11/03/24. Experiment - 10

Aim: Create representation of document by calculating Term Frequency and Inverse Document.

Source Code :-

```
from sklearn.feature_extraction.text import TfidfVectorizer

def compute_tfidf_from_file(file_path):
    #open and read the file
    with open(file_path, 'r') as file:
        documents = file.readlines()
    # create TF-IDF vectorizer and compute TF-IDF matrix
    vectorizer = TfidfVectorizer()
    tfidf_matrix = vectorizer.fit_transform(documents)
    feature_names = vectorizer.get_feature_names_out()
    #Print out the TF-IDF score for each word in each document
    for i, doc in enumerate(documents):
        print(f"Document {i+1} TF-IDF scores:")
        for word, score in zip(feature_names, tfidf_matrix[i].toarray()[0]):
            if score > 0:
```

```
        print(f"  {word}: {score:.4f}")
    print("\n")

# Example usage
file_path = "doc.txt"  # Replace this with the path to your text file
compute_tfidf_from_file(file_path)
```

open jupyter notebook

→ click on new
→ new file.
→ give names as data.txt, → enter.
↳ give data as -

"The cat in the hat.",
"The dog barked at the cat.",
"The cat and the dog are friends."
"The sun is red."

OUTPUT:-

Document 1 TF-IDF scores:
cat : 0.3413
hat : 0.5348
in : 0.5348
the : 0.5581

Document 2 TF-IDF scores:

at : 0.4928
baked : 0.4928
cat : 0.2821
dog : 0.3485
friends : 0.4420
like : 0.4613

Document 4 TF-IDF

is : 0.5528
red : 0.5528
rose : 0.5528
like : 0.2885.