Sentence tokenizer 🡪 Sentence tokenization is **the process of splitting text into individual sentences**. nltk.sent\_tokenize(data)

Word tokenizer 🡪 Sentence tokenization is **the process of splitting text into individual sentences**. nltk.word\_tokenize(data)

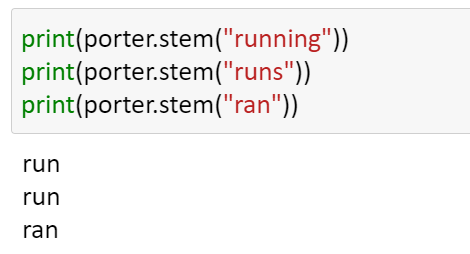
Parts of Speech tags 🡪 nltk.pos\_tag()

Stopwords 🡪 stopwords.words(‘language’)

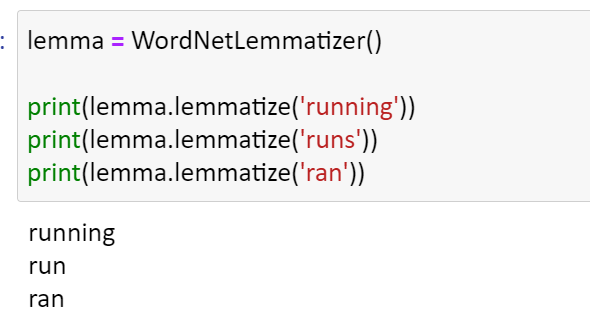
Punctuation 🡪 string.punctuation

**Stemming 🡪 will simply chop up the words (e,es,ing etc)**

1. Porter Stemmer
2. Lancaster Stemmer
3. Snowball Stemmer

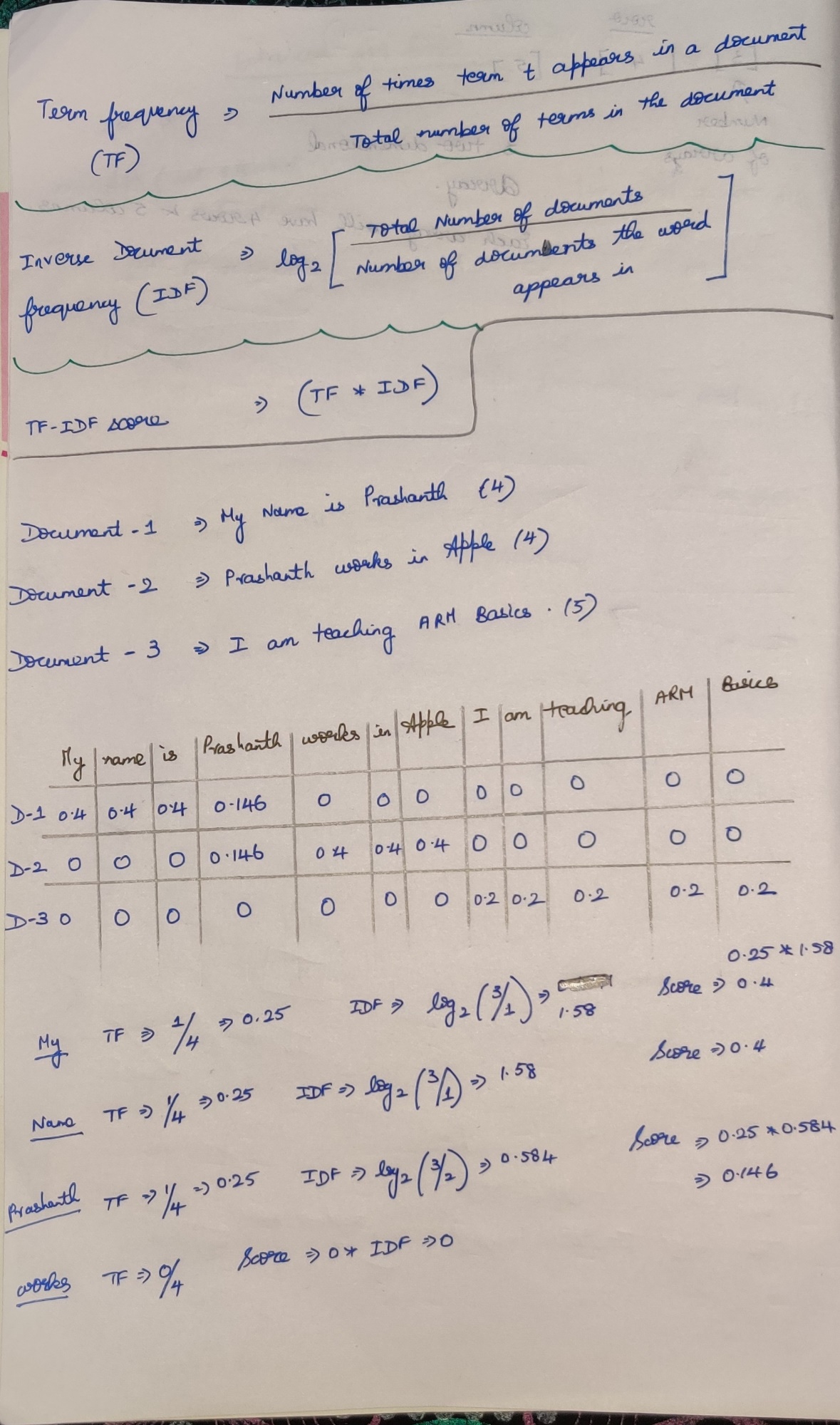


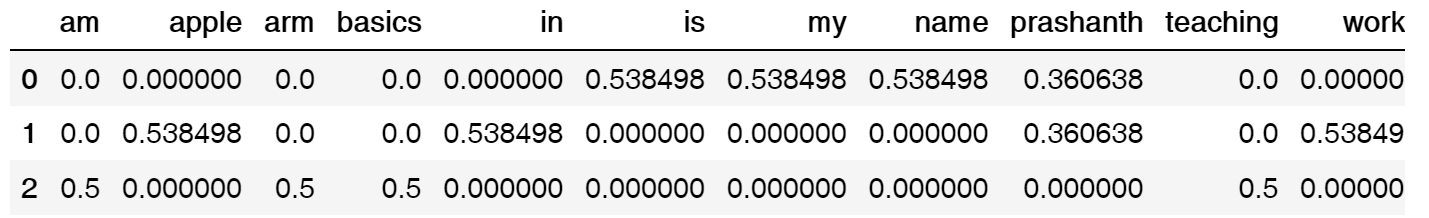
**Lemmatization 🡪**



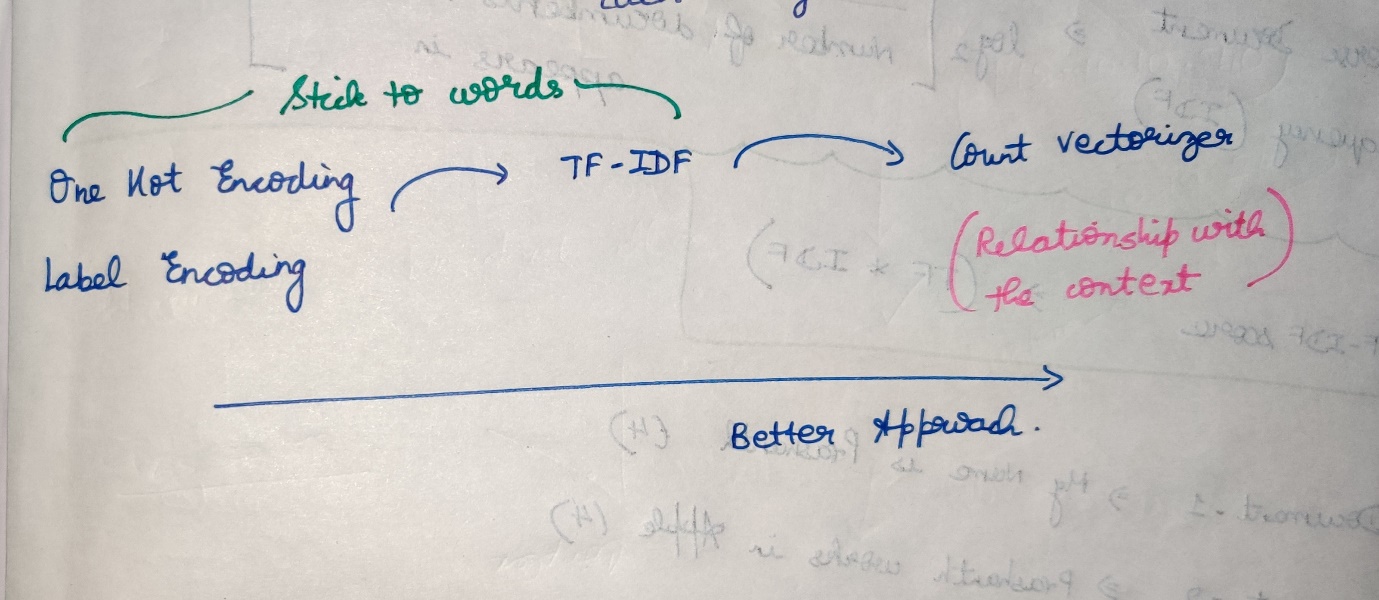
Named Entity Recognition(NER)

