Avikumar Talaviya

Summary

This article provides a comprehensive guide to the fundamentals of

The Fundamentals of Natural Language Processing: A Beginner's Guide

Learn the fundamentals of Natural Language Processing in this article to master NLP techniques

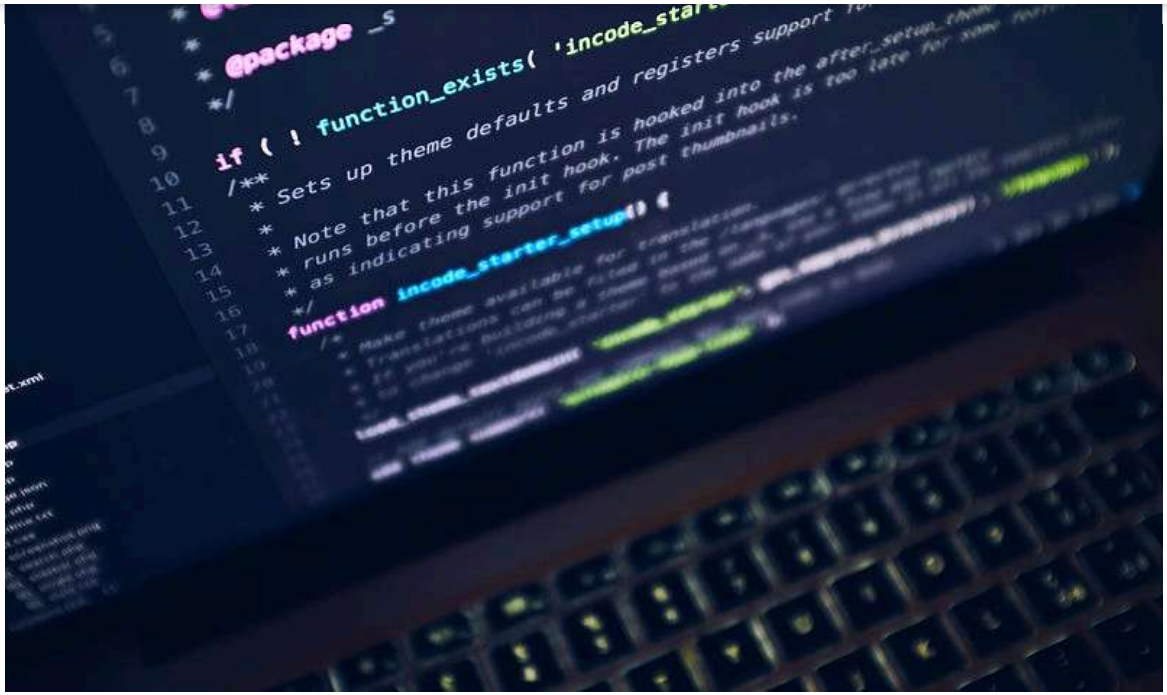


Photo by Luca Bravo on Unsplash

NLP is a sub-field of machine learning which leverages analysis, generation, and understanding of human languages to derive meaningful insights from it.

Natural language processing is becoming popular as Large language models (LLMs) are growing and used widely in the market. Having foundation knowledge of NLP concepts and techniques can help you become an NLP data scientist, NLP engineer, or distinguished ML engineer to stand out in the job market.

as a data scientist.

Note: *This article was originally published on DataKwery. please check out the original article link* here

Table of contents:

- **Text pre-processing**
- **Text feature extraction**
- **Text sort**
- **Named entity recognition**
- **Parts-of-speech tagging**
- **Text generation**
- **Text-to-speech and speech-to-text techniques**
- **Conclusion**

Text pre-processing:

Pre-processing techniques such as tokenization, stemming, and lemmatization, can help to convert raw text into a format that can be easily analyzed.

Tokenization

Let's look at the below code to tokenize the given sentence using the NLTK library.

