Group 25 Final Project Report: AI Essay Detector

Fei Xie, Ghena Hatoum, Haniye Hamidizadeh
{xief17,hatoumg,hamidizf}@mcmaster.ca

1 Introduction

The widespread and accessible nature of Generative AI (GenAI) tools, such as ChatGPT and Gemini, has created a significant shift in the educational landscape. While these tools are valuable for quick information retrieval, their rapid adoption precedes the necessary adaptations in academic practices.
The core issue is that unchecked reliance on GenAI for academic work can have long-term developmental consequences for students. It prevents them from engaging in the process of formulating original thoughts and arguments, which is critical for developing their unique intellectual voice. A reliable AI Text Detector, integrated within the education system, would allow educators to foster responsible use of GenAI as an educational aid rather than a replacement for student effort.
Developing an effective detector is challenging for several reasons:

- Low Accuracy of Existing Models: Current models are not accurate enough for real-world academic use.
- Evolving Realism: GenAI systems are constantly improving, generating increasingly human-like text, requiring the detector to be dynamically updated.
- Critical False Positive Rate: The model must achieve a near-zero false positive rate to ensure no student is wrongfully accused of plagiarism while maintaining high accuracy in identifying genuine misuse.

Detecting AI-generated text is closely connected to research in text classification. An overview of how methods such as TF-IDF have been used in traditional machine-learning models before the development of deep-learning techniques as seen in the survey "A Survey on Text Classification: From Traditional to Deep Learning" (Li et al., 2022).

In practical applications, AI text detection has been explored through competitions, such as the kaggle AI Text Detection Competition (Al-Ahmadi). Public notebooks from the competition show examples of baseline systems that use TF-IDF combined with classifiers such as Logistic Regression or SVM. Two examples are the notebooks by Faris Al-Ahmadi (Al-Ahmadi) and Jasmine Mohamed (Mohammad), where TF-IDF is used to represent essays numerically before training classification models.
More recent work focuses on neural network approaches. LSTM models are often used for text classification tasks because they can process text as a sequence and keep track of earlier context in the input. An example of this is shown in an LSTM text-classification notebook (Khotijah). Transformer-based models such as DistilBERT have also been applied to similar classification problems. The KerasNLP starter notebook from the same Kaggle competition shows how DistilBERT can be fine-tuned for detecting AI-generated text (Audevart).
There is also research specifically focused on identifying patterns unique to AI-generated writing. The "DetectGPT" paper (Mitchell et al., 2023) shows that AI-generated text often has different patterns in how words are used compared to human writing and discusses ways to detect those differences.

2 Dataset

We will be using the DAIGT V2 Train Dataset from Kaggle (Kleczek) to train the model, as well as adding a new, manually compiled dataset with AI generated and human written essays to evaluate the trained models performance on.

2.1 DAIGT V2 Train Dataset

To start, the DAIGT V2 Train Dataset is a comprehensive dataset containing a collection of other

datasets. One of the sub datasets contain argumentative essays written by grade 6 to 12 students. The remaining sub datasets contain AI-generated essays from numerous models as seen in Table 1. The dataset is prelabelled and additional columns include prompt_name (original persuade prompt), source (original dataset), text (actual text content), and RDizzl3_seven (Classifier built for a previous Kaggle competition, indicating whether the essays were written in response to one of the seven essay prompts for the competition).
Our team has dropped all the features other than text and the label.

2.1.1 Preprocessing

Using the text feature, we have extracted additional features. These include:
Word Statistics
The features include the total number of words in the text/document, the sum of words that repeat frequently (greater or equal to 5 times), and the sum of words that appear infrequently (less than 5 times). All these features are integers.
Punctuations Statistics
The features include the number and ratios of the following punctuations: [', -, ;, :, ., !, ?, (,), [,], {, }, /, ", ', _]. The model incorporates the percentage of characters that are one of the 17 different punctuation marks shown above. These features are integers. It also includes the total punctuation percentage, which is the sum of these individual percentages. These values are between 0 and 1.
TF-IDF Vectorizer
The model uses the TF-IDF vectorizer provided by ski-kit as another feature set. Term frequency (TF) measures how often a word appears in a specific document as shown in Equation 1. A higher frequency suggests greater importance within that document. Inverse document frequency (IDF) captures the term popularity as an inverse of the overall corpus as shown in Equation 2. Our TF-IDF matrix has a max of 10000 features.

2.2 New Dataset

The new dataset contains 15 AI generated essays using ChatGPT 5.1 and 13 human written essays. This dataset is put through the same preprocessing steps as the DAIGT V2 Train Dataset and evaluated using the trained model. The table of the prompts used for the essay can be found in Table 2. The human essays were sourced from multiple websites, which can be found in Table 3. The essays vary in quality, with some being used as examples of excellent essays, while others scored low in their respective rubric.

3 Features and Inputs

Our feature design focused on capturing different characteristics of essay writing, from structure to vocabulary. Since the raw text alone is not expressive enough for classification, we extracted several sets of features that highlight different aspects of how the writing is formed. The first group we added was punctuation-based features. This came from our intuition that AI generated essays tend to use certain punctuation marks more consistently. For each punctuation symbol, we computed its ratio relative to the total character count, which resulted in 18 punctuation features. After calculating them, we noticed clear differences in the averages. For instance, AI essays used noticeably more commas, dashes, brackets, and colons, while human essays used more semicolons, question marks, and slashes. These differences are small individually, but when combined, they capture stylistic habits that help distinguish the two types of writing.

