CS345 - Project 3

Matthew Linsky: ML934
Nisha Murali: NLM126
Shrey Patel: SAP389

Evaluation of AI Text Detector

This program includes two distinct algorithms to differentiate between human-written and AI-generated text. Both are implemented in Python using `scikit-learn` and a custom feature extraction function, . The first algorithm is a comprehensive, generalizable model (the ***non-p-hacked* model**), while the other is deliberately ***p-hacked***, overfitted to a single feature so that it performs well only on the training data.

Feature Extraction (`parse_sample` algorithm)

The `parse_sample` algorithm creates and uses various metrics, such as **complexity scores** (`flesch_kincaid, gunning_fog`, and `smog_index`) and **perplexity scores** (`lmppl`) to generate means and standard deviations. These help the logistic regression model learn patterns in sentence structure so that it can identify whether the document is written by AI or human. Lower complexity and perplexity scores indicate greater predictability, which suggests AI-generated text.

The algorithm extracts the following features from text samples:

- Words per Sentence
- Paragraph count
- Sentences per Paragraph
- Words per Paragraph
- Popular words and punctuation marks

Classification Algorithms

1. Non-Hacked P-score Algorithm

The “non-hacked P-score algorithm”, or the **regular model**, is a **logistic regression model** trained on the linguistic metrics extracted by the `parse_sample` method and normalized by `StandardScalar`. The model is trained on an 80/20 train-test split to maintain data blindness.

```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

The model generates **p-scores** for each writing sample it is provided. Higher p-scores indicate higher confidence that a sample has been produced by AI. The target outcome is correctly predicting the ‘is_ai’ column value; the metrics from `parse_sample` would be used as input to do this. The AI probability threshold was altered from the standard 0.5 value to a value of **0.75,** so as to reduce the chance of false positives. Overall performance was adequate at > 50% accuracy on training and test data.

2. Hacked P-score Algorithm

The hacked p-score algorithm, or the **p-hacked model**, is intentionally overfitted to training data and is not expected to generalize to new datasets. It takes its p-scores from a single metric derived from `parse_sample`: **`pop_ai`**, a count of the most recurring words and punctuations