```
#TF_IDF
from sklearn.feature_extraction.text import TfidfVectorizer
import pandas as pd

def calculate(documents):
  vectorizer= TfidfVectorizer()
  tfidf_matrix= vectorizer.fit_transform(documents)
  terms = vectorizer.get_feature_names_out()
  df=pd.DataFrame(tfidf_matrix.toarray(), columns=terms)
  return df

documents=["Hey there this is dee", "hello dee", " Hey i m surprised"]
print(calculate(documents))
```

```
        dee     hello       hey        is  surprised     there      this
0  0.373022  0.000000  0.373022  0.490479   0.000000  0.490479  0.490479
1  0.605349  0.795961  0.000000  0.000000   0.000000  0.000000  0.000000
2  0.000000  0.000000  0.605349  0.000000   0.795961  0.000000  0.000000
```

```
#ngrams
from nltk import ngrams

sentence= input("enter a snetence")
n= int(input("enter a number fo rn grams"))
n_grams=ngrams(sentence.split(),n)
for i in n_grams:
  print(i)
```

```
enter a snetencehi this is dee and my name is
enter a number fo rn grams3
('hi', 'this', 'is')
('this', 'is', 'dee')
('is', 'dee', 'and')
('dee', 'and', 'my')
('and', 'my', 'name')
('my', 'name', 'is')
```

```
#Word Embedding

from gensim.models import FastText
from gensim.test.utils import common_texts
```

```
corpus = common_texts
model = FastText(sentences= corpus,vector_size=100,window=5,min_count=1,workers=4,sg=1)

word_embeddings= model.wv['computer']
word_emb1= model.wv.most_similar('computer')
print(word_emb1)
```

```
[('user', 0.15659411251544952), ('response', 0.12383826076984406), ('eps', 0.030704911798238754), ('system', 0.025573883205652237), ('interfac
```

Generated code may be subject to a license | alashetty93/PlacementAssignment1 | aditya0072001/cleantxty

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize, sent_tokenize

nltk.download('punkt_tab')
nltk.download('stopwords')
nltk.download('averaged_perceptron_tagger_eng')

stop_words=stopwords.words('english')

text = input("enter a snetence")
tokenized= sent_tokenize(text)

for i in tokenized:
  wordList= nltk.word_tokenize(i)
  wordList= [w for w in wordList if not w in stop_words]
  tagged= nltk.pos_tag(wordList)
  print(tagged)
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.
enter a snetencehi this is london
[('hi', 'NN'), ('london', 'NN')]
```

Start coding or generate with AI.