* NLP Pipeline *

* Break the problem down into sub-problems. Then try to develop a step-by-step procedure to solve them.

(1) Data Acquisition →

→ Available Data (CSV, Txt, Pdf, XLS)
→ Others Data (DB, Internal, API, scrapping)
→ No data (create your own data).

Note → If you have less data then use Data Augmentation.

→ Replace with synonyms.

Ex :- I am a Data scientist.
replaced.
I am a AI Engineer.

→ Bigram Flip

Ex : I am mamta.
Mamta is my name.

→ Back Translate

→ Add additional noise →

Ex :- I am a data scientist.
→ I Love this job. (Extra)

(2) Text Preparation: →

(1) clean up: HTML tags, Emogi, spelling Correction

(2) Basic Preprocessing

(3) Advance preprocessing.

(1) Basic Preprocessing →

Tokenization → Sentence
Tokenization → Word

Steps: →

(1) Stop words removal
(2) Stemming
(3) Lammetization
(4) Removing punctuation (. ! , ? , $)
(5) Lowercase
(6) Language Detection.

(2) Advance Preprocessing: —

(1) Parts of speech tagging.
(2) Parshing
(3) Coreffence resolution.

(3) Feature Engineering →

(1) Text Vectorization
(2) TFIDF, Bag of words, OneHot, Word2Vec, Encoding

(4) Modeling →

(I) Heuristic

(II) ML

(III) DL

(IV) Cloud API (AWS, GCP, Azure)

(5) Evaluation →

(I) Intrinsive (ML - classification test: accuracy, precision, confusion Matrix)

(II) Extrinsive (Host app to productionly uses) Slow ratings

(6) Deployment → (Monitoring, Networking)

* Co-occurrence Vectors →

The role of co-Occurrence Vectors →

(1) Statistical Insight →

Find insights about word frequency and distribution with co-occurrence vectors.

(2) Language Modeling → Improve your understanding of the language structure and underlying semantics

(3) Network Analysis → Uncover relationships between words and concepts with network analysis.

Co-Occurrence Vectors are an essential element of modern NLP and have numerous applications in improving text data analysis.
With the continued application of new techniques and methods, their use will only continue to grow.

* Constructing Co-Occurrence Matrices: →

① Definition → Understand the definition and mathematical representation of co-occurrence matrices.

② Window Size → Choose an appropriate window size to capture the context of the words.

③ Contextual Weighting →

Explore different weighting schemes to emphasize the context that is most informative.

* Application of Co-Occurrence Vectors in NLP →

① Mobile Apps → Discover how co-occurrence vectors are used in NLP mobile apps like predictive texting.

② Customer Support → Learn how customer support teams leverage co-occurrence vectors to identify themes and automate responses.

③ Book Recommendation Systems →

Find out how to build a book recommendation engine using co-occurrence vectors to capture user preferences.

* Doc2Vec →

→ A novel technique for computing vector representation of entire documents.

→ How does it work?

Doc2vec learns vector representations of documents by predicting words within the document.

→ Why is it important?

→ Doc2vec enables us to analyze documents as continuous vectors in multi-dimensional space, opening up new possibilities for next.

* Benefits of Doc2vec in NLP ⇒

(1) Improved Text Classification →

Doc2vec can be used to transform documents into continuous vectors that can be used as input to machine learning models for text classification.

(2) Efficient Document Retrieval →

Doc2vec enables us to search for documents based on semantic similarity, rather than just keyword-based matching, leading to more accurate and efficient document.

(3) Robust Information Extraction →

→ Doc2vec can be used to extract meaningful information from unstructured text data, enabling more accurate and efficient information extraction for NLP tasks.

* Amplifying NLP with Doc2vec: Real-world Applications

① Question Answering Systems →

Doc2vec can be used to create question answering systems that can automatically answer questions based on the semantic content of the document.

② Clinical NLP →

Doc2vec can be used to analyze and extract information from clinical reports, enabling more efficient and accurate analysis of patient medical data.

③ Sentiment Analysis →

Doc2vec can be used to analyze the sentiment of large volumes of text data, enabling more accurate and efficient sentiment analysis for market research and social media monitoring.

Q What are Contextualized Document Representations?

Ans → Contextualized document representations are document vectors that capture not only the meaning of the words in the document, but also their contextual relationships with other words in the document.

