NLP Concepts

Code 1

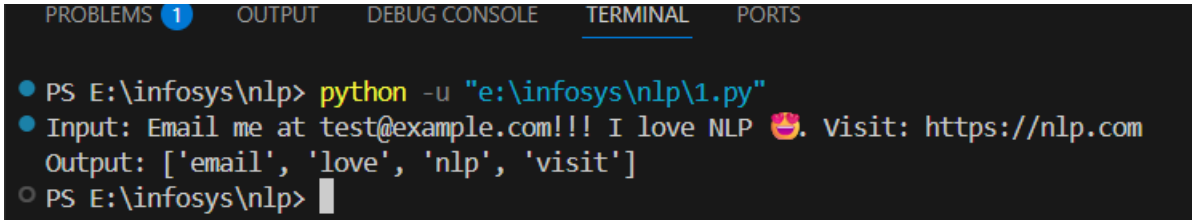
- **Concept:**

Text preprocessing – cleaning, tokenization, stopword removal, lemmatization (spaCy).

- **Output Example:**

Input Text: "Email me at test@example.com!!! I love NLP 😍. Visit: https://nlp.com"

Clean Tokens: ['email', 'filter', 'love', 'nlp', 'visit']



- **Observations:**

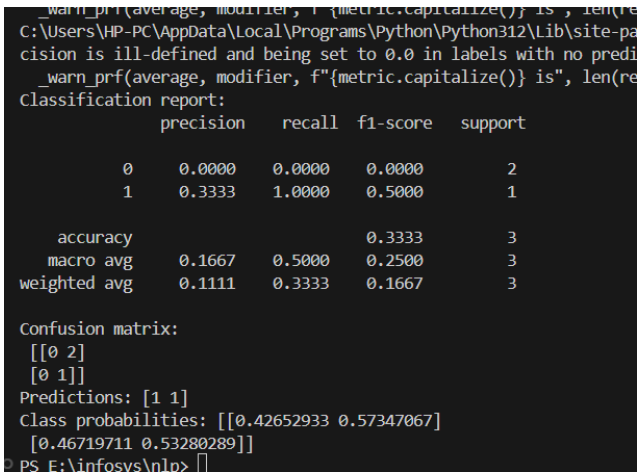
1. Converts messy raw sentences into neat, structured tokens.
2. Removes noise like emails, URLs, mentions, and stopwords.
3. Lemmatization ensures different forms of a word are treated as the same.

Code 2

- **Concept:**

Sentiment classification with TF-IDF + Logistic Regression.

- **Output Example:**



- **Observations:**

1. TF-IDF highlights important words for classification.
2. Logistic Regression learns patterns for positive vs negative text.
3. The model produces class probabilities for more interpretability.

Code 3

- **Concept:**

Hyperparameter tuning using GridSearchCV.

- **Output Example:**

```
                python -u "e:\infosys\nlp\1.py"
Best params: {'clf__C': 4.0, 'tfidf__analyzer': 'char_wb', 'tfidf__min_df': 1, 'tfidf__ngram_range': (1, 2)}
Best CV score (f1): 0.7222222222222222
Sample prediction: [1]
PS E:\infosys\nlp>
```

- **Observations:**

1. Grid search tests many parameter combinations to find the best.
2. Hyperparameter tuning boosts model accuracy.
3. The pipeline automates feature extraction + model training.

Code 4

- **Concept:**

Topic modeling with Latent Dirichlet Allocation (LDA).

- **Output Example:**

```
PS E:\infosys\nlp> python -u "e:\infosys\nlp\1.py"
Topic 0: love dogs play fetch bark rose
Topic 1: investors inflation ease expect sleep purr
Topic distribution: [[0.5 0.5]]
PS E:\infosys\nlp>
```

- **Observations:**

1. Groups words into coherent topics automatically.
2. Can classify new documents into discovered topics.
3. Helps uncover hidden themes in large text datasets.