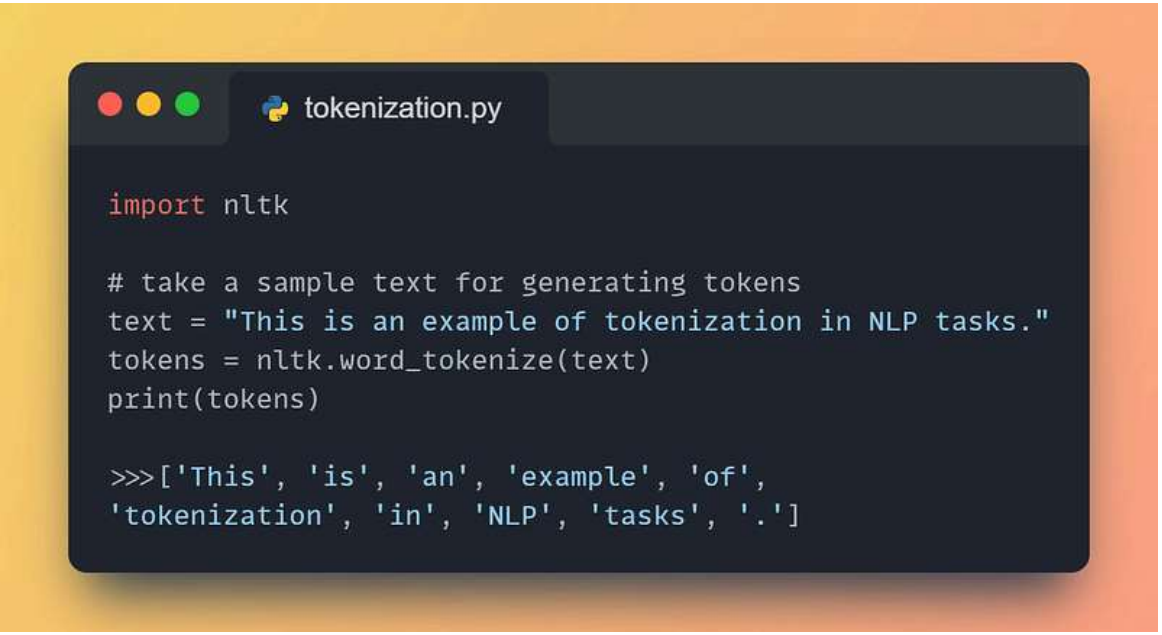


Image by author

Stemming

- It is the process of reducing words to their base or root form. This can be useful in classification or information retrieval tasks

The below code demonstrates the stemming using the PorteStemmer method of the NLTK library.

Stemming.py

```
import nltk
from nltk.stem import PorterStemmer

# create stemmer object usign PorterStemmer method of nltk library
stemmer = PorterStemmer()
words = ["run", "runner", "ran", "runs", "easily", "fairly"]
stemmed_words = [stemmer.stem(word) for word in words]
print(stemmed_words)

>>>['run', 'runner', 'ran', 'run', 'easili', 'fairli']
```

Image by author

Lemmatization

- Lemmatization is the process of reducing a word to its base or root form, which is known as the lemma. It is a more sophisticated version of stemming, as it takes into account the context and the part of speech of the word.

Here is an example of lemmatization using the `WordNetLemmatizer` class from the `nltk` library in Python:

```python
import nltk
from nltk.stem import WordNetLemmatizer

# Download the wordnet package
nltk.download('wordnet')

# Define the word to be lemmatized
word = "running"

# Create an instance of the WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

# Lemmatize the word
lemma = lemmatizer.lemmatize(word, pos='v')
print(lemma)
```

Image by author

Text feature extraction:

Techniques such as Vocabulary/bag-of-words, n-grams, count vectorization, and word embeddings, to represent text as numerical features for use in machine learning models.

- Vocabulary in NLP refers to the set of unique words or tokens in a given text or corpus.

Here is an example of creating a vocabulary in Python using the NLTK library:

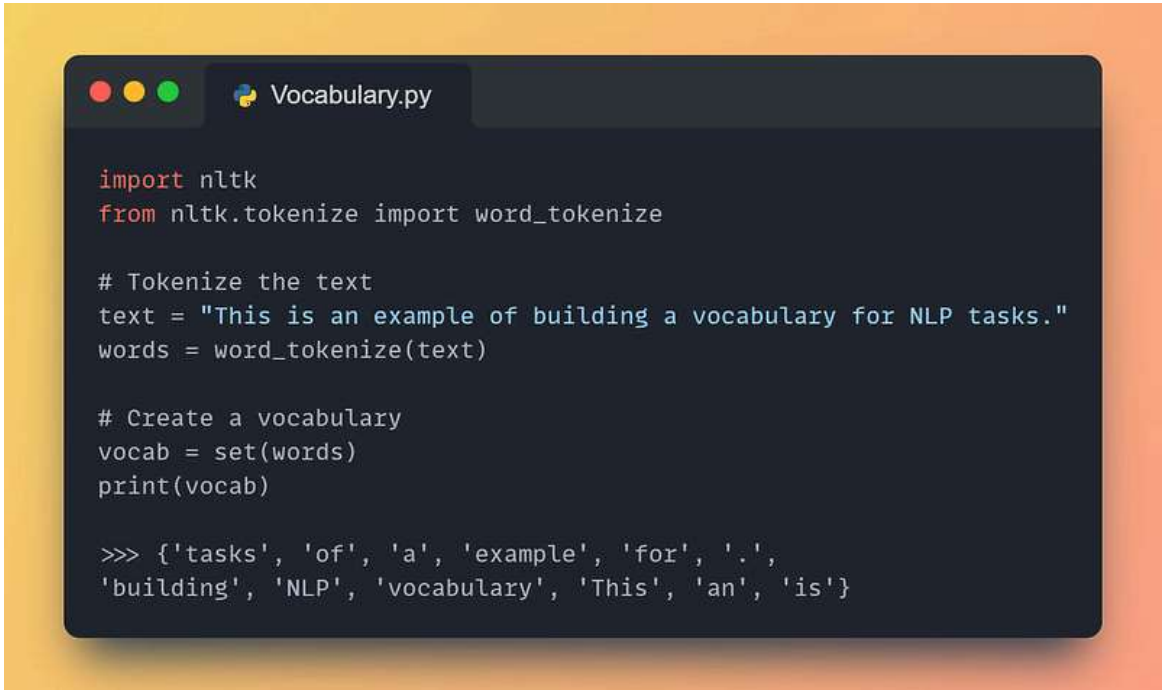


Image by Author

N-grams

- An n-gram is a contiguous sequence of n items from a given sample of text or speech, where n can be any positive integer. In NLP, n-grams are often

Here is an example of creating n-grams in Python using the NLTK library:

