# The Effect of AI Tools on Static Analysis of Source Code for Authorship Attribution

Collin Bale, Genevieve Kochel, Natalie Simkins
Mentors: Tashi Stirewalt and Assefaw Gebremedhin

## INTRODUCTION

- In **software security**, understanding code **provenance**—its origins and history—is crucial for protecting systems and data. Provenance verification helps identify risks, trace unauthorized contributions, and maintain accountability in software development.
- **Stylometry** analysis, traditionally used for **authorship attribution** in literature, is a valuable tool for attributing source code to its authors. However, it faces challenges, such as undisclosed contributions and attempts to disguise authorship.
- The rise of **AI tools in software development** has transformed programming, but it also complicates traditional authorship attribution. Developers may not disclose their use of AI, thereby avoiding scrutiny for unapproved tools, which highlights the need to assess potential risks, such as **supply chain vulnerabilities and security flaws**.
- Initiatives, such as President Biden's Executive Order 14028 mandating **Software Bills of Materials (SBOMs)**, highlight the importance of securing code provenance. An SBOM **details all components of a software application**, helping organizations identify vulnerabilities and ensure software integrity.
- This study examines **how AI tools impact static stylometry analysis for authorship attribution**, investigating how machine learning (ML) models can adapt to evolving writing styles. Training these models to recognize human authors' unique signatures alongside AI-generated patterns can enhance their ability to detect external influences on code and predict unknown contributions.

