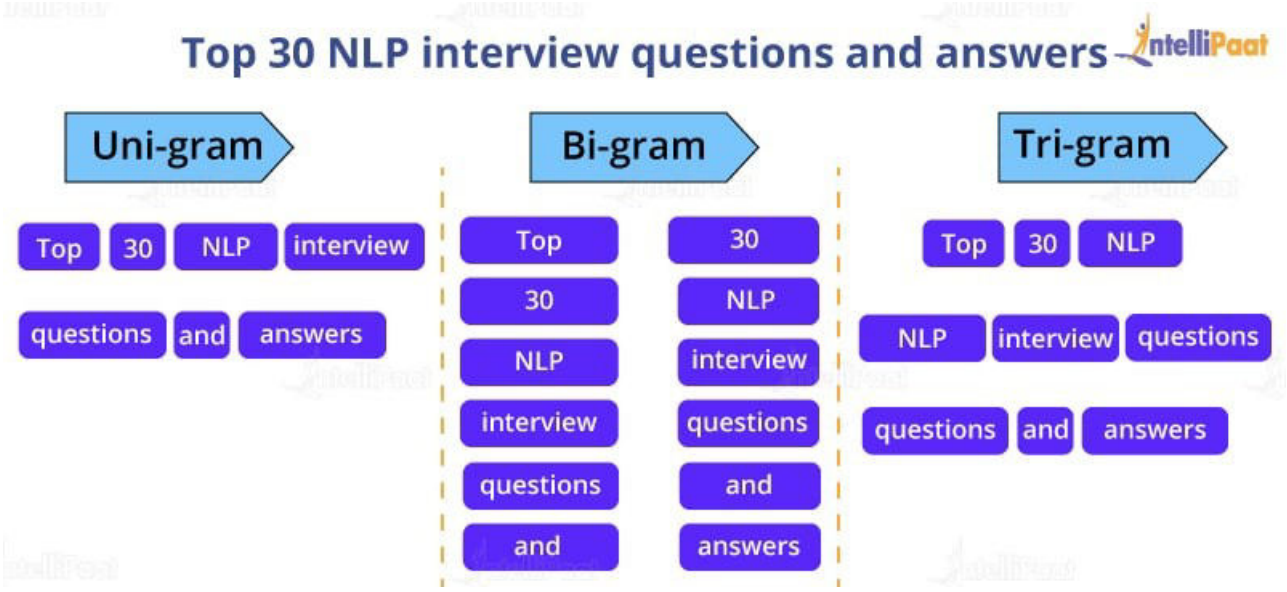
```
[ 'Hi' , 'Guys' , ' . ' , 'Welcome' , 'to' , 'Intellipaat' , ' . ' , 'This' ,
'is' ,   'a', 'blog' , 'on' , 'the' , 'NLP' , 'interview' , 'questions' , 'and' ,
'answers' , ' . ' ]
```

9. Explain how we can do parsing.

Parsing is the method to identify and understand the syntactic structure of a text. It is done by analyzing the individual elements of the text. The machine parses the text one word at a time, then two at a time, further three, and so on.

- When the machine parses the text one word at a time, then it is a **unigram**.
- When the text is parsed two words at a time, it is a **bigram**.
- The set of words is a **trigram** when the machine parses three words at a time.

Look at the below diagram to understand unigram, bigram, and trigram.



Now, let's implement parsing with the help of the **nltk** package.

```
import nltk
text = ”Top 30 NLP interview questions and answers”
```

We will now tokenize the text using **word_tokenize**.

```
text_token= word_tokenize(text)
```

Now, we will use the function for extracting unigrams, bigrams, and trigrams.

```
list(nltk.unigrams(text))
```

Output:

```
[ "Top 30 NLP interview questions and answer"]

list(nltk.bigrams(text))
```

Output:

```
["Top 30", "30 NLP", "NLP interview", "interview questions",   "questions and",
"and answer"]
```

```
list(nltk.trigrams(text))
```

Output:

```
["Top 30 NLP", "NLP interview questions", "questions and answers"]
```

For extracting **n-grams**, we can use the function **nltk.ngrams** and give the argument *n* for the number of parsers.

```
list(nltk.ngrams(text,n))
```