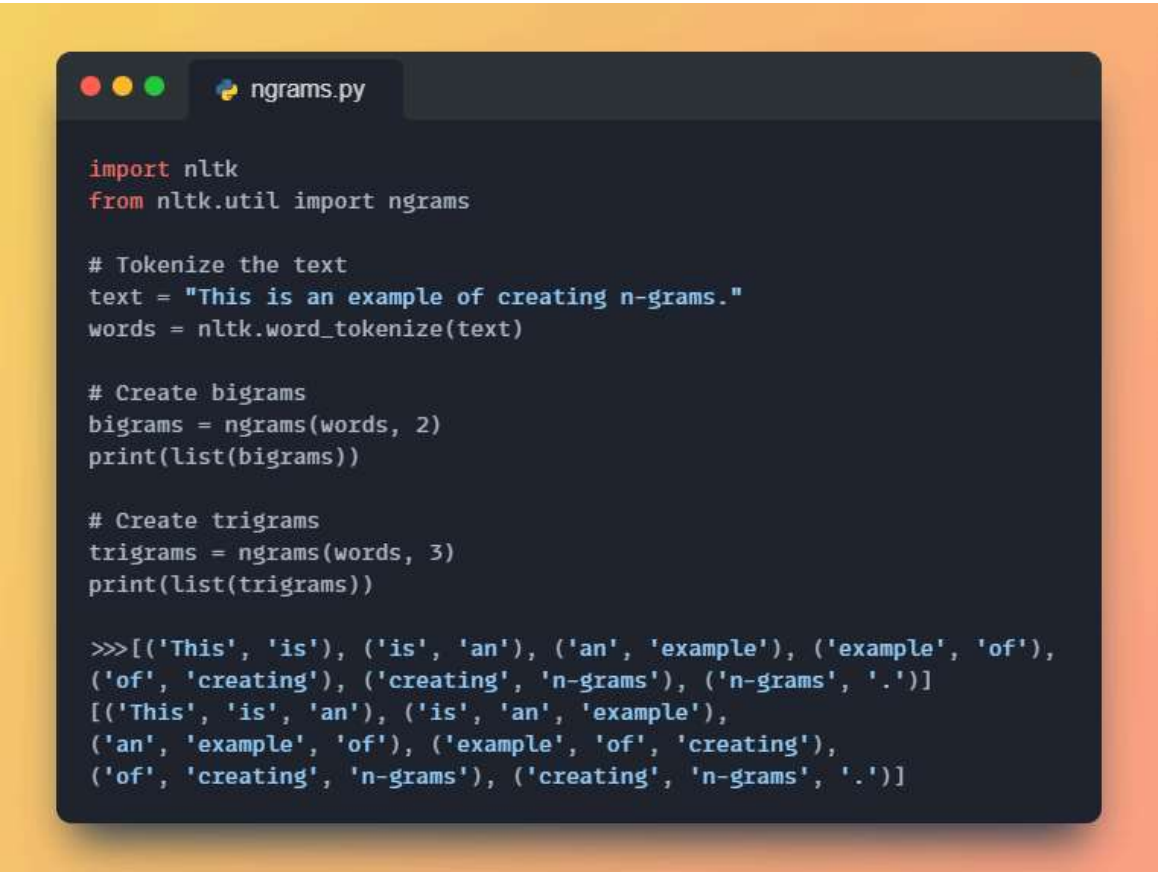


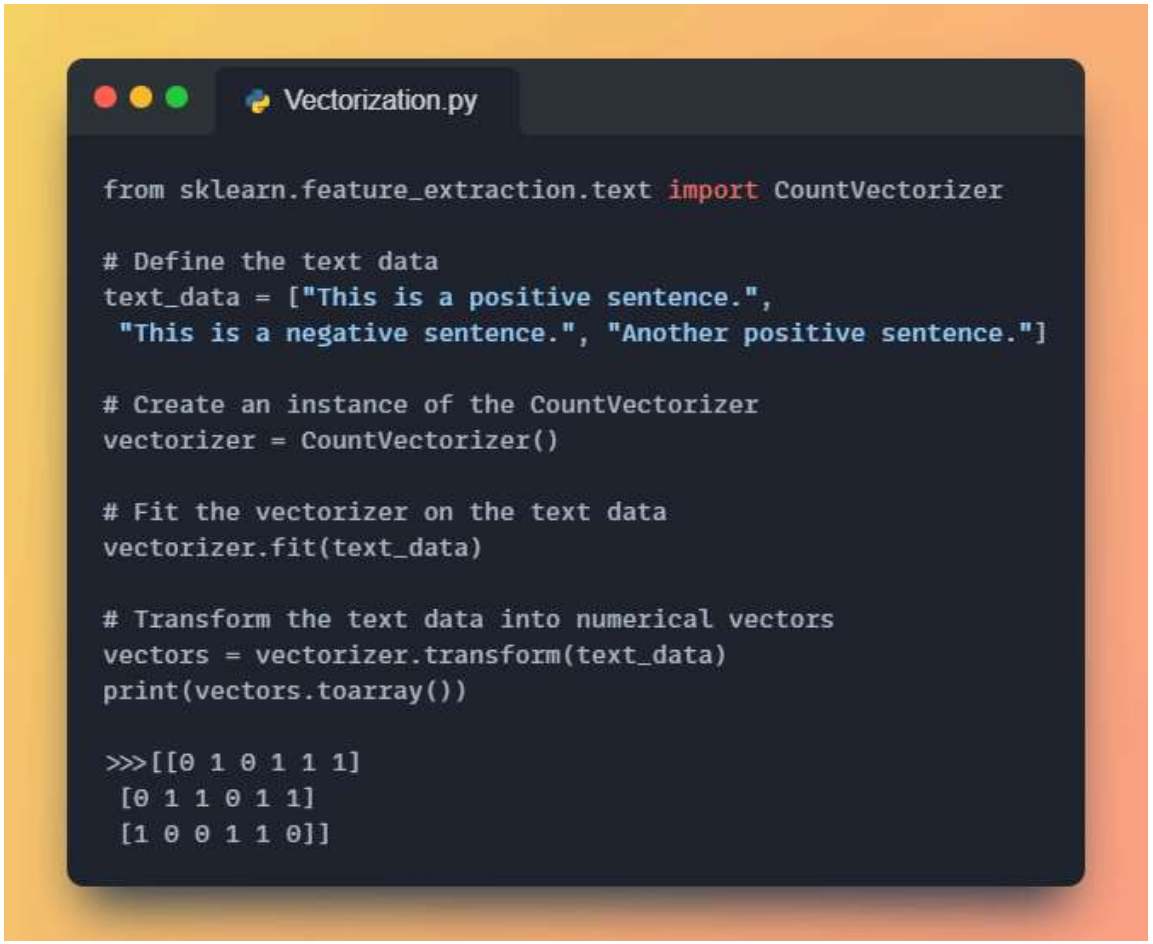
Image by Author

Count vectorization

- Text vectorization is the process of converting text data into numerical vectors, which can be used as input for machine learning models. One of

and word order but keeping track of the number of occurrences of each word.

