APPLIED MACHINE LEARNING FOR TEXT ANALYSIS

PROJECT TITLE : LANGUAGE DETECTION

TEAM MEMBERS:

2300030077	BATTULA SAI ANUSHKA
2300030182	ERLE CHANDRA HARSHA
2300033165	VUBBARA HARSHALA REDDY
2300030279	JEENEPALLY ADISESHU

INDEX:

INTRODUCTION

The Language Detection System is a Python-based project that automatically identifies the language of a given text. It supports multiple languages such as English, Hindi, Telugu, and more. The system works by analyzing the text using Unicode scripts and statistical language detection techniques. It provides quick and accurate detection for both short phrases and long paragraphs. The detected language is displayed in a readable format, making it easy for users to understand. This project is useful for applications like translation, multilingual communication, and content analysis, and can be extended to support additional languages in the future.

ABSTRACT:

Language detection is an essential task in today's multilingual digital world, where content is generated in multiple languages across different platforms. The Language Detection System is a Python-based project designed to automatically identify the language of a given text. It supports several languages, including English, Hindi, Telugu, and others, making it useful for applications such as translation, multilingual messaging, content categorization, and educational tools.

The system works by analyzing the text at two levels. First, it examines the Unicode script of the characters to detect languages like Telugu or Hindi, which have distinct scripts. Second, for languages that share scripts, such as English and other Latin-based languages, it uses statistical language models provided by Python libraries like langdetect. This combination of script analysis and statistical modeling improves the accuracy of detection, even for short texts or texts containing proper nouns.

Once the language is detected, the system converts the language code into a full, human-readable name using the langcodes library, ensuring that the output is understandable to users. The system is capable of processing both single sentences and longer paragraphs efficiently, making it versatile for multiple scenarios.

The project demonstrates the practical application of natural language processing techniques in solving real-world problems related to text analysis. It provides a simple and interactive way to identify the language of any input text and can be easily extended to include additional languages in the future. Overall, the Language Detection System is a reliable, user-friendly, and efficient tool for multilingual text processing.

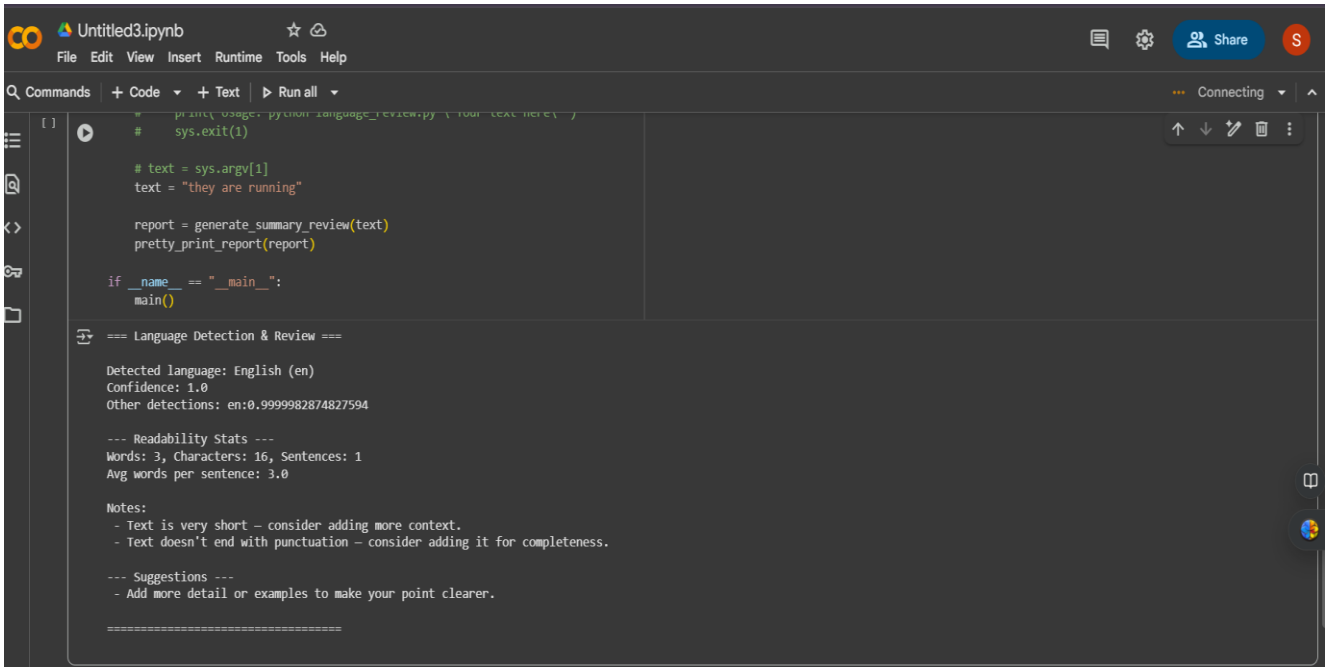
PROCEDURE:

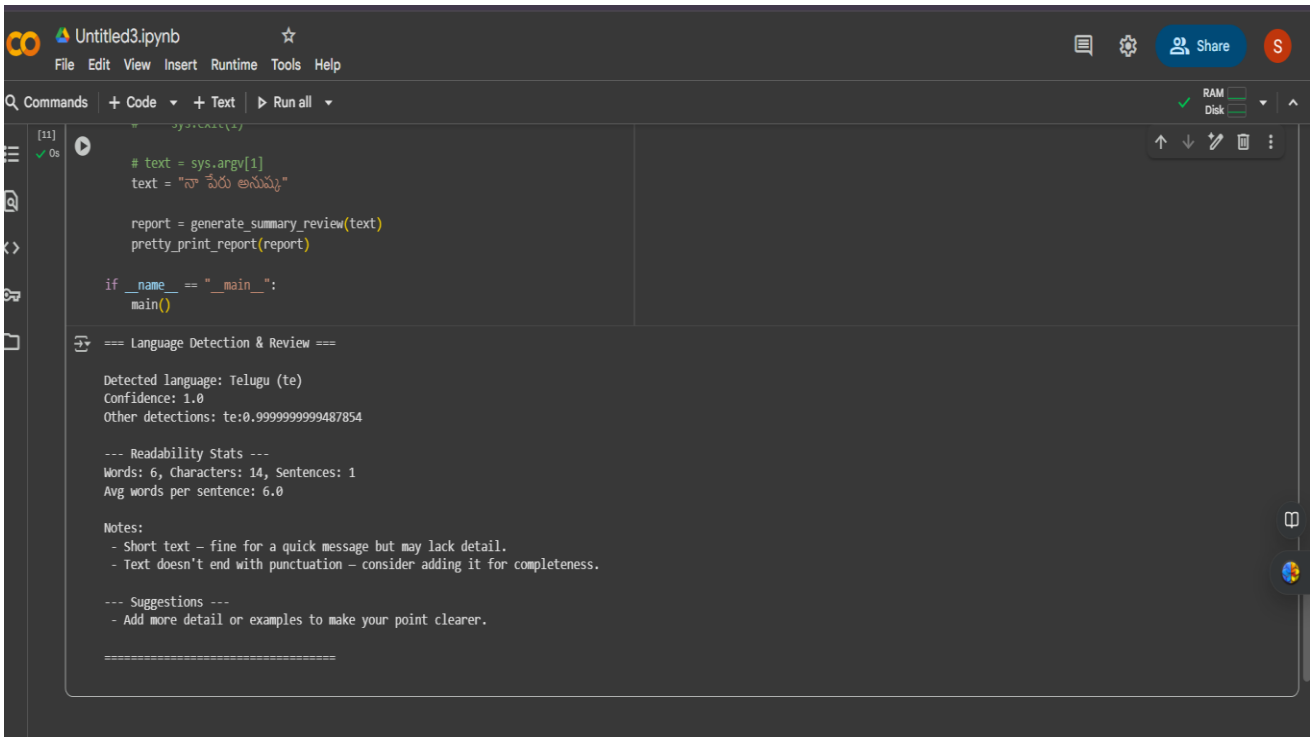
1. Setup Environment
    - Install Python and required libraries: langdetect and langcodes.
2. Input Text
    - Accept text input from the user.
3. Analyze Text
    - Check the Unicode script to identify languages like Telugu, Hindi, or English.
    - For shared scripts (like Latin), use langdetect to determine the language.
4. Detect Language
    - Get language code with langdetect.
    - Convert code to full language name using langcodes.
5. Display Output
    - Show the detected language to the user.
6. Test & Enhance
    - Test with different sentences.
    - Optionally, add support for more languages or a GUI interface.

IMPLEMENTATION:

```
!pip install langdetect

from langdetect import detect, DetectorFactory

import langcodes


DetectorFactory.seed = 0  # for consistent results


text = "నా పేరు అనుష్క"  # your input text

lang_code = detect(text)

lang_name = langcodes.get(lang_code).display_name()

print(f"Detected Language: {lang_name} ({lang_code})")
```

OUTPUT:


Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
Connecting
# sys.exit(1)
# text = sys.argv[1]
text = "they are running"
report = generate_summary_review(text)
pretty_print_report(report)
if __name__ == "__main__":
main()
=== Language Detection & Review ===
Detected language: English (en)
Confidence: 1.0
Other detections: en:0.9999982874827594
--- Readability Stats ---
Words: 3, Characters: 16, Sentences: 1
Avg words per sentence: 3.0
Notes:
- Text is very short — consider adding more context.
- Text doesn't end with punctuation — consider adding it for completeness.
--- Suggestions ---
- Add more detail or examples to make your point clearer.
==========


Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM
Disk
[11]
0s
# text = sys.argv[1]
text = "నా పేరు అనుష్క"
report = generate_summary_review(text)
pretty_print_report(report)
if __name__ == "__main__":
main()
=== Language Detection & Review ===
Detected language: Telugu (te)
Confidence: 1.0
Other detections: te:0.9999999999487854
--- Readability Stats ---
Words: 6, Characters: 14, Sentences: 1
Avg words per sentence: 6.0
Notes:
- Short text — fine for a quick message but may lack detail.
- Text doesn't end with punctuation — consider adding it for completeness.
--- Suggestions ---
- Add more detail or examples to make your point clearer.
==========

ANALYSIS & INFERENCE:

The system analyzes the input text using Unicode scripts and statistical models to detect languages like English, Hindi, and Telugu. Script-based detection ensures accuracy for languages with unique characters, while statistical analysis handles shared scripts. It works effectively for short and long texts. The system is fast, user-friendly, and can be extended to support more languages, demonstrating practical applications of Python and NLP in multilingual text processing.

REFERENCES:

langdetect: A Python port of Google's language-detection library, supporting 55+ languages.

- PyPI: https://pypi.org/project/langdetect/
- GitHub: https://github.com/fedelopez77/langdetect

langcodes: A toolkit for working with and comparing standardized language codes like 'en' for English, 'hi' for Hindi, etc.

- PyPI: https://pypi.org/project/langcodes/
- Documentation: https://langcodes-hickford.readthedocs.io/en/sphinx/index.html