10. Explain Stemming with the help of an example.

In Natural Language Processing, stemming is the method to extract the root word by removing suffixes and prefixes from a word.
For example, we can reduce ‘stemming’ to ‘stem’ by removing ‘m’ and ‘ing.’
We use various algorithms for implementing stemming, and one of them is PorterStemmer.
First, we will import **PorterStemmer** from the nltk package.

```
from nltk.stem import PorterStemmer
```

Creating an object for PorterStemmer

```
pst=PorterStemmer()
pst.stem(“running”), pst.stem(“cookies”), pst.stem(“flying”)
```

Output:

```
(‘run’, ‘cooki', ‘fly’ )
```

11. Explain Lemmatization with the help of an example.

We use stemming and lemmatization to extract root words. However, stemming may not give the actual word, whereas lemmatization generates a meaningful word.
In lemmatization, rather than just removing the suffix and the prefix, the process tries to find out the root word with its proper meaning.
Example: ‘Bricks’ becomes ‘brick,’ ‘corpora’ becomes ‘corpus,’ etc.
Let’s implement lemmatization with the help of some nltk packages.
First, we will import the required packages.

```
from nltk.stem import wordnet
from nltk.stem import WordnetLemmatizer
```

Creating an object for WordnetLemmatizer()

```
lemma= WordnetLemmatizer()
list = [“Dogs”, “Corpora”, “Studies”]
for n in list:
print(n + “:” + lemma.lemmatize(n))
```

Output:

```
Dogs: Dog
Corpora: Corpus
Studies: Study
```

12. What is Parts-of-speech Tagging?

The parts-of-speech (POS) tagging is used to assign tags to words such as nouns, adjectives, verbs, and more. The software uses the POS tagging to first read the text and then differentiate the words by tagging. The software uses algorithms for the parts-of-speech tagging. POS tagging is one of the most essential tools in Natural Language Processing. It helps in making the machine understand the meaning of a sentence.
We will look at the implementation of the POS tagging using stop words.
Let's import the required nltk packages.

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize, sent_tokenize
stop_words = set(stopwords.words('english'))
txt = "Sourav, Pratyush, and Abhinav are good friends."
```

Tokenizing using sent_tokenize

```
tokenized_text = sent_tokenize(txt)
```

To find punctuation and words in a string, we will use **word_tokenizer** and then remove the stop words.

```
for n in tokenized_text:
wordsList = nltk.word_tokenize(i)
wordsList = [w for w in wordsList if not w instop_words]
```

Now, we will use the POS tagger.

```
tagged_words = nltk.pos_tag(wordsList)
print(tagged_words)
```

Output:

```
[('Sourav', 'NNP'), ('Pratyush', 'NNP'), ('Abhinav', 'NNP'), ('good',  'JJ'),
('friends', 'NNS')]
```

13. Explain Named Entity Recognition by implementing it.

Named Entity Recognition (NER) is an information retrieval process. NER helps classify named entities such as monetary figures, location, things, people, time, and more. It allows the software to analyze and understand the meaning of the text. NER is mostly used in NLP, Artificial Intelligence, and Machine Learning. One of the real-life applications of NER is chatbots used for customer support.
Let's implement NER using the **spacy** package.
Importing the spacy package:

```
import spacy
nlp = spacy.load('en_core_web_sm')
```

```
Text = "The head office of Google is in California"
document = nlp(text)for ent in document.ents:
print(ent.text, ent.start_char, ent.end_char, ent.label_)
```

Output:

```
Office 9 15 Place
Google 19 25 ORG
California 32 41 GPE
```

14. How to check word similarity using the spacy package?

To find out the similarity among words, we use word similarity. We evaluate the similarity with the help of a number that lies between 0 and 1. We use the spacy library to implement the technique of word similarity.

