Feature Extraction using TF-IDF

You are given a list of sentences here.

```
['this is the first document Document.',
'this is the second ',
'this document is the third document.',
'and this is the fourth one.',
'is this the fifth document?']
```

Consider each of the sentences as unique documents. Your task is to replicate the results of scikit-learn's `TfidfVectorizer` **(Output)** from scratch in python.

Scikit-learn's implementation :

```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(smooth_idf = False, norm = None)
X = vectorizer.fit_transform(corpus)
vectorizer.get_feature_names_out()
```

```
import pandas as pd
df = pd.DataFrame(X.todense(), columns=vectorizer.get_feature_names_out())
print(df)
```

Output (to be replicated) :

	and	document	fifth	first	fourth	is	one	second	the	third	this
0	0.000000	3.021651	0.000000	2.609438	0.000000	1.0	0.000000	0.000000	1.0	0.000000	1.0
1	0.000000	0.000000	0.000000	0.000000	0.000000	1.0	0.000000	2.609438	1.0	0.000000	1.0
2	0.000000	3.021651	0.000000	0.000000	0.000000	1.0	0.000000	0.000000	1.0	2.609438	1.0
3	2.609438	0.000000	0.000000	0.000000	2.609438	1.0	2.609438	0.000000	1.0	0.000000	1.0
4	0.000000	1.510826	2.609438	0.000000	0.000000	1.0	0.000000	0.000000	1.0	0.000000	1.0

The columns represent a set of all unique words in the corpus, and the row index value represents the ith index of the corpus/list of documents, i.e. the ith document.

Steps to replicate `TfidfVectorizer`

1. Remove punctuation from sentences, convert each word into lowercase and get all the distinct words from the corpus
2. Calculate the Term Frequency (TF) of each of the words in each document

$$TF(w,doc) = count(w,doc)$$

where, count(w,doc) : count of occurence of each unique word in the document

3. Calculate the Inverse Document Frequency (IDF) of each unique words

$$IDF(w,D) = \ln\frac{|D|}{df(w,D)} + 1$$

where ,

- Document Frequency, df(w,D) : Number of documents that contains the target word **w**
- |D| : Total number of documents in the corpus

4. Compute the TF-IDF score for each word :

$$TF\text{-}IDF(w,doc,D) = TF(w,doc) * IDF(w,D)$$

5. Print the dataframe as shown in the **Output** above

Note : Libraries like pandas, numpy, math, and re are sufficient to replicate the above functionality