Code 5

- **Concept:**

NER, POS tagging & noun chunks with spaCy.

- **Output Example:**

- **Observations:**

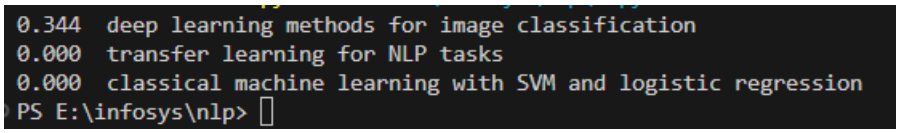
1. Identifies names, locations, organizations, and dates.
2. POS tagging reveals grammatical structure.
3. Noun chunking extracts useful phrases from text.

Code 6

- **Concept:**

Semantic search using TF-IDF + cosine similarity.

- **Output Example:**



- **Observations:**

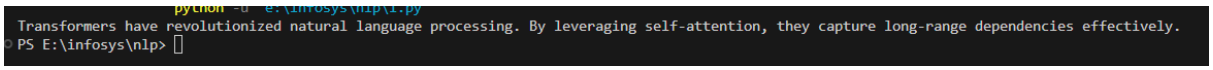
1. Finds the most relevant documents for a query.
2. Cosine similarity measures closeness between documents.
3. Basis of modern search engines & recommendation systems.

Code 7

- **Concept:**

Extractive summarization using word frequency.

- **Output Example:**



- **Observations:**

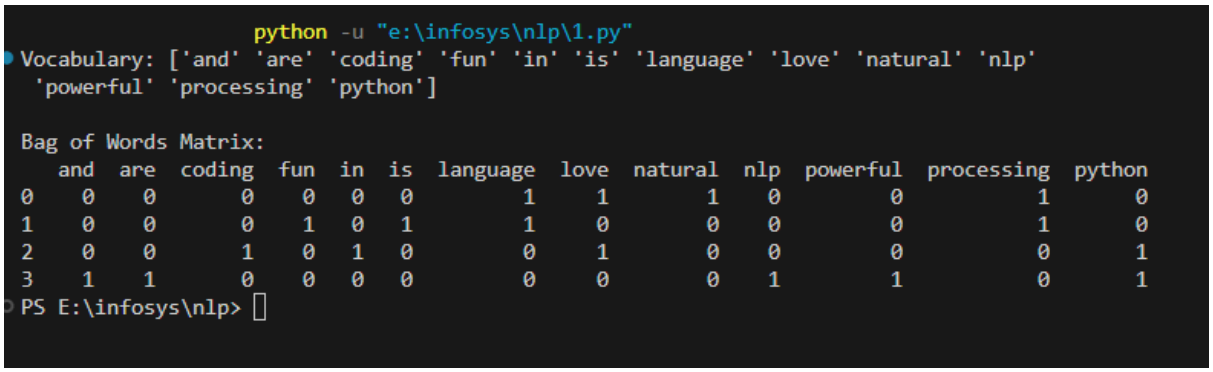
1. Selects key sentences instead of rewriting text.
2. Provides quick summaries for long articles.
3. Easy to implement but may miss context.

Code 8

- **Concept:**

Bag of Words (BoW).

- **Output Example:**



- **Observations:**

1. Represents text by counting word occurrences.
2. Simple, fast, and easy to use.

3. Ignores word order and meaning → can lose context.

Code 9

- **Concept:**

TF-IDF representation.

- **Output Example:**

```
                python -u "e:\infosys\nlp\1.py"
Vocabulary: ['advances' 'and' 'artificial' 'deep' 'fun' 'intelligence' 'is' 'learning'
 'machine']

TF-IDF Matrix:
   advances    and  artificial   deep    fun  intelligence     is  learning  machine
0     0.000  0.000       0.000  0.000  0.609          0.00  0.609     0.360    0.360
1     0.552  0.000       0.000  0.552  0.000          0.42  0.000     0.326    0.326
2     0.000  0.552       0.552  0.000  0.000          0.42  0.000     0.326    0.326
PS E:\infosys\nlp>
```

Observations:

1. Assigns higher weight to rare but important words.
2. Better than BoW for text classification.
3. Still bag-of-words based, so it misses word order.