```
import spacy
nlp = spacy.load('en_core_web_md')
print("Enter the words")
input_words = input()
tokens = nlp(input_words)
for i in tokens:
print(i.text, i.has_vector, i.vector_norm, i.is_oov)
token_1, token_2 = tokens[0], tokens[1]
print("Similarity between words:", token_1.similarity(token_2))
```

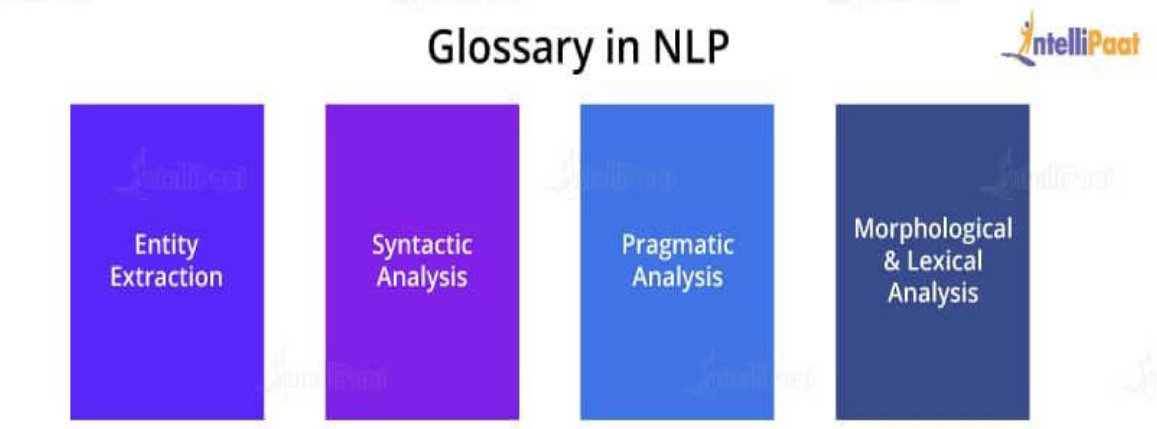
Output:

```
hot True 5.6898586 False
cold True6.5396233 False
Similarity: 0.597265
```

This means that the similarity between the words ‘hot’ and ‘cold’ is just 59 percent.

15. List the components of Natural Language Processing.

The major components of NLP are as follows:



- **Entity extraction:** Entity extraction refers to the retrieval of information such as place, person, organization, etc. by the segmentation of a sentence. It helps in the recognition of an entity in a text.
- **Syntactic analysis:** Syntactic analysis helps draw the specific meaning of a text.
- **Pragmatic analysis:** To find useful information from a text, we implement pragmatic analysis techniques.
- **Morphological and lexical analysis:** It helps in explaining the structure of words by analyzing them through parsing.

16. Define the terminology in NLP.

This is one of the most often asked NLP interview questions.

The interpretation of Natural Language Processing depends on various factors, and they are:

Weights and Vectors

- Use of TF-IDF for information retrieval
- Length (TF-IDF and doc)
- Google Word Vectors
- Word Vectors

Structure of the Text

- POS tagging
- Head of the sentence
- Named Entity Recognition (NER)

Sentiment Analysis

- Knowledge of the characteristics of sentiment
- Knowledge about entities and the common dictionary available for sentiment analysis

Classification of Text

- Supervised learning algorithm
- Training set

- Validation set
- Test set
- Features of the text
- LDA

Machine Reading

- Removal of possible entities
- Joining with other entities
- DBpedia

FRED (lib) Pikes

17. What is Latent Semantic Indexing (LSI)?

Latent semantic indexing is a mathematical technique used to improve the accuracy of the information retrieval process. The design of LSI algorithms allows machines to detect the hidden (latent) correlation between semantics (words). To enhance information understanding, machines generate various concepts that associate with the words of a sentence.

The technique used for information understanding is called singular value decomposition. It is generally used to handle static and unstructured data. The matrix obtained for singular value decomposition contains rows for words and columns for documents. This method best suits to identify components and group them according to their types.

The main principle behind LSI is that words carry a similar meaning when used in a similar context. Computational LSI models are slow in comparison to other models. However, they are good at contextual awareness that helps improve the analysis and understanding of a text or a document.

18. What are Regular Expressions?

A regular expression is used to match and tag words. It consists of a series of characters for matching strings.

Suppose, if A and B are regular expressions, then the following are true for them:

- If {ε} is a regular language, then ε is a regular expression for it.
- If A and B are regular expressions, then A + B is also a regular expression within the language {A, B}.
- If A and B are regular expressions, then the concatenation of A and B (A.B) is a regular expression.
- If A is a regular expression, then A* (A occurring multiple times) is also a regular expression.