# **Tf-Idf (Term frequency–Inverse document frequency)**





The above table is only correct



TF-IDF simply focuses on a particular word.  
It never understands the contextual meaning like how one word is related to other.

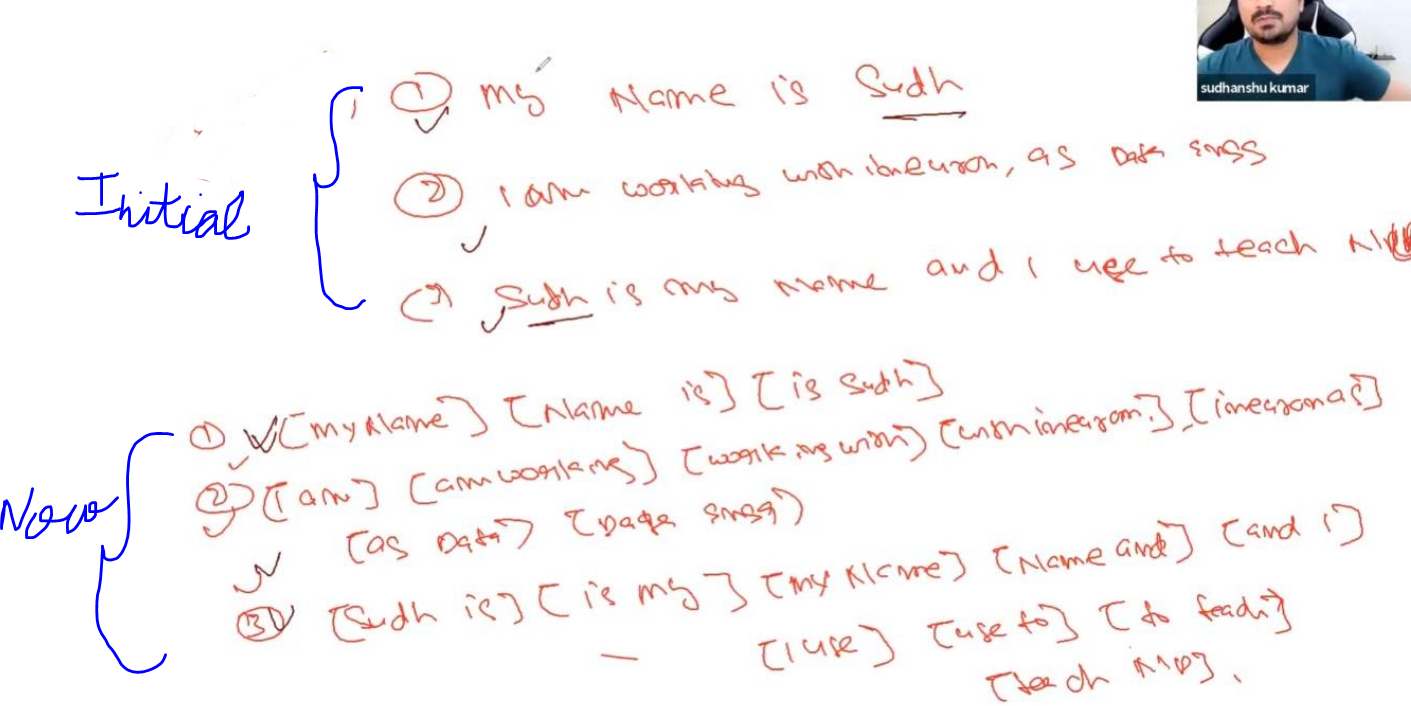
To over-come Count vectorization comes into picture

# **Count Vectorization**

## Bag of Words

## N-gram Model

In bigram and multigram, there exists a relationship between previous text and the next text.



Initial 🡪 Sudhanshu is repeated twice  
Now 🡪 [is Sudhanshu] and [Sudhanshu is] repeated only once.