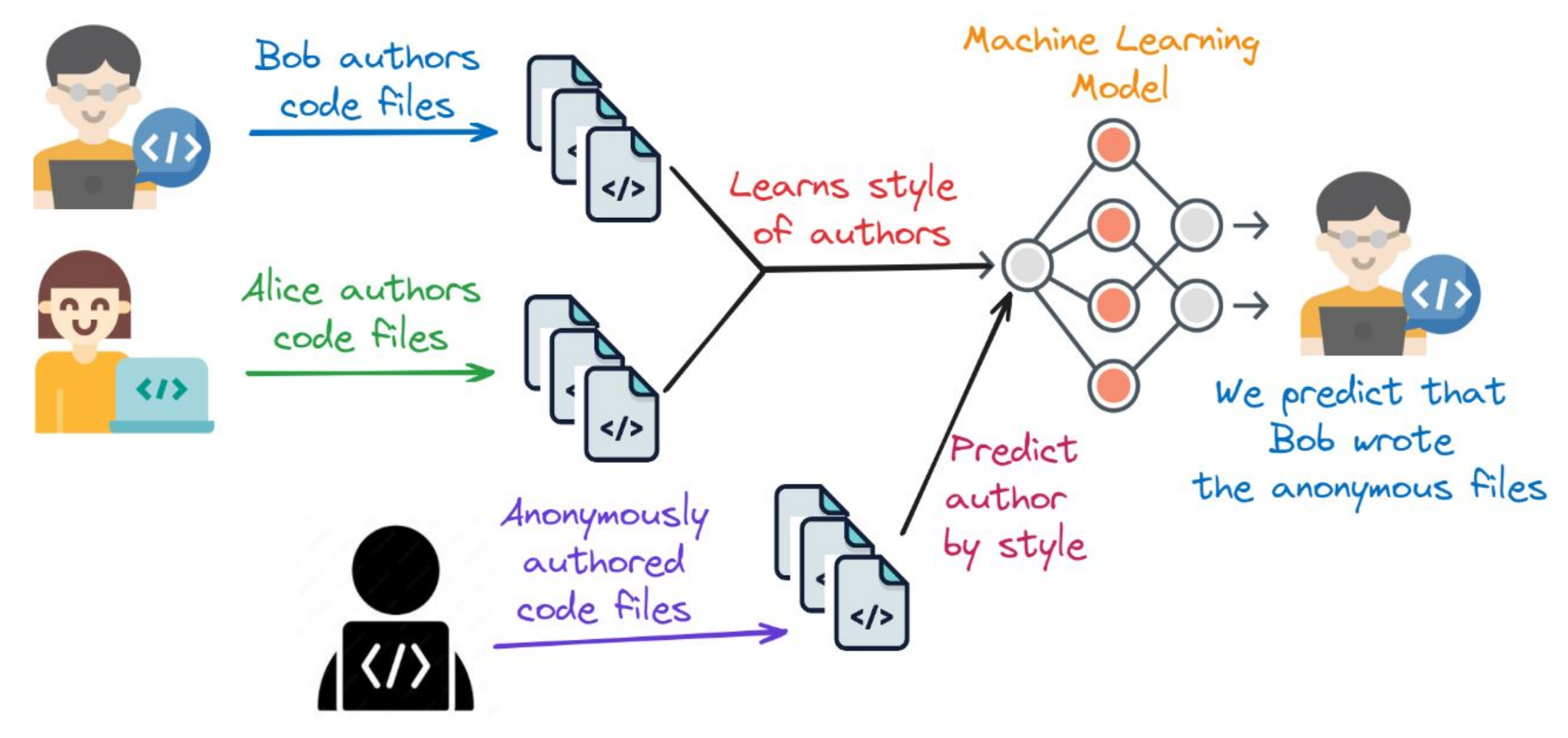


*Figure 1: Illustrative overview of ML-based stylometry for