19. What is Regular Grammar?

Regular grammar is used to represent a regular language.

A regular grammar comprises rules in the form of **A -> a**, **A -> aB**, and many more. The rules help detect and analyze strings by automated computation.

Regular grammar consists of four tuples:

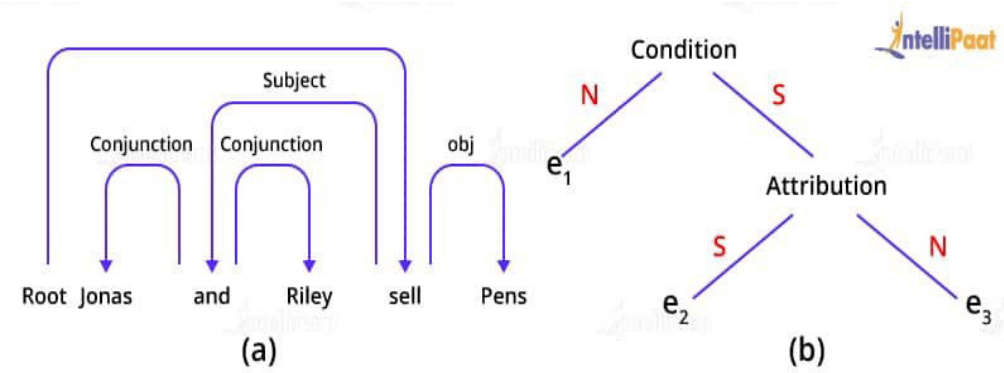
1. ‘N’ is used to represent the non-terminal set.
2. ‘∑’ represents the set of terminals.
3. ‘P’ stands for the set of productions.
4. ‘S € N’ denotes the start of non-terminal.

20. Explain Dependency Parsing in NLP.

Dependency parsing helps assign a syntactic structure to a sentence. Therefore, it is also called syntactic parsing. Dependency parsing is one of the critical tasks in NLP. It allows the analysis of a sentence using parsing algorithms. Also, by using the parse tree in dependency parsing, we can check the grammar and analyze the semantic structure of a sentence.

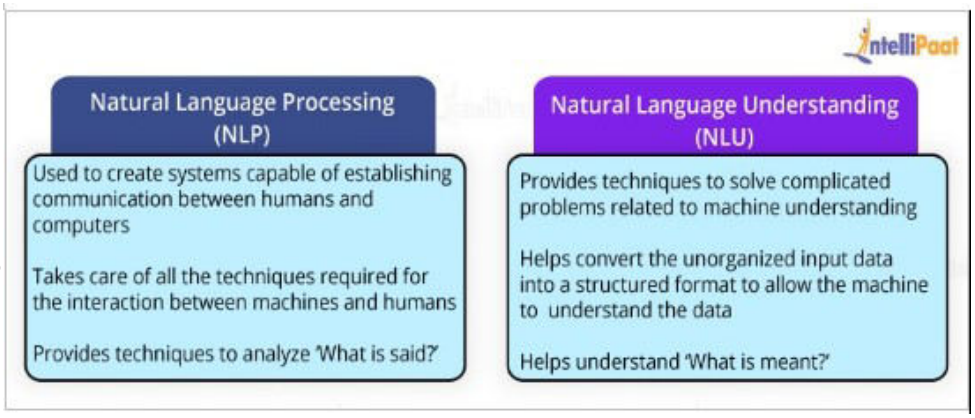
For implementing dependency parsing, we use the spacy package. It implements token properties to operate the dependency parse tree.

The below diagram shows the dependency parse tree:



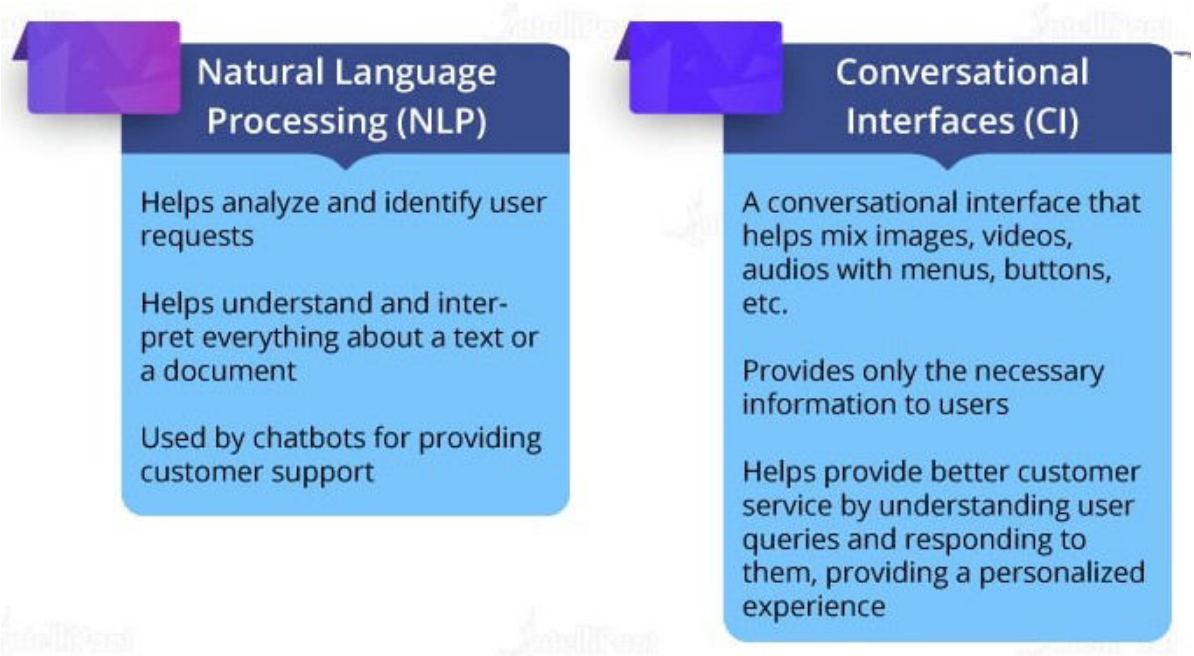
21. What is the difference between NLP and NLU?

The below table shows the difference between NLP and NLU:



22. What is the difference between NLP and CI?

The below table shows the difference between NLP and CI:



23. What is Pragmatic Analysis ?

Pragmatic analysis is an important task in NLP for interpreting knowledge that is lying outside a given document. The aim of implementing pragmatic analysis is to focus on exploring a different aspect of the document or text in a language. This requires a comprehensive knowledge of the real world. The pragmatic analysis allows software applications for the critical interpretation of the real-world data to know the actual meaning of sentences and words.

Example:

Consider this sentence: ‘Do you know what time it is?’

This sentence can either be asked for knowing the time or for yelling at someone to make them note the time. This depends on the context in which we use the sentence.

24. What is Pragmatic Ambiguity?

Pragmatic ambiguity refers to the multiple descriptions of a word or a sentence. An ambiguity arises when the meaning of the sentence is not clear. The words of the sentence may have different meanings. Therefore, in practical situations, it becomes a challenging task for a machine to understand the meaning of a sentence. This leads to pragmatic ambiguity.

Example:

Check out the below sentence.

‘Are you feeling hungry?’

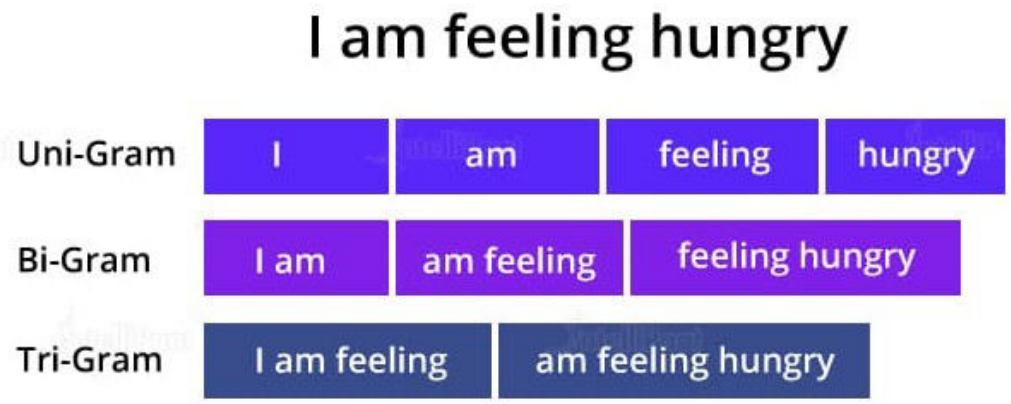
The given sentence could be either a question or a formal way of offering food.

25. What are unigrams, bigrams, trigrams, and n-grams in NLP?

When we parse a sentence one word at a time, then it is called a unigram. The sentence parsed two words at a time is a bigram.

When the sentence is parsed three words at a time, then it is a trigram. Similarly, n-gram refers to the parsing of *n* words at a time.

Example: To understand unigrams, bigrams, and trigrams, you can refer to the below diagram:



Therefore, parsing allows machines to understand the individual meaning of a word in a sentence. Also, this type of parsing helps predict the next word and correct spelling errors.

26. What are the steps involved in solving an NLP problem?

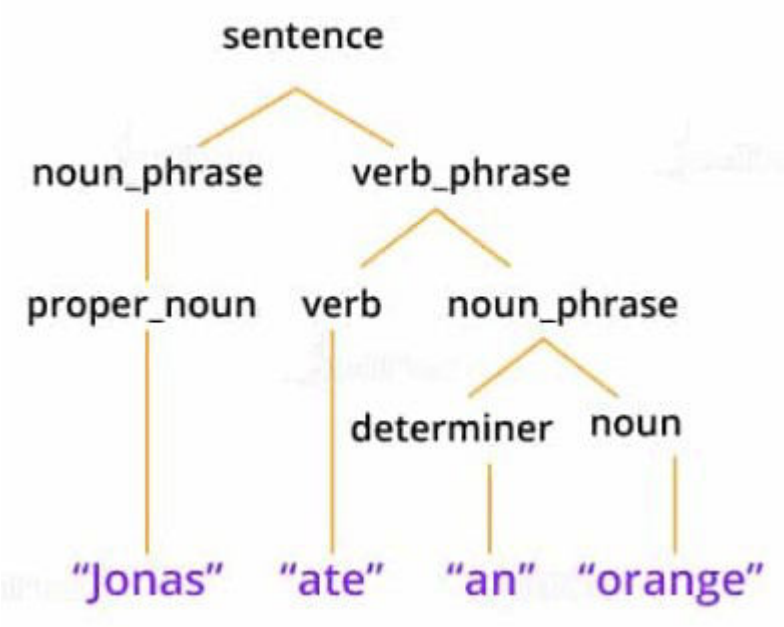
Below are the steps involved in solving an NLP problem:

1. Gather the text from the available dataset or by web scraping
2. Apply stemming and lemmatization for text cleaning
3. Apply feature engineering techniques
4. Embed using **word2vec**
5. Train the built model using neural networks or other Machine Learning techniques
6. Evaluate the model’s performance
7. Make appropriate changes in the model
8. Deploy the model

27. What is Parsing in the context of NLP?

Parsing in NLP refers to the understanding of a sentence and its grammatical structure by a machine. Parsing allows the machine to understand the meaning of a word in a sentence and the grouping of

words, phrases, nouns, subjects, and objects in a sentence. Parsing helps analyze the text or the document to extract useful insights from it. To understand parsing, refer to the below diagram:



In this, 'Jonas ate an orange' is parsed to understand the structure of the sentence.

28. What is Feature Extraction in NLP?

Features or characteristics of a word help in text or document analysis. They also help in sentiment analysis of a text. Feature extraction is one of the techniques that are used by recommendation systems. Reviews such as 'excellent,' 'good,' or 'great' for a movie are positive reviews, recognized by a recommender system. The recommender system also tries to identify the features of the text that help in describing the context of a word or a sentence. Then, it makes a group or category of the words that have some common characteristics. Now, whenever a new word arrives, the system categorizes it as per the labels of such groups.

29. What is precision and recall?

The metrics used to test an NLP model are precision, recall, and F1. Also, we use accuracy for evaluating the model's performance. The ratio of prediction and the desired output yields the accuracy of the model.

Precision is the ratio of true positive instances and the total number of positively predicted instances.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$= \frac{True\ Positive}{Total\ Predicted\ Positive}$$

Recall is the ratio of true positive instances and the total actual positive instances.

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

$$= \frac{True\ Positive}{Total\ Actual\ Positive}$$

30. What is F1 score in NLP?

F1 score evaluates the weighted average of recall and precision. It considers both false negative and false positive instances while evaluating the model. F1 score is more accountable than accuracy for an NLP model when there is an uneven distribution of class. Let us look at the formula for calculating F1 score:

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

https://www.analyticsvidhya.com/blog/2017/01/ultimate-guide-to-understand-implement-natural-language-processing-codes-in-python/

https://medium.com/towards-artificial-intelligence/natural-language-processing-nlp-with-python-tutorial-for-beginners-1f54e610a1a0