Here is an example of text vectorization using the bag-of-words approach in Python using the `CountVectorizer` class from the `sklearn` library:



Text sort:

Techniques for classifying text into predefined categories, such as sentiment analysis and spam detection.

- Text sorting is one of the most important NLP tasks of all, which involves assigning predefined categories or labels to a given piece of text.

An example of text classification in Python using the scikit-learn library is:

```python
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline

# load the dataset
data = fetch_20newsgroups()

# Create the modeling pipeline
model = make_pipeline(TfidfVectorizer(), MultinomialNB())
model.fit(data.data, data.target)

# predict the labels of text samples
labels = model.predict(["This is some example text"])
print(labels)
```

Image by Author

In this example, we are using ***20newsgroups*** data, which contains 18000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups. The pipeline vectorizes the text using TfidfVectorizer, then trains a Naive Bayes classifier using the vectorized data and returns the predicted label.

Named entity recognition:

people, organizations, and locations.

- An entity can be thought of as a category type present in a given text.
- For example, the name of a certain personality, the name of an organization, location, etc.

Here is a code example to explain Named entity recognition:

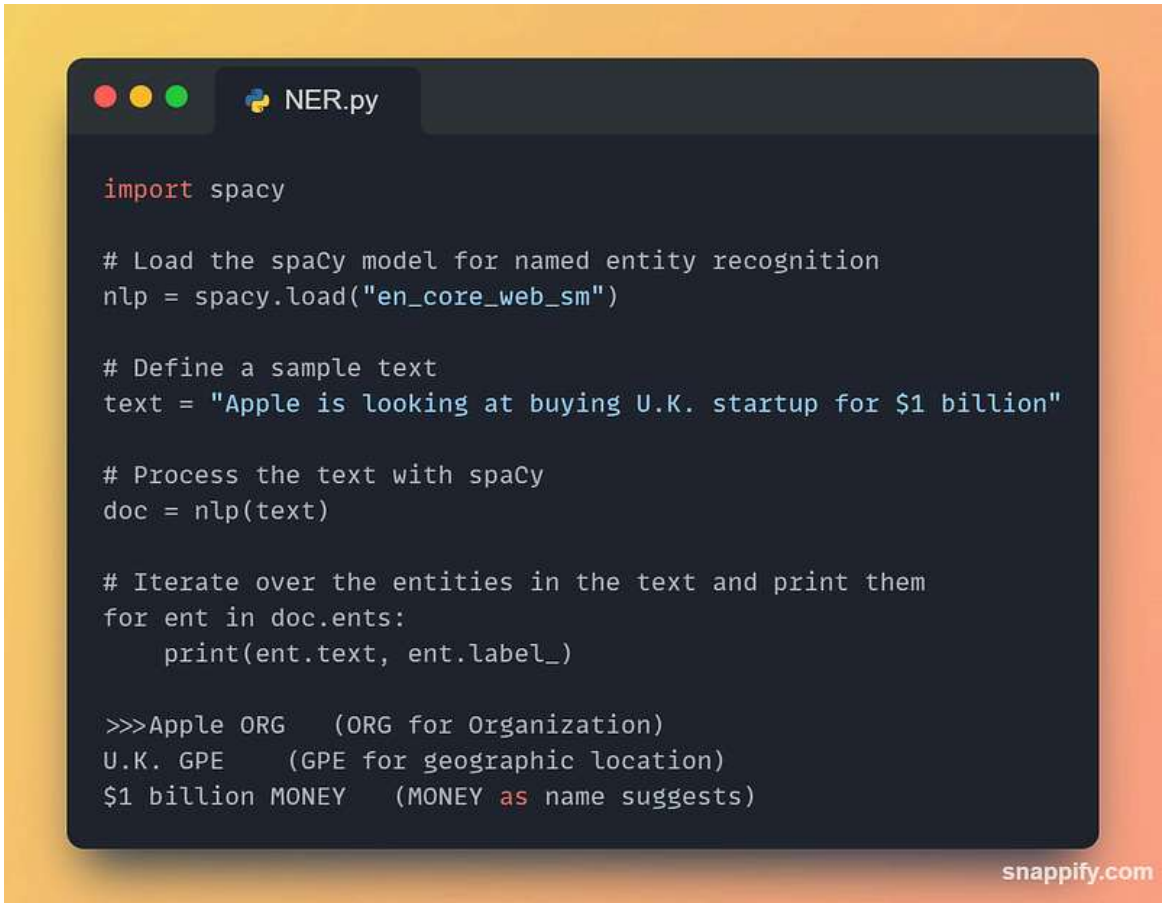


Image by Author

Techniques for identifying the parts of speech of words in a sentence, such as nouns, verbs, and adjectives.

NLTK library of python has a method called ‘pos_tag’ which allows tagging parts of speech with just one line of code.

Let’s take a code example of POS tagging using python programming language:

```
def sentense_pos_tagger(text):

    # tokenize the text
    sample = word_tokenize(text)
    tags = nltk.pos_tag(sample)

    return tags

sentense_pos_tagger(ndf['cleaned_text'][0])

------------------------[Output]--------------------------
[('i', 'NN'),
 ('read', 'VBP'),
 ('them', 'PRP'),
 ('in', 'IN'),
 ('change', 'NN'),
 ('in', 'IN'),
 ('meaning', 'VBG'),
 ('the', 'DT'),
 ('history', 'NN'),
 ('of', 'IN'),
 ('slavery', 'NN')]
```

Image by author

Text generation:

Techniques for generating new text based on a given input, such as machine translation and text summarization.

given input or model. It is a popular area of research in Natural Language Processing (NLP) and has numerous applications such as chatbots, content creation, and language translation.

There are several techniques for text generation, including:

- **Markov Chain:** It is a statistical model that predicts the next word based on the probability distribution of the previous words in the text. It generates new text by starting with an initial state, and then repeatedly sampling the next word based on the probability distribution learned from the input text.
- **Sequence-to-Sequence (Seq2Seq) Model:** It is a deep learning model that consists of two recurrent neural networks (RNNs), an encoder, and a decoder. The encoder takes the input text and produces a fixed-length vector representation, while the decoder generates the output text based on the vector representation.
- **Generative Adversarial Network (GAN):** It is a deep learning model that consists of two neural networks, a generator, and a discriminator. The generator produces new text samples, while the discriminator tries to distinguish between the generated text and the real text. The two networks are trained in an adversarial manner, where the generator tries to produce more realistic text, while the discriminator tries to become better at recognizing fake text.
- **Transformer-based Models:** Transformer models are a type of neural network architecture designed for NLP tasks, such as text

perform well on text generation tasks as well.

Text-to-Speech and Speech-to-Text:

Techniques for converting speech to text and text to speech.

Text-to-Speech (TTS) and Speech-to-Text (STT) are two important applications of Natural Language Processing (NLP).

- **Text-to-Speech (TTS):** TTS is a technology that allows computers to generate human-like speech from written text. The goal of TTS is to produce speech that is natural, expressive, and matches the intonation, rhythm, and prosody of human speech as closely as possible. TTS systems typically consist of two components: a text analysis module that analyzes the input text, and a speech synthesis module that converts the analyzed text into speech.
- **Speech-to-Text (STT):** STT is a technology that allows computers to transcribe spoken words into written text. STT systems are used in a wide range of applications, including voice-controlled virtual assistants, dictation software, and automatic speech recognition (ASR) systems. STT systems typically use acoustic models and language models to transcribe speech into text.

There are other advanced concepts like Attention-based models, Transformers, and BERT which are widely used in various NLP tasks. Attention-based models and transformers are bringing a new paradigm in the

generative AI in general, and in particular generative text-based applications like ChatGPT are increasingly becoming popular on day to day basis.

Conclusion:

In the conclusion, Text-based data is everywhere and businesses of all sizes generate tons of text data. To analyze, interpret and gain meaningful full insights out of raw text data, beginner data scientists must learn NLP techniques to succeed in a career.

To learn NLP techniques and concepts in-depth make sure you check out various recommended courses attached on the sidebar of this web page.

Naturallanguageprocessing | Artificial Intelligence | Machine Learning | Data Science | Data Visualization

Muneeb S. Ahmad

in-depth learning here.

22 min read

Jo Wang

Deep Learning Part 5 -How to prevent overfitting

Techniques used to prevent overfitting in deep learning models:

4 min read

Ali Shafique

LLM Prompt Engineering Techniques and Best Practices

What is Artificial Intelligence?

10 min read

Amit Yadav

Langchain vs Huggingface

If you think you need to spend $2,000 on a 120-day program to become a data scientist, then listen to me for a minute.

10 min read

Abdur Rahman

7 min read

Mdabdullahalhasib

A Complete Guide to Embedding For NLP & Generative AI/LLM

Understand the concept of vector embedding, why it is needed, and implementation with LangChain.

11 min read