authorship attribution*

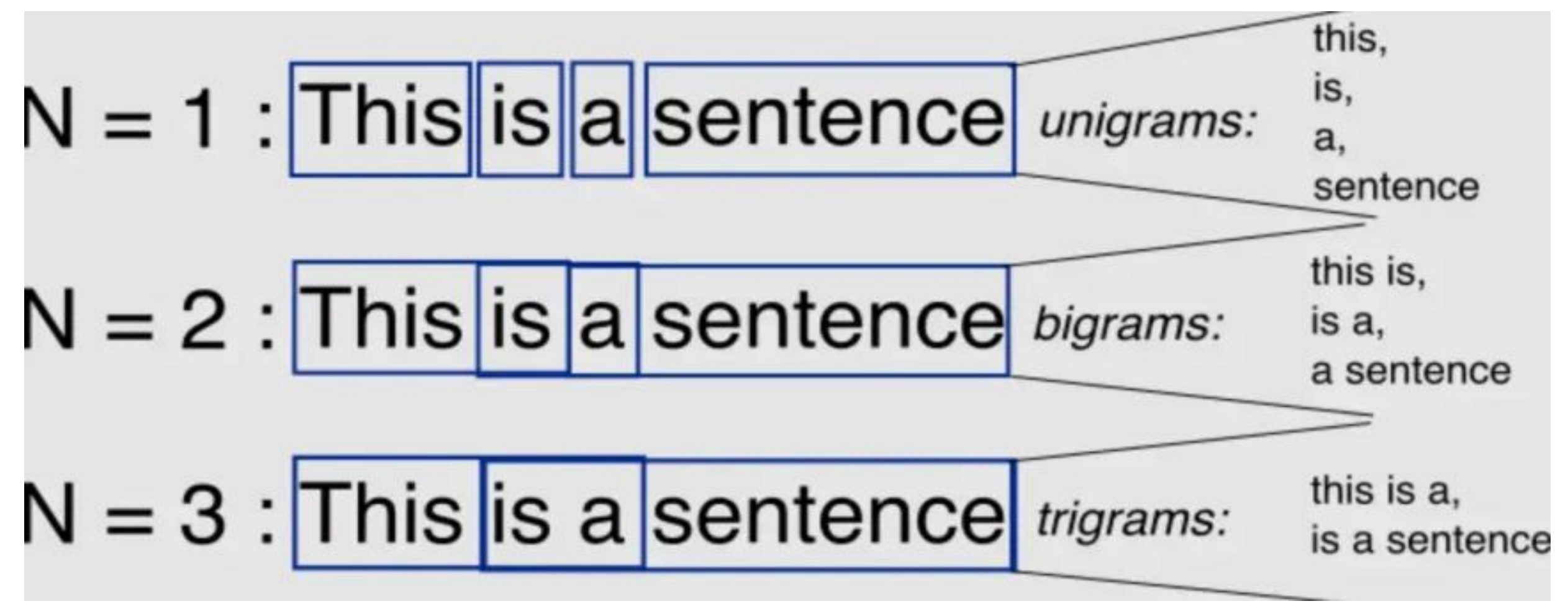
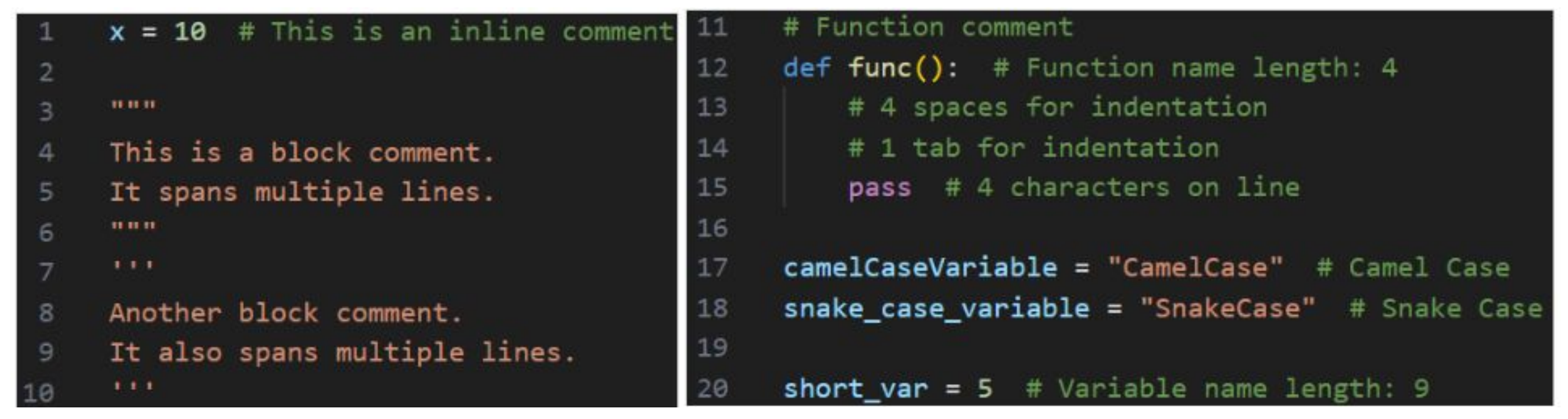

