EXNO:5 NLP Task: POS Tagging and Document Ranking using TF-IDF and cosine similarity.

AIM:

To perform part of speech (POS) Tagging on a given text using spacy.

PROCEDURE:

- Install and load the spacy english model
- Input cursor text and apply 'spacy's' NLP Pipeline to perform POS tagging an each word.
- Take a user query asking how AI supports Students in learning.
- combine the doc and the query into a single campus
- use TDP-IDF rectorizer to transform the text carpus into numerical vectors.
- compute cosine similarity b/w the query vector and each doc vector.
- Rank the documents based on similarity scores in & descending order.
- Display the POS tags for the input text and the ranked list of relevant documents.

O/P:

AI → propn

driven → verb

platforms → Noun

personalize → verb

learning → verb

Score : 0.16 → AI helps automatic grading and administrative tasks in schools.

Score : 0.10 → Intelligent tutoring system adapt to each student's learning style.

PROGRAM:

```
Package to install
pip install spacy
python -m spacy download en_core_web_sum
import spacy
nlp = spacy.load ("en_core_web_sum")
text = "AI_driven platforms personalize learning
 path and help students grap concepts
                         faster"

doc = nlp (text)
for token in doc:
    print (f"{token.txt:15}→{token.pos→}")
from sklearn.feature_extraction.text import
                  Tfidf vectorizer
from sklearn.metrices.pairwise.import cosine
                  & similarity.

documents = [
"AI tools analyze student performance and
                  provide
  real-time feedback".
" Intelligent turning tutoring systems adapt
to each students learning style"
query = " How does AI support students
          in learnings").
```

```
capus = documents + [query]
 rectorizer = Tfidf vectorizer()
Tfial_matrix = vectorizer.fit_transform(corpus)
Similarities = cosine_similarity[tdidf_matrix[-1],
        tfidt_matrix[:-1]. flatten()
ranked_docs = scored (zip (similarities,
        documents) , reverse = True)

Print ("\n Top relevant documents: \n")

for score, doc in ranked docs
  Print (f "score: {score: 2f} → {doc}")
```

RESULT:
The system accurately tags each word in the input text with the grammatical role, echancing, understanding of sentence structure successfully.