* ELMO: → (A Powerful Contextualized Encoder)

ELMO is a state-of-the-art contextual encoder that can generate contextualized embeddings for words and sentences, enabling more accurate and efficient NLP tasks.

BERT: (A Transformer-based Encoder)

BERT is a powerful transformer-based encoder that can generate contextualized embeddings for words and sentences, enabling highly accurate NLP tasks such as text classification and question answering.

* Unraveling the challenges of Implementing Doc2Vec

(1) Data Preprocessing challenges →

Preparing the large volume of text data for Doc2vec training can be challenging, as it requires extensive cleaning, normalization, and tokenization

(2.) Hyperparameter Tuning →

→ Finding the optimal combination of hyperparameters for the Doc2vec model can be challenging, as it requires extensive experimentation and Tunning.

(3.) Model Complexity →

the Doc2vec model can be computationally intensive to train, especially when dealing with very large volumes of text data.

*

* Text Blob → (Simplified Text Preprocessing)

→ Text Blob is a Python library for preprocessing textual data.
→ It provides a simple API for dividing into common (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

* Text Blob is an exceptional library that makes NLP tasks a breeze. It's features allow for comprehensive text analysis with minimal code, making it a must-have for today's NLP workflow.

* Tokenization →

Text Blob's tokenizer splits documents into sentences or words efficiently. Customize your tokenizer to suit the task at hand.

* Word Frequencies →

Find out which words appear the most frequently. Create bar charts and word clouds to visualize your findings.

* N-gram Creation ⇒

Create word n-grams to uncover import patterns in your document. Text Blob allows for trigram and bigram generation.

Stopwords →

Remove noise from your documents. TextBlob makes it easy to remove stopwords and focus only on important words.

* Language Translation →

Translate your text into another language with ease. Carve out your niche in global markets easily and effectively.

* Language Detection →

Automatically detect which language your content is written in. Great for multi-language websites and document repositories.

* NLTK →

Natural Language toolkit (NLTK) is a powerful open-source toolkit used to process human language data. It provides a range of functions that can help with text processing, manipulation and analysis.

NLTK's features →

-> NLTK has a wide range of features that are useful for various language processing tasks.

-> It also offers an extensive corpus of text data, ranging from Shakespeare's plays to the Brown Corpus of American English

* Applications of NLTK →

NLTK has been used in a vast array of applications, from chatbots and machine translation to sentiment analysis and spam detection.

→ It is also widely used in academic research, as it provides a reliable and standardized toolkit for processing and analyzing human language data.

* Tokenization and Text Preprocessing with NLTK

* Tokenization → Tokenization refers to breaking up text into smaller units or tokens, such as words or phrases.

NLTK provides several tokenization tools, such as the word_tokenize() function and the RegexpTokenizer class.

Text Preprocessing Practical

(1) How to perform text Preprocessing.

-> HTML tags remove
-> Punctuation remove
-> Emoji Handle
-> Lammatization
-> Steaming
-> Lower Case

(2.) Text Representation & Word Embedding
(↓ Text Vectorization)

-> One Hot Encoding
-> BOW (Bag of words)
-> TFIDF
-> Word2_vec (convert text to vector (number))

(3.) Project -> Sentiment Analysis using ML Approach.

-> ML Algorithm used.

Gensim →

Gensim is a Python library for topic modeling, document similarity analysis, and word embeddings. It provides a user-friendly interface for natural language processing tasks.

→ It is an open-source library for unsupervised semantic modeling of text.

* Purpose and Uses →

Gensim allows users to analyze and extract meaning from large collections of text documents.

(1) Word Embeddings →

Gensim provides tools for training and using word embeddings, such as Word2vec and Fast text.

(2) Topic Modeling →

With Gensim, it's easy to discover latent topics in a collection of documents using techniques like Latent Dirichlet Allocation (LDA).

(3) Document Similarity →

Gensim allows you to measure the similarity between documents, which is useful for tasks like recommender systems.

Latent Dirichlet Allocation (LDA) →

Gensim supports LDA for discovering topics in a corpus and associating them with relevant documents.

Latent Semantic Analysis (LSA) →

LSA is another technique supported by Gensim for finding hidden semantic structures in a document collection.