N = 1 : This is a sentence unigrams: this, is, a, sentence
N = 2 : This is a sentence bigrams: this is, is a, a sentence
N = 3 : This is a sentence trigrams: this is a, is a sentence

*Figure 2: Example of word itemized n-grams sized 1-3.*

## PREPROCESSING

- To prepare the data for ML, **vectorization** converted the textual content of each file into numerical feature representations of character **n-grams** from **five to nine characters long**.
- N-grams are contiguous sequences of "n" items, and only the **top five thousand most frequent** n-grams were retained after a frequency analysis to form a sparse matrix of token counts.
- **10 Python specific features** were also isolated for feature extraction including:

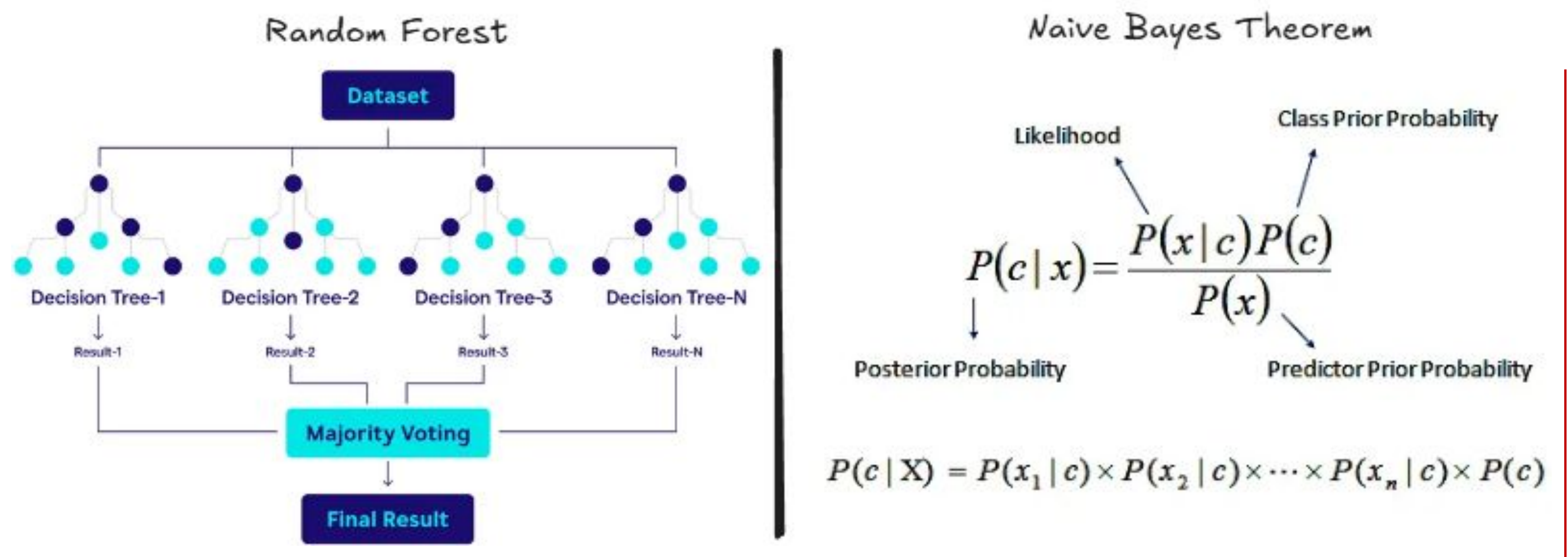
1. ***Inline comments*** *(at the end of a line).*
2. ***Block comments*** *(in triple single/double quotes).*
3. ***Function comments*** *(just above function defs).*
4. ***Camel Case*** *(e.g., "variableName").*
5. ***Snake Case*** *(e.g., "variable_name").*
6. ***Length of variable names****.*
7. ***Length of function names****.*
8. ***Length of each line****.*
9. ***Number of tabs****.*
10. ***Number of spaces****.*



```python
x = 10  # This is an inline comment

"""
This is a block comment.
It spans multiple lines.
"""
'''
Another block comment.
It also spans multiple lines.
'''
```

```python
# Function comment
def func():  # Function name length: 4
    # 4 spaces for indentation
    # 1 tab for indentation
    pass  # 4 characters on line

camelCaseVariable = "CamelCase"  # Camel Case
snake_case_variable = "SnakeCase"  # Snake Case

short_var = 5  # Variable name length: 9
```

*Figure 3: Examples of each Python-based feature.*



$$P(c|x)=\frac{P(x|c)P(c)}{P(x)}$$

$$P(c|X) = P(x_1|c)\times P(x_2|c)\times\cdots\times P(x_n|c)\times P(c)$$

*Figure 4: Overview of key concepts related to RF (Left) and NB (right).*

## MACHINE LEARNING

- **Random Forest (RF)** is a "forest" of decision trees through Bootstrap Sampling, where each tree is trained on a random subset of the data.
- **Multinomial Naive Bayes (NB)** is a probabilistic model popularly used for text classification, based on Bayes' Theorem.
- A hyperparameter search space was defined and **Bayesian optimization** was applied.
- A **five-fold cross-validation** with a stratified K-fold approach to preserve the label distribution.
- Evaluation metric was the **Receiver Operating Characteristic Area Under the Curve (ROC AUC)** score.
- **Dataset** was created featuring source code samples from **human authors and AI tools**.