We also included a small set of word-based features, such as word count, repeated words, and uncommon words. These three features measure basic writing complexity and repetition patterns. Our reasoning was that AI models sometimes repeat ideas more or use more polished vocabulary compared to students. These features did not perform well on their own, but they added small improvements when combined with the other feature groups. The main representation of the essay text came from TF-IDF with unigrams and bigrams. We chose this approach because it is lightweight to compute and captures both individual words and short phrase patterns. TF-IDF turns the essay into numerical weights, and in our case it produced 10,000 dimensions. This became the largest feature set and also the most informative. Most of the model's performance came from these phrase frequency patterns, which seemed to reflect differences in how AI models structure sentences compared to humans.

To vary the inputs and compare their impact, we trained models using several combinations:

punctuation only, TF-IDF only, TF-IDF with punctuation, and TF-IDF with punctuation plus the word statistics. We also experimented once with SBERT embeddings, since they create dense sentence level vectors that might capture meaning more directly. However, SBERT did not improve accuracy, was much slower to compute, made the pipeline more complicated, and did not help with detecting newly generated GPT-5 essays. This may be because our training data came from older AI models, and the SBERT embeddings did not shift the decision boundary enough to adapt to newer writing styles. In practice, the classifier's predictions on GPT-5 essays were almost unchanged whether SBERT was included or not. Because of this, we removed SBERT from our final setup. In the end, our model uses all three hand engineered feature sets together, with TF-IDF contributing most of the predictive power and the other features adding smaller signals related to style and writing habits.

4 Implementation

Our implementation went through several stages because we tested multiple models and several feature combinations before choosing the final setup. Before comparing models, we checked the label distribution in the dataset and found that about 61 percent of all essays were human written, giving us a simple majority baseline of roughly 0.61 accuracy. Alongside this, we used a punctuation only model as a stronger feature baseline, which reached about 0.83 accuracy and became our main comparison point when evaluating different feature combinations.

At the start of the project, we experimented with an MLP classifier to see whether using a neural network from the beginning would give us a stronger starting point. Once we compared it with Logistic Regression, which we had originally planned to use based on our proposed solution and related work, we found that the accuracy was almost the same. Logistic Regression trained much faster and had already been used successfully in similar text classification setups that relied on TF IDF with N grams. Because the performance was similar and the training speed was much better, we continued with Logistic Regression during the early stages of the project.

After the progress report, we updated the classifier to Linear SVC. Support Vector Machines are commonly used for text classification and tend to perform well with sparse TF IDF features, which fits our preprocessing pipeline. Linear SVC optimizes a hinge loss objective to find a separating boundary between the two classes, and it trains efficiently even with high dimensional inputs. This made it a suitable choice for our final model.

We also attempted to implement an LSTM model, as originally planned in our proposal, but our preprocessing pipeline was not designed for sequence-based architectures. TF-IDF transforms text into an unordered vector, which does not align with the sequential input format required for LSTM. Converting our entire pipeline to support sequential models would have required major restructuring, and our initial attempts did not run successfully, so we decided not to continue with that approach.

To understand how much each type of feature contributed to performance, we ran several experiments with different combinations of inputs. These included punctuation only, word statistics only, TF IDF only, TF IDF with punctuation, and TF IDF with punctuation and word statistics. TF IDF alone performed the best, reaching almost perfect accuracy on the validation and test sets. The word statistics model performed poorly on its own, while punctuation alone provided a stronger baseline. Combining everything gave results similar to TF IDF alone, but the smaller feature sets helped with stability, so we kept them in the final configuration.

Once we had a model that performed very well on the original dataset, we tested it on a new set of essays that included human written samples from online sources and AI generated samples produced by GPT 5.1. This allowed us to evaluate generalization rather than relying only on the original training distribution. The performance on the new dataset dropped noticeably compared to the original test set. Since the new essays came from newer AI models, we suspected the drop was related to dataset shift rather than a failure of the classifier itself. This motivated us to try SBERT embeddings, since they create dense

sentence level vectors that might capture more meaning than simple TF IDF features. However, adding SBERT did not improve the results, and the predictions were almost unchanged. Training time also increased significantly and took around half an hour. We believe this is because the model was already overfitted to features from older AI generated essays in the training dataset, so adding SBERT could not shift the decision boundary enough to help with GPT 5 style writing. Since SBERT did not reduce the error on the new dataset and made the pipeline much slower, we removed it.

Our final model uses Linear SVC with TF IDF, punctuation features, and word statistics. This version consistently outperformed both the majority baseline and the punctuation only baseline. Although it did not generalize well to essays written by GPT 5, this appears to be due to changes in writing style across different generations of large language models rather than an issue with the classifier. Within the scope of the training distribution, the final setup gave us the strongest and most stable performance.

5 Evaluation

The total training dataset [see 2.1] was split into 80/10/10 for training, validation, and test respectively. Cross-validation was not used as we had sufficient datapoints, almost 45 thousand, to adequately train the model. Cross-validation in this case would've been too computationally expensive for relatively small benefits to reduce variance of the training sets. For the test set, we use the same evaluation methods as our progress report, where we evaluate the accuracy, precision, recall and f1-score of both the validation and test sets after training. However, since we are still only evaluating from the training dataset, the results have become less relevant, as most of our models would have almost 100% accuracy and F1 scores. Following the restructuring of our model to allow us to pass in data frames and predict our model on new data. Our model evaluation strategy