Introduction :

Language detection is an essential component in various applications, such as translation tools, content moderation systems, and multilingual user interfaces. This project demonstrates the implementation of a text-based language detection system using Python. The system leverages machine learning and natural language processing (NLP) techniques to identify the language of a given text input. By creating a robust and efficient language detection model, we explore fundamental concepts such as feature extraction, dataset preprocessing, model training, and text normalization.

Aim :

To develop a command-line language detection model capable of accurately predicting the language of a given text input using supervised machine learning techniques.

Theory :

Language detection models rely on analyzing textual patterns and mapping them to predefined linguistic features. In this project, we use Python to develop a model that identifies the language of an input sentence based on a predefined dataset containing language-word mappings. The project involves:

- **Data Preprocessing:** Organizing and cleaning language datasets to create a foundation for model training.
- **Feature Extraction:** Transforming textual data into a numerical format that machine learning algorithms can process.
- **Model Training:** Using supervised learning to train a machine learning model on labeled data.
- **Text Normalization:** Standardizing text input to ensure compatibility with the model.
- **Evaluation:** Measuring the model's performance through rigorous testing.

The project demonstrates the interplay of theoretical concepts such as supervised learning, NLP techniques, and file handling in building functional AI solutions.

Procedure :

1. **Environment Setup:**
    - Install Python and required libraries (pandas, scikit-learn, string).
    - Configure the development environment for seamless execution.
2. **Dataset Preparation:**
    - Format the dataset in a .csv file, organized into language-specific word lists.
    - Load the dataset into Python using the pandas library.

- Parse and clean the dataset to remove duplicates or inconsistencies.

3. **Data Preprocessing:**
   - Normalize text by converting it to lowercase and removing unnecessary punctuation.
   - Tokenize input sentences into individual words.
4. **Feature Extraction and Model Training:**
   - Represent language-word mappings using dictionaries.
   - Implement word-matching techniques to identify the presence of language-specific features.
   - Build a rule-based model to calculate language matches for input sentences.
5. **Interactive Command Interface:**
   - Develop a user-friendly command-line interface for language detection.
   - Allow users to input sentences for analysis.
   - Display detected languages along with the number of matches.
6. **Error Handling:**
   - Validate user inputs to avoid empty or invalid sentences.
   - Handle unmatched words gracefully by offering the option to add them to the dataset.
   - Provide meaningful error messages in case of system issues.
7. **Testing:**
   - Test the system with multilingual sentences to evaluate accuracy.
   - Handle edge cases, such as mixed-language inputs or unseen words.
   - Ensure the system correctly identifies all supported languages.

Code :

1036.py

```python
import pandas as pd
import string
from collections import Counter
import csv

class LanguageDetectionModel:
    def __init__(self, dataset_path='Dataset.csv'):
        self.dataset_path = dataset_path
        self.languages = {}
        self.load_dataset()

    def load_dataset(self):
        """Load and organize the dataset into a dictionary of language-specific word sets."""
        try:
            # Read the raw content of the CSV file
            with open(self.dataset_path, 'r', encoding='utf-8') as file:
                content = file.read()

            # Split content into lines and process each line
            lines = content.strip().split('\n')
            current_language = None

            for line in lines:
                if line.startswith('Language,'):
                    current_language = line.split(',')[1].strip()
                    self.languages[current_language] = set()
                elif current_language and line.strip():
                    # Split the line by comma and add words to the language set
                    words = line.split(',')
                    self.languages[current_language].update(
                        word.strip().lower() for word in words if word.strip()
                    )
```

```
        if not self.languages:
            raise ValueError("No language data loaded from the dataset")

        # Debugging: Print loaded languages for verification
        print("Languages loaded:", list(self.languages.keys()))

    except Exception as e:
        print(f"Error loading dataset: {e}")
        exit(1)


def normalize_text(self, text):
    """Remove punctuation and convert to lowercase."""
    text = text.lower()
    # Remove punctuation except for apostrophes
    text = ''.join(char for char in text if char not in string.punctuation or char == "'")
    return text

def detect_language(self, sentence):
    """Detect the language of the input sentence."""
    words = self.normalize_text(sentence).split()
    language_matches = {lang: 0 for lang in self.languages}
    unmatched_words = []

    for word in words:
        word_matched = False
        for language, word_set in self.languages.items():
            if word in word_set:
                language_matches[language] += 1
                word_matched = True
        if not word_matched:
            unmatched_words.append(word)

    # Debugging: Check if "English" is processed correctly
    if "English" not in language_matches:
        print("Debug: English language not loaded properly")

    return language_matches, unmatched_words


def handle_unmatched_words(self, unmatched_words):
    """Process unmatched words and add them to the dataset."""
    if not unmatched_words:
        return

    print("\nUnmatched words found:")
    for word in unmatched_words:
        print(f"Word: {word}")
        print("Select language:")
        for idx, language in enumerate(self.languages.keys(), 1):
            print(f"{idx}. {language}")

        while True:
            try:
                choice = int(input("Enter language number: "))
                if 1 <= choice <= len(self.languages):
                    selected_language = list(self.languages.keys())[choice - 1]
                    # Add to memory
                    self.languages[selected_language].add(word)
                    # Add to file
                    self.append_word_to_file(word, selected_language)
                    break
                else:
                    print("Invalid choice. Please try again.")
            except ValueError:
                print("Please enter a valid number.")

def append_word_to_file(self, word, language):
    """Append a new word to the dataset file."""
    try:
        with open(self.dataset_path, 'r', encoding='utf-8') as file:
            lines = file.readlines()

        # Find the line containing the language
        for i, line in enumerate(lines):
            if line.startswith(f'Language,{language}'):
                # Append the word to the next line
                if i + 1 < len(lines):
                    lines[i + 1] = lines[i + 1].strip() + ',' + word + '\n'
                break
```

```
            with open(self.dataset_path, 'w', encoding='utf-8') as file:
                file.writelines(lines)
        except Exception as e:
            print(f"Error adding word to dataset: {e}")

    def predict_language(self, language_matches):
        """Determine the most likely language(s) based on word matches."""
        if not any(language_matches.values()):
            return "Language detection unavailable - no matches found"

        max_matches = max(language_matches.values())
        if max_matches == 0:
            return "No language matches found"

        predicted_languages = [
            lang for lang, count in language_matches.items()
            if count == max_matches
        ]

        if len(predicted_languages) == 1:
            return f"Detected language: {predicted_languages[0]} ({max_matches} words matched)"
        else:
            return f"Multiple possible languages detected: {', '.join(predicted_languages)} ({max_matches} words matched in
each)"

def main():
    model = LanguageDetectionModel()

    while True:
        print("\nLanguage Detection System")
        print("————————————————————")
        sentence = input("Enter a sentence: ")

        if not sentence.strip():
            print("Please enter a valid sentence.")
            continue

        language_matches, unmatched_words = model.detect_language(sentence)

        # Display prediction
        prediction = model.predict_language(language_matches)
        print(f"\n{prediction}")

        # Display match statistics
        print("\nMatch statistics:")
        for language, count in language_matches.items():
            print(f"{language}: {count} matches")

        # Handle unmatched words
        if unmatched_words:
            print(f"\nUnmatched words: {', '.join(unmatched_words)}")
            add_words = input("Would you like to add these words to the dataset? (y/n): ")
            if add_words.lower() == 'y':
                model.handle_unmatched_words(unmatched_words)

        # Continue or exit
        choice = input("\nWould you like to enter another sentence? (Y/N): ").strip().lower()
        if choice == 'n':
         break

if __name__ == "__main__":
    main()
```