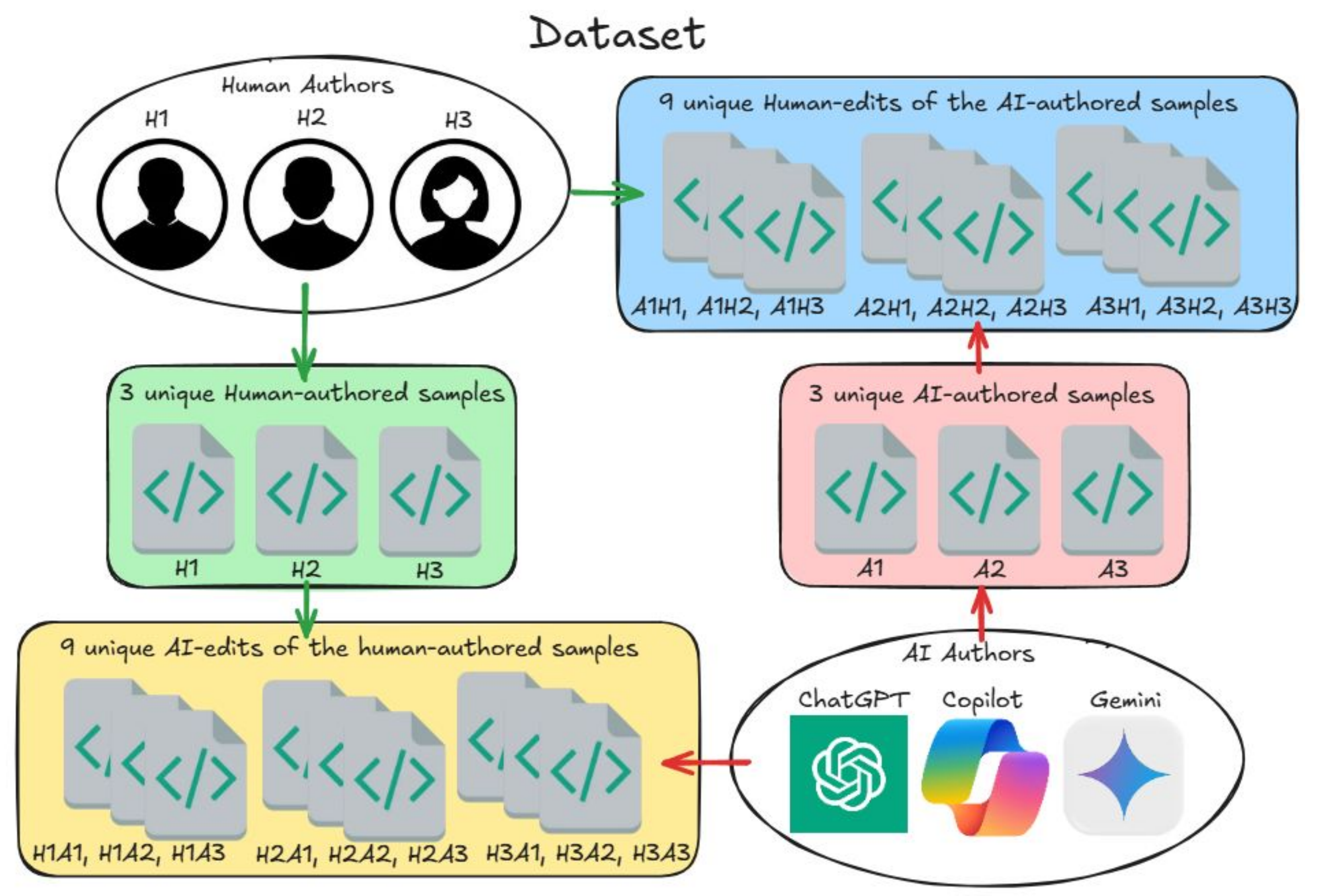


*Figure 5: Overview of the dataset composition.*

## RESULTS

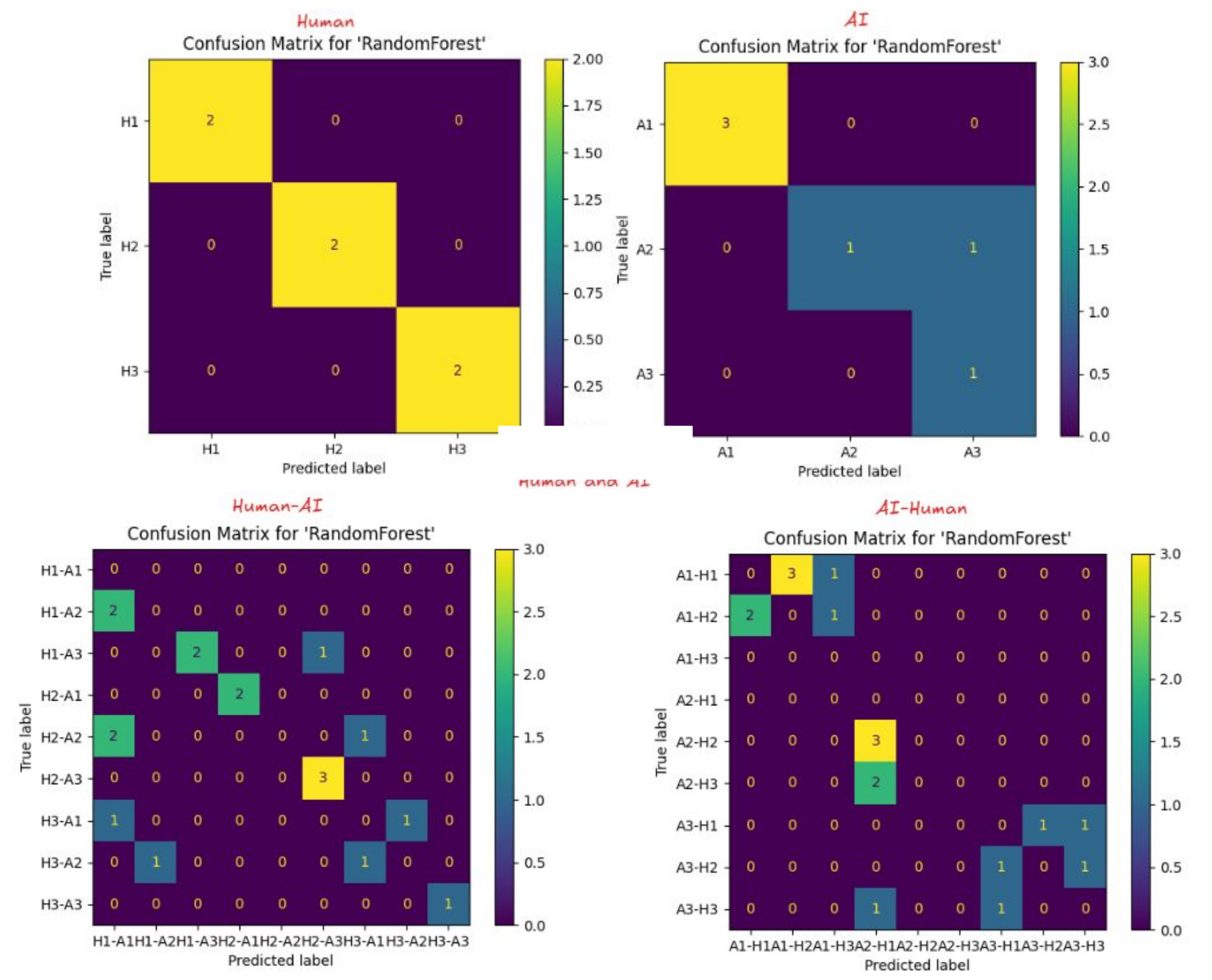


*Figure 6: Confusion matrices for each category's best model.*

## CONCLUSION

As supply chains face more targeted threats, **ensuring software provenance is essential for security**. Progress in stylometry helps detect disinformation and malware; however, its application in **evaluating undisclosed AI tool usage during development is limited**. The rising use of AI by developers to enhance productivity **introduces vulnerabilities** that can lead to **stylistic replication** and **obfuscation**. To combat this, consideration of AI tools as unique authors can improve the detection of their influence on source code files.

## ACKNOWLEDGMENTS

This work is supported by funding for the VICEROY Northwest Institute for Cybersecurity Education and Research (CySER) provided by The Office of the Undersecretary of Defense for Research and Engineering, in collaboration with the Air Force Research Laboratory and Griffiss Institute.