**Text Preprocessing and Feature Extraction using Scikit-Learn**

**1. Introduction**

Text preprocessing is a crucial step in Natural Language Processing (NLP) to clean and transform text into a structured format. This is essential for building machine learning models, including Large Language Models (LLMs) like ChatGPT.

In this task, we perform text preprocessing and feature extraction using two methods:

* **CountVectorizer** (Word Count Matrix)
* **TfidfVectorizer** (Term Frequency-Inverse Document Frequency)

**2. Text Preprocessing Steps**

Before converting text into numerical vectors, we follow these preprocessing steps:

**Lowercasing:** Convert all text to lowercase.

**Removing Punctuation and Special Characters:** Remove symbols, numbers, and unnecessary characters.

**Tokenization:** Split text into individual words.  
 **Removing Stopwords:** Remove common words (e.g., "the," "is," "and") that don’t add value.  
**Stemming/Lemmatization (Optional):** Reduce words to their base form (e.g., "running" → "run").

**3. Feature Extraction Methods**

Once the text is cleaned, we transform it into numerical form using the following methods:

**3.1 CountVectorizer (Word Count Matrix)**

CountVectorizer converts text into a matrix of token (word) counts.

**Example Output of CountVectorizer:**

|  | **hello** | **today** | **win** | **lottery** | **meet** | **later** | **congratulations** | **gift** | **card** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Msg 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Msg 2 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| Msg 3 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| Msg 4 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| Msg 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

* Each row represents a document (text message).
* Each column represents a word (feature).
* Values indicate how many times a word appears in that document.

**3.2 TfidfVectorizer (TF-IDF Matrix)**

TF-IDF assigns importance scores to words based on their frequency in a document and across multiple documents.

**Example Output of TfidfVectorizer:**

|  | **hello** | **today** | **win** | **lottery** | **meet** | **later** | **congratulations** | **gift** | **card** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Msg 1 | 0.45 | 0.67 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Msg 2 | 0 | 0 | 0.78 | 0.78 | 0 | 0 | 0 | 0 | 0 |
| Msg 3 | 0 | 0 | 0 | 0 | 0.64 | 0.64 | 0 | 0 | 0 |
| Msg 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0.77 | 0.77 | 0.77 |

* Values are TF-IDF scores instead of raw counts.
* Higher values mean the word is more important in that document.

**4. Importance of Feature Extraction in AI Models**

* These methods **convert text into numerical format**, which is essential for machine learning models.
* Techniques like **CountVectorizer and TfidfVectorizer** are used in **spam detection, sentiment analysis, and chatbots**.
* Large Language Models (LLMs) like ChatGPT use **more advanced vectorization techniques** beyond TF-IDF.

**5. Conclusion**

By performing text preprocessing and feature extraction, we take the first step in building AI models that understand human language. This structured data can now be used for **classification, clustering, and deep learning models**.

**Next Steps:**

Would you like to use this vectorized text for training a machine learning model like Naive Bayes for spam detection?