The implementation includes Python classes and functions to handle dataset loading, text preprocessing, language matching, and user interaction. Detailed code will be provided separately.

Dataset.csv :

```
Dataset.csv > data
Language,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
Language,English  ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
hello,world,good,morning,night,afternoon,evening,yes,no,please,thank,you,welcome,sorry,excuse,help,friend,family,love,happy,sad,angry,tir
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
Language,Spanish  ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
hola,mundo,buenos,días,tardes,noches,sí,no,por,favor,gracias,de,nada,perdón,disculpa,ayuda,amigo,familia,amor,feliz,triste,enojado,cansad
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
Language,French  ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
bonjour,monde,merci,salut,oui,non,s'il,te,plaît,pardon,désolé,excusez-moi,aide,ami,famille,amour,heureux,triste,énervé,fatigué,affamé,ass
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
Language,Hindi  ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
नमस्ते,धन्यवाद,सुप्रभात,शुभरात्रि,कृपया,माफ़,क्षमा,शुभकामनाएँ,अलविदा,कैसे,आप,क्या,कहाँ,क्यों,कौन,कब,यह,वह,यहाँ,वहाँ,मैं,तुम,हम,आपका,उसका,उनका,इसका,खुश,दुखी,
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
Language,Gujarati  ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
નમસ્તે,આભારી,શુભપ્રભાત,શુભરાત્રિ,કૃપા,માફ,માફ કરજો,શુભેચ્છાઓ,આવજો,કેવી રીતે,તમે,શું,ક્યાં,શા માટે,કોણ,ક્યારે,આ,તે,અહીં,ત્યાં,હું,તમે,અમે,તમારું,તેનું,તેમનું,આનું,ખુ
```

Output :

```
C:\WINDOWS\py.exe
Languages loaded: ['English', 'Spanish', 'French', 'Hindi', 'Gujarati']

Language Detection System
------------------------
Enter a sentence: Hola, ¿cómo estás? ¿Qué estás haciendo estos días? Está bien.

Detected language: Spanish (8 words matched)

Match statistics:
English: 0 matches
Spanish: 8 matches
French: 0 matches
Hindi: 0 matches
Gujarati: 0 matches

Unmatched words: está, bien
Would you like to add these words to the dataset? (y/n): y

Unmatched words found:
Word: está
Select language:
1. English
2. Spanish
3. French
4. Hindi
5. Gujarati
Enter language number: 2
Word: bien
Select language:
1. English
2. Spanish
3. French
4. Hindi
5. Gujarati
Enter language number: 2

Would you like to enter another sentence? (Y/N):
```

The program successfully detects the language of input sentences, displaying match statistics and unmatched words. Outputs will be provided with sample inputs.

Features :

- **Command-Line Interface:** An interactive interface for seamless user interaction.
- **Multilingual Support:** Detects multiple languages, including English, Spanish, French, Hindi, and Gujarati.
- **Dynamic Dataset Updates:** Adds new words to the dataset dynamically upon user confirmation.
- **Comprehensive Error Handling:** Ensures smooth execution with user-friendly error messages.
- **Real-Time Language Detection:** Predicts the language with high accuracy and provides detailed statistics.

Result :

The language detection model successfully implemented core functionalities, including sentence normalization, multilingual support, and unmatched word handling. The program delivers accurate predictions for input sentences and supports dynamic dataset updates.

Conclusion :

This project illustrates the practical application of natural language processing and machine learning concepts in building a language detection model. By implementing the system in Python, I gained hands-on experience with text preprocessing, dataset management, and interactive user interfaces. The project serves as a foundation for further exploration into NLP and machine learning, demonstrating the potential of AI in enhancing linguistic applications.