in AI-generated text. **`pop_ai`** was selected over perplexity because it was simpler and well suited to the model’s purposes. `pop_ai` is scaled by dividing the sum of its values by their **maximum value in the training data**, producing a `p_hacked_score`. This score predicts whether a sample is AI-generated

or human-written; if it is a greater value than the AI probability threshold, it is AI-generated, and if it is lesser, then it is human-written.

This model was also trained on an 80/20 train-test split to maintain data blindness, and the AI probability threshold is 0.5. The target output is to predict the value of ‘is_ai’. Despite its simplicity, the model performs well on training data but degrades on unseen test data—demonstrating classic p-hacking behavior.

Threshold Justification

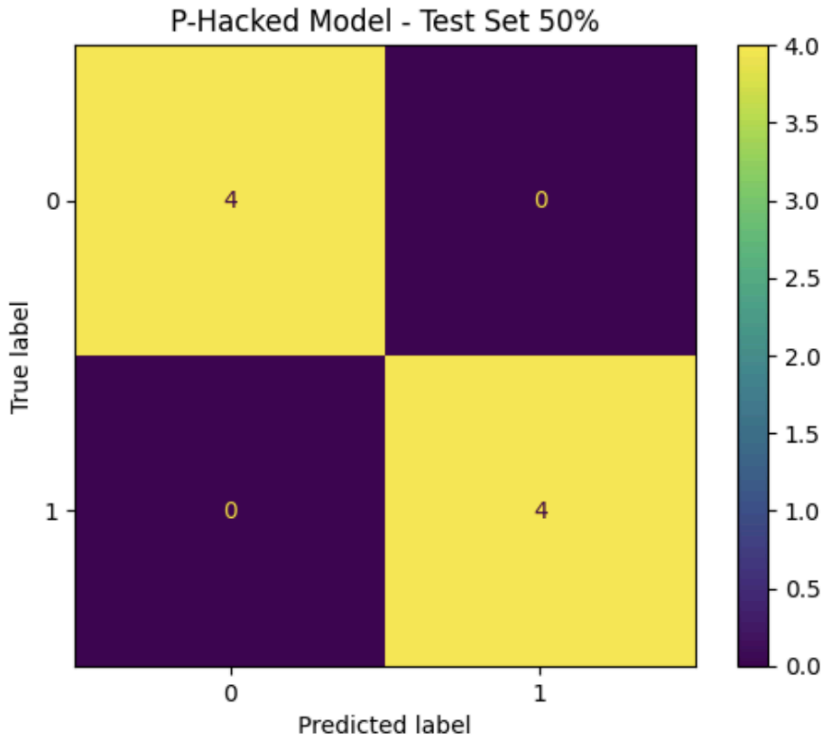
The **regular model** uses a threshold of **0.75** to optimize performance on the test set by reducing false positives while maintaining high confidence in its detection of AI samples. The **p-hacked model** uses a standard **0.5 threshold**, consistent with simple binary classification on a single scaled metric.

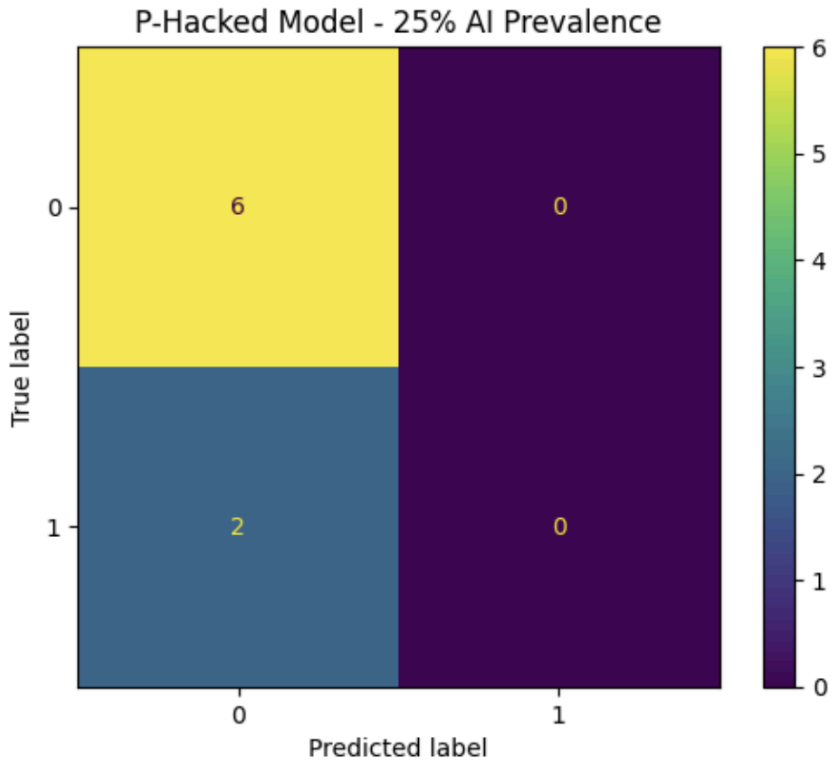
Testing on Different AI Prevalence Levels (Confusion Matrix)