Code 10

- **Concept:**

Word embeddings with Word2Vec.

- **Output Example:**

```
Vector for 'language':
 [ 1.56351421e-02 -1.90203730e-02 -4.11062239e-04  6.93839323e-03
 -1.87794445e-03  1.67635437e-02  1.80215668e-02  1.30730132e-02
 -1.42324204e-03  1.54208085e-02 -1.70686692e-02  6.41421322e-03
 -9.27599426e-03 -1.01779103e-02  7.17923651e-03  1.07406788e-02
  1.55390287e-02 -1.15330126e-02  1.48667218e-02  1.32509926e-02
 -7.41960062e-03 -1.74912829e-02  1.08749345e-02  1.30195115e-02
 -1.57510047e-03 -1.34197120e-02 -1.41718509e-02 -4.99412045e-03
  1.02865072e-02 -7.33047491e-03 -1.87401194e-02  7.65347946e-03
  9.76895820e-03 -1.28571270e-02  2.41711619e-03 -4.14975407e-03
  4.88066689e-05 -1.97670180e-02  5.38400887e-03 -9.50021297e-03
  2.17529293e-03 -3.15244915e-03  4.39334614e-03 -1.57631524e-02
 -5.43436781e-03  5.32639725e-03  1.06933638e-02 -4.78302967e-03
 -1.90201886e-02  9.01175756e-03]

Most similar to 'learning': [('love', 0.21057100594043732), ('deep', 0.16704079508781433), ('natural',
0.15019884705543518), ('python', 0.1320440024137497), ('processing', 0.1267007291316986), ('artificial'
, 0.09984553605318807), ('is', 0.042373016476631165), ('fun', 0.040677648037672044), ('for', 0.0124421762
30251789), ('advances', -0.012591077946126461)]

Similarity between 'python' and 'language': 0.044917457
```

- **Observations:**

1. Represents words as dense vectors.
2. Captures semantic similarity (king–man+woman≈queen).
3. Greatly improves model performance over BoW/TF-IDF.

Code 11

- **Concept:**

Naive Bayes for sentiment classification.

- **Output Example:**

```
Classification Report:
              precision    recall  f1-score   support

           0       0.50      1.00      0.67         1
           1       0.00      0.00      0.00         1

    accuracy                           0.50         2
   macro avg       0.25      0.50      0.33         2
weighted avg       0.25      0.50      0.33         2
```

- **Observations:**

1. Works very well for text with independent features.
2. Widely used in spam detection & sentiment analysis.
3. Fast to train even on large datasets.

Code 12

- **Concept:**

Cosine similarity between documents.

- **Output Example:**

```
                python -u "e:\infosys\nlp\1.py
Cosine Similarity Matrix:
 [[1.         0.64424596 0.        ]
 [0.64424596 1.         0.12413287]
 [0.         0.12413287 1.        ]]
PS E:\infosys\nlp>
```

- **Observations:**

1. Measures similarity between document vectors.

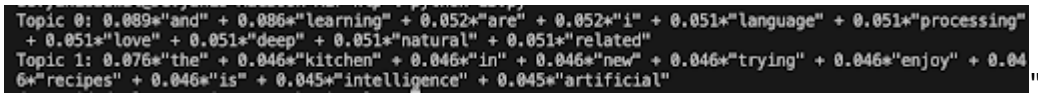
2. First two docs are close (both about NLP/ML).
3. Dissimilar docs (like cooking vs AI) get score near 0.

Code 13

- **Concept:**

Topic modeling with gensim LDA.

- **Output Example:**



"

- **Observations:**

1. Automatically groups documents into hidden themes.
2. Helps organize large corpora (e.g., news, research).
3. Topics can be labeled and used for recommendation.