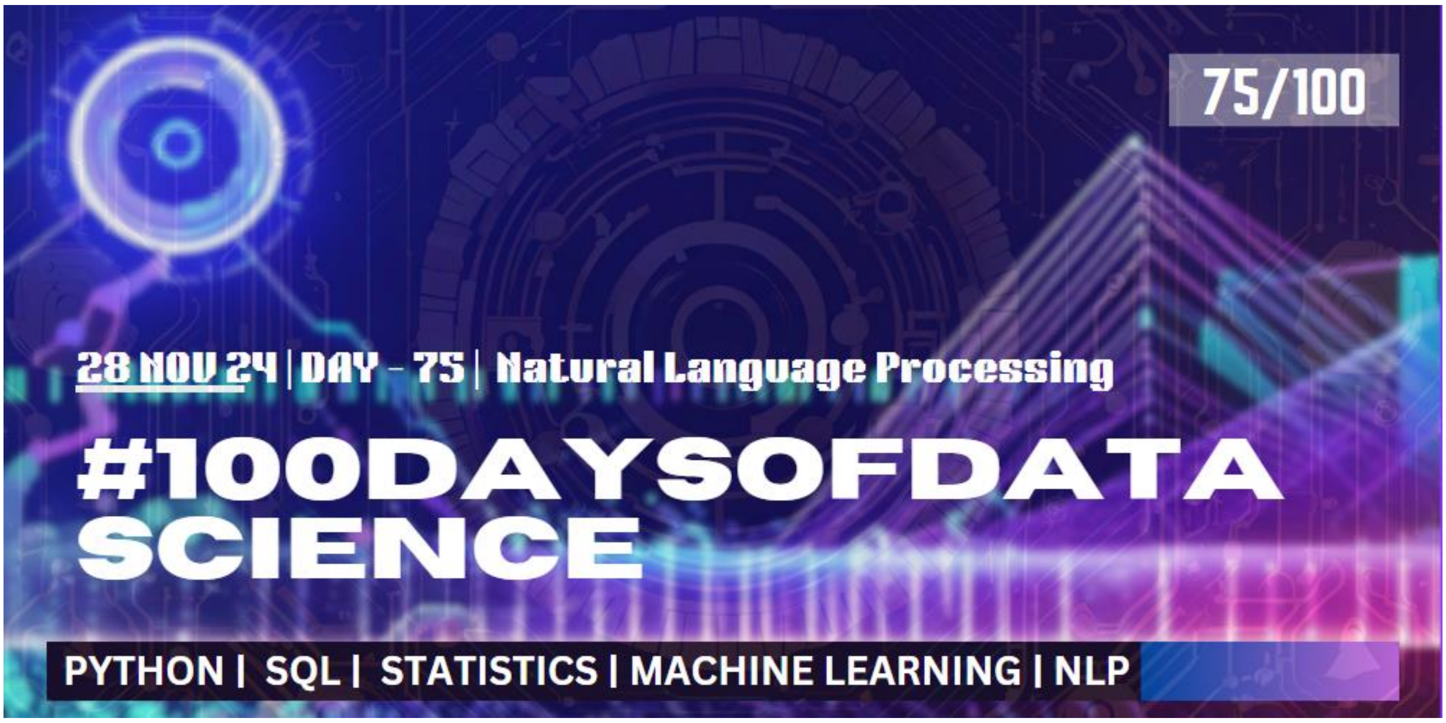
75/100
28 NOV 24 | DAY - 75 | Natural Language Processing
#100DAYSOFDATA SCIENCE
PYTHON | SQL | STATISTICS | MACHINE LEARNING | NLP

N-Gram – [Unigrams, Bigrams, Trigrams]

What is N-gram?
N-gram is a text representation technique used in Natural Language Processing (NLP) that breaks down a sequence of text into contiguous sequences of *n* words (or tokens). It extends the Bag of Words (BoW) model by preserving local word order, enabling the capture of contextual information in the text. N-grams are widely used for text classification, sentiment analysis, and language modeling.

Key Aspects of N-grams
1. Unigrams: Single words or tokens.
- Represent text as individual words without considering any context or sequence.
- **Example**:
  Input Sentence: "Jim and Pam traveled by bus."
  Unigrams: {Jim, and, Pam, traveled, by, bus}
  Representation:
  1,1,1,1,1,11, 1, 1, 1, 1, 11,1,1,1,1,1

2. Bigrams: Pairs of consecutive words.
- Capture basic word order and relationships between adjacent words.
- **Example**:
  Input Sentence: "Jim and Pam traveled by bus."
  Bigrams: {Jim and, and Pam, Pam traveled, traveled by, by bus}
  Representation:
  1,1,1,1,11, 1, 1, 1, 11,1,1,1,1

3. Trigrams: Triplets of consecutive words.
- Provide deeper context by considering three-word sequences.
- **Example**:
  Input Sentence: "Jim and Pam traveled by bus."
  Trigrams: {Jim and Pam, and Pam traveled, Pam traveled by, traveled by bus}
  Representation:
  1,1,1,11, 1, 1, 11,1,1,1

Advantages of N-grams:
1. **Preservation of Local Context**:

    - Unlike BoW, n-grams retain sequential information, making it easier to understand local dependencies.
2. **Flexibility Across Tasks**:
    - Unigrams are effective for basic text classification, while bigrams and trigrams capture phrase-level meaning for tasks like sentiment analysis and speech recognition.
3. **Compatibility with Machine Learning**:
    - N-grams can be directly used as features in machine learning models like Naive Bayes or SVMs.

Common Libraries for N-gram Implementation:

1. **Scikit-learn**:
    - Provides CountVectorizer and TfidfVectorizer for efficient n-gram generation.
2. **NLTK**:
    - Offers utilities for tokenizing text and creating custom n-grams.
3. **Gensim**:
    - Useful for topic modeling and text vectorization, including n-grams.

When to Use N-grams:
Ideal For:

- Sentiment analysis, where bigrams or trigrams can help identify phrases like "not good" or "very happy."
- Language modeling to predict the next word in a sequence (e.g., autocomplete systems).
- Document similarity and classification tasks requiring context capture.

Avoid For:

- Very large corpora without preprocessing, as n-grams can lead to high-dimensional sparse matrices.
- Tasks requiring deep semantic understanding or long-range dependencies, such as machine translation.

Strengths and Limitations of N-grams:

Strengths	Limitations
Retains local word order	Loses long-range dependencies
Enhances context representation	High-dimensional feature space
Improves predictive power	Computationally expensive for larger *n*

Selecting N-grams for Analysis:
Applications:

- **Spam Detection**: Bigrams like "win free" or "click here" are common spam indicators.
- **Sentiment Analysis**: Trigrams such as "not at all good" improve context understanding.
- **Text Summarization**: N-grams help identify recurring phrases in documents.

Considerations:

- Choose lower *n* (e.g., unigram or bigram) for smaller datasets to avoid overfitting.
- Use techniques like stop-word removal and TF-IDF weighting to reduce dimensionality.

N-grams bridge the gap between simple word-based models and complex deep learning techniques by incorporating word order and local context into text representation. Despite their limitations, n-grams are a cornerstone of classical NLP tasks and remain a valuable tool for feature extraction in various text analysis applications.
By leveraging unigrams, bigrams, and trigrams effectively, we can capture key insights from textual data, enabling smarter and more contextual decisions in NLP models. 🎉