3. Student Activity

Student Activity: Exploring AI-Powered Coding Assistants

Objective:
To help students understand and practice using AI-powered coding assistants through hands-on examples. This activity will guide students through the basic concepts of AI coding assistants, focusing on Natural Language Processing (NLP), Machine Learning (ML), and code suggestions.

Time Required:
Approximately 30-45 minutes

Activity Steps

1. Understanding AI-Powered Coding Assistants

Example 1: Basic Code Suggestion

- **Task:** Open a code editor that supports AI coding assistants, such as Visual Studio Code with GitHub Copilot.
- **Action:** Start typing a simple function in Python, such as `def add_numbers(a, b):`.
- **Observation:** Notice how the AI suggests completing the function with a return statement like `return a + b`.

Example 2: Error Correction

- **Task:** Intentionally write a piece of code with a syntax error, such as `print("Hello World"`.
- **Action:** Observe how the AI suggests the correct syntax by adding the missing parenthesis.
- **Observation:** Accept the suggestion and see the corrected code.

Example 3: Code Block Generation

- **Task:** Write a comment in the code editor, such as `# Create a function to multiply two numbers`.
- **Action:** Observe how the AI generates a complete function based on the comment.
- **Observation:** Review the generated code and understand how the AI interpreted the comment.

---

2. Exploring Natural Language Processing (NLP) in Coding Assistants

Example 1: Comment-Based Code Generation

- **Task:** Write a comment like `# Function to check if a number is even`.
- **Action:** Observe how the AI suggests a function that checks if a number is even.
- **Observation:** Analyze the suggested code and see how it matches the comment.

Example 2: Instruction Interpretation

- **Task:** Write a comment such as `# Sort a list of numbers in ascending order`.
- **Action:** Observe the AI's suggestion for a sorting function.
- **Observation:** Compare the AI's suggestion with a standard sorting algorithm.

Example 3: Multi-Language Support

- **Task:** Write a comment in a different language (if supported), such as Spanish: `# Crear una función para sumar dos números`.
- **Action:** Observe how the AI interprets the comment and suggests code.
- **Observation:** Evaluate the AI's ability to understand and generate code from non-English comments.

---

3. Exploring Machine Learning (ML) in Coding Assistants

Example 1: Learning from Patterns

- **Task:** Write a loop structure, such as `for i in range(5):`.
- **Action:** Observe how the AI suggests common loop operations like printing the index.
- **Observation:** Understand how the AI uses learned patterns to make suggestions.

Example 2: Improving Suggestions Over Time

- **Task:** Write a series of similar functions, such as different mathematical operations.
- **Action:** Notice how the AI's suggestions become more accurate with each function.
- **Observation:** Discuss how the AI improves its suggestions based on repeated patterns.

Example 3: Contextual Code Suggestions

- **Task:** Start a complex function, such as a recursive Fibonacci sequence.
- **Action:** Observe how the AI suggests completing the function based on the context.
- **Observation:** Analyze the AI's understanding of the recursive pattern.

Conclusion

Through this activity, students will gain hands-on experience with AI-powered coding assistants, understanding how they leverage NLP and ML to assist in coding tasks. By practicing with real examples, students will see the practical benefits of these tools in enhancing coding efficiency and accuracy.

Discussion Points:

- How did the AI's suggestions help in writing code faster?
- What were some limitations or challenges faced while using the AI assistant?
- How can these tools be integrated into daily coding practices?

Note: Encourage students to explore different AI coding assistants and experiment with various programming languages to broaden their understanding.