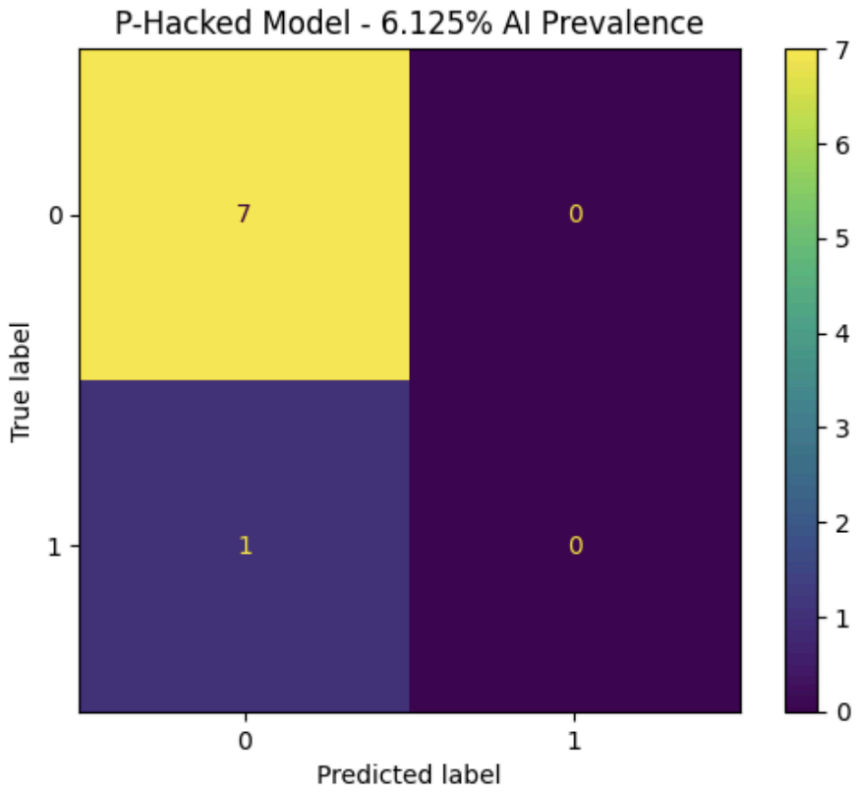
The ‘confusion matrix’ evaluates how well the AI detection models perform under different real-world AI-to-human ratios (called *prevalence levels*). It does this by generating datasets (`simulate_prevalence`) that have custom proportions of AI vs. human writing. One dataset’s samples are **25% AI** and **75% human.** Another dataset’s samples are **6.125% AI** and **93.875% human.**

The AI probability threshold is kept to 0.5. This is to understand how prediction algorithms that are ‘expecting’ a balanced ratio of AI-to-human samples would function in a less controlled, more ‘realistic’ input environment, where samples strongly skew one way or another.

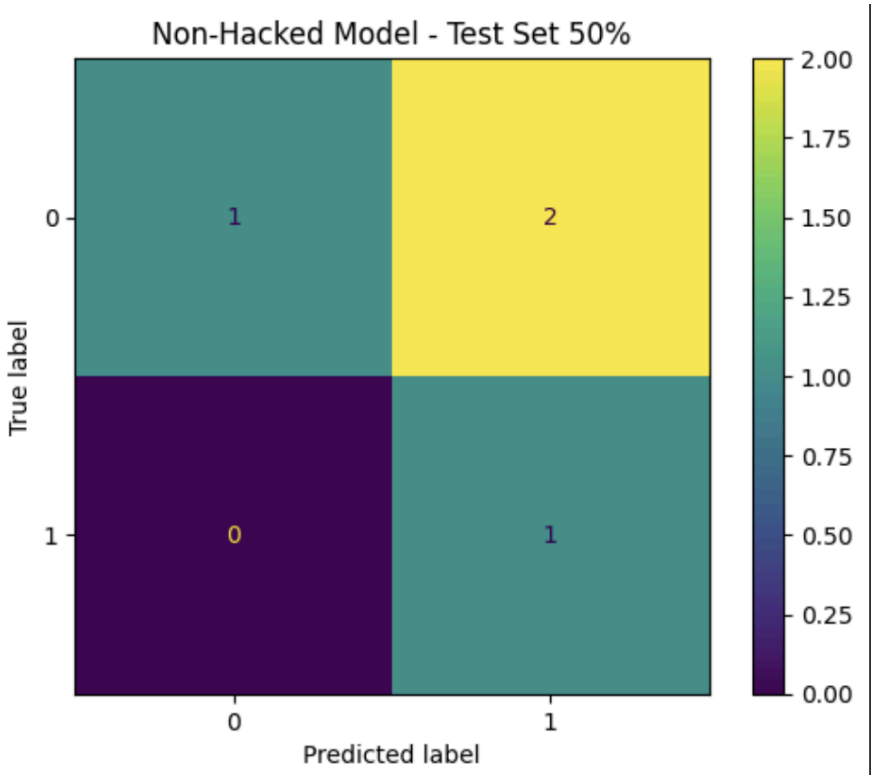
The following graphs visualize the results for the p-hacked algorithm with every prevalence level:

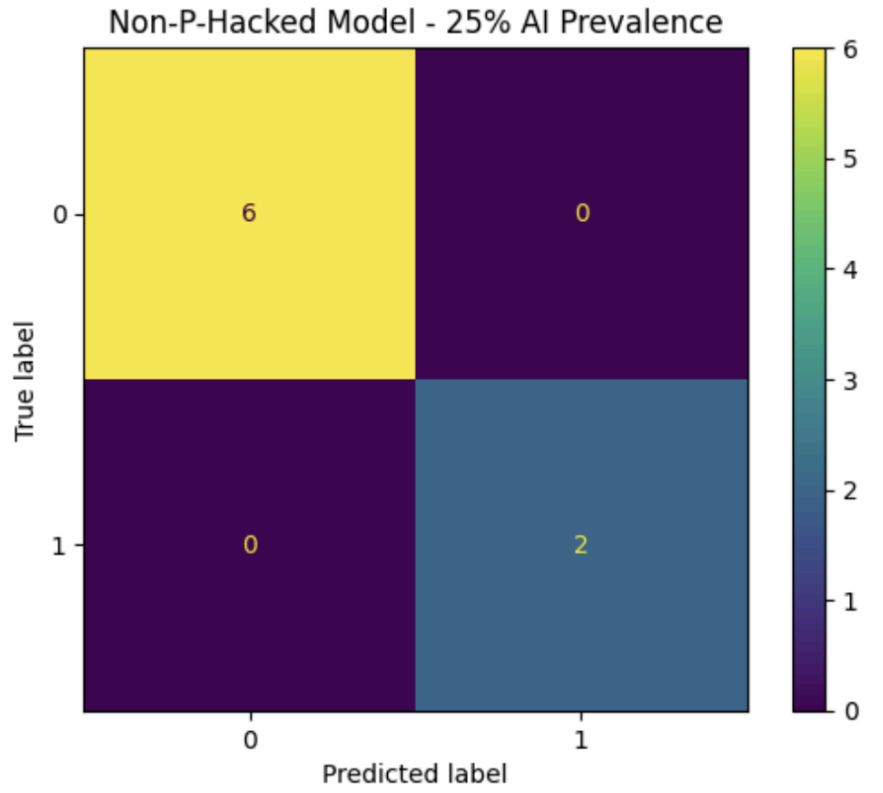


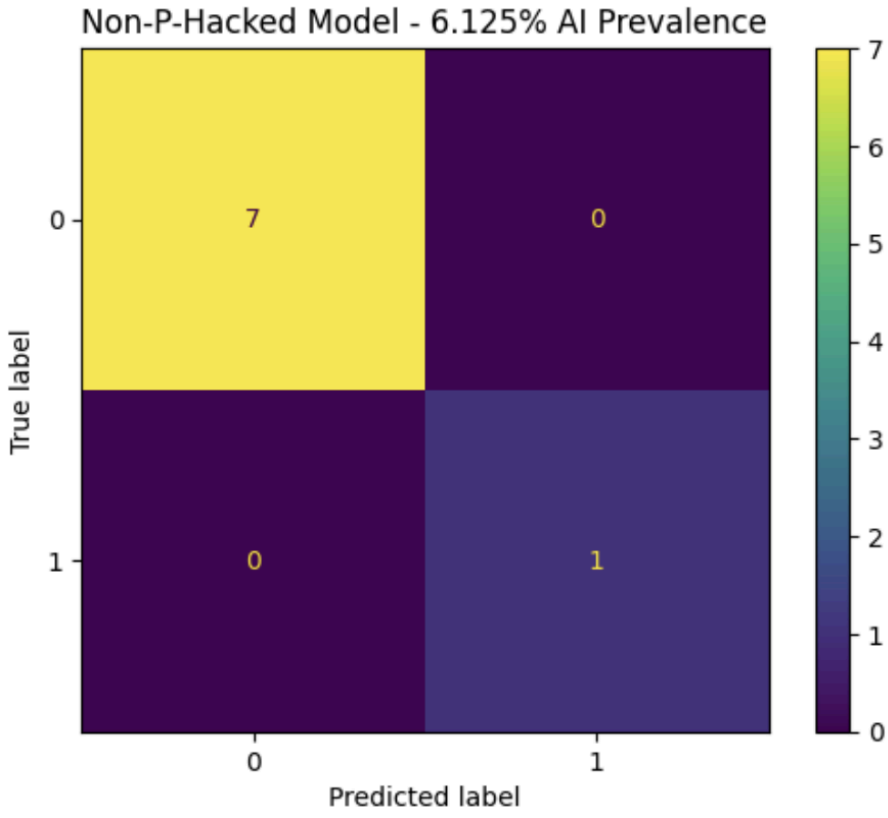




The following graphs visualize the results for the regular algorithm with every prevalence level:




Non-P-Hacked Model - 25% AI Prevalence
True label
Predicted label
0
1
6
0
0
2
6
5
4
3
2
1
0


Non-P-Hacked Model - 6.125% AI Prevalence
True label
Predicted label
0
1
7
0
0
1
7
6
5
4
3
2
1
0