1. **What is perplexity?**

In information theory, perplexity is a measurement of how well a probability distribution or probability model predicts a sample. It may be used to compare probability models. A low perplexity indicates the probability distribution is good at predicting the sample.

In natural language processing, perplexity is a way of evaluating language models. A language model is a probability distribution over entire sentences or texts. The perplexity PP of a discrete probability distribution p is defined as

**2. What are the problem with ReLu and how they are solved?**

a. Exploding gradient (Solved by gradient clipping)

b. Dying ReLu — No learning if the activation is 0 (Solved by parametric ReLu or leaky ReLu)

c. Mean and variance of activations is not 0 and 1 (Partially solved by subtracting around 0.5 from activation)

**3. What is the difference between learning latent features using SVD and getting embedding vectors using deep network?**

SVD uses linear combination of inputs while a neural network uses non-linear combination.

**4. What Is The Significance Of Tf-idf?**

Tf–idf or TF IDF stands for term frequency–inverse document frequency. In information retrieval TF IDF is is a numerical statistic that is intended to reflect how important a word is to a document in a collection or in the collection of a set.

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

IDF(W) = ln(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.

**5. What Is Bagging And Boosting In Ensemble Method?**

Bagging is a way to decrease the variance in the prediction by generating additional data for training from dataset using combinations with repetitions to produce multi-sets of the original data.

Boosting is an iterative technique which adjusts the weight of an observation based on the last classification.

**6. 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.

Sentence tokenization

*>>> from nltk.tokenize import sent\_tokenize*

*>>> sent\_tokenize(Para)*

Word tokenisation

*>>> from nltk.tokenize import word\_tokenize*

*>>> word\_tokenize(Para)*

**7. 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.

>>> from nltk.stem import PorterStemmer

>>> pst=PorterStemmer()

>>> pst.stem(“running”), pst.stem(“cookies”), pst.stem(“flying”)

Output:

(‘run’, ‘cooki', ‘fly’ )

**8. 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.

>>> from nltk.stem import wordnet

>>> from nltk.stem import WordnetLemmatizer

>>> lemma= WordnetLemmatizer()

>>> list = [“Dogs”, “Corpora”, “Studies”]

>>> for n in list:

>>> print(n + “:” + lemma.lemmatize(n))

Output:

Dogs: Dog

Corpora: Corpus

Studies: Study

**9. 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.

>>> 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.

>>> tokenized\_text = sent\_tokenize(txt)

>>> for n in tokenized\_text:

>>> wordsList = nltk.word\_tokenize(i)

>>> wordsList = [w for w in wordsList if not w instop\_words]

>>> tagged\_words = nltk.pos\_tag(wordsList)

>>> print(tagged\_words)

Output:

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

Note: Use <https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html> for code mapping. Which is like;

JJ is Adjective, NNP is Proper Noun, singular, NNS is Noun, plural

**10. Explain Named Entity Recognition.**

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.

>>> 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

**11. 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.

**12. What are the common metrics to evaluate langauge models?**

1. GLUE (General Language Understanding Evaluation) and Super GLUE - It is a benchmark based on different types of tasks rather than evaluating a single task. The three major categories of tasks are single-sentence tasks, similarity and paraphrase tasks, and inference tasks.
2. SQuAD (Stanford Question Answering Dataset) - It is a reading comprehension dataset with questions created through crowdsourcing. A passage is given and questions are asked based on the passage. The answer to these questions is a segment of text from the passage.
3. BLEU (BiLingual Evaluation Understudy) - It is a performance metric to measure the performance of machine translation models. It evaluates how good a model translates from one language to another. It assigns a score for machine translation based on the unigrams, bigrams or trigrams present in the generated output and comparing it with the ground truth. It has many problems but it was one of the first methods to assign a score to machine translation models. It always gives a score between 0 and 1.
4. MS MACRO (MAchine Reading COmprehension Dataset) - It is a large scale dataset focused on machine reading comprehension, with Question answering, Passage ranking and Key phrase Extraction tasks.The evaluation of these tasks is done using BLEU and ROGUE(Recall-Oriented Understudy for Gisting Evaluation) metrics.
5. XTREME (Cross-lingual TRansfer Evaluation of Multilingual Encoders) - This evaluation dataset and metrics is the most recent one and is used to evaluate SOTA models for cross-lingual tasks and pre-trained models performance for zero-shot learning.

Below are the generic model merics like

1. Confusion Matrix - precision and recall
2. F1 Score
3. Gain and Lift Charts
4. Kolmogorov Smirnov Chart
5. AUC – ROC
6. Log Loss
7. Gini Coefficient
8. Concordant – Discordant Ratio
9. Root Mean Squared Error

Note: For detail https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/

**13. 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.

**14. What does a NLP pipeline generally consist of?**

Any typical NLP problem can be proceeded as follows:

1. Text gathering(web scraping or available datasets)
2. Text cleaning(stemming, lemmatization)
3. Feature generation(Bag of words)
4. Embedding and sentence representation(word2vec)
5. Training the model by leveraging neural nets or regression techniques
6. Model evaluation
7. Making adjustments to the model
8. Deployment of the model.

**15. What are some popular Python libraries used for NLP?**

Stanford’s CoreNLP, SpaCy , NLTK and TextBlob.

**16. How is feature extraction done in NLP?**

The features of a sentence can be used to conduct sentiment analysis or document classification. For example if a product review on Amazon or a movie review on IMDB consists of certain words like ‘good’, ‘great’ more, it could then be concluded/classified that a particular review is positive.

Bag of words is a popular model which is used for feature generation. A sentence can be tokenized and then a group or category can be formed out of these individual words, which further explored or exploited for certain characteristics(number of times a certain word appears etc).

Latent semantic indexing, word2vec. etc are some other methods other than Bag-of-words.

**17. What is word2vec?**

Word2Vec  embeds words in a lower-dimensional vector space using a shallow neural network. The result is a set of word-vectors where vectors close together in vector space have similar meanings based on context, and word-vectors distant to each other have differing meanings. For example, apple and orange would be close together and apple and gravity would be relatively far. There are two versions of this model based on skip-grams (SG) and continuous-bag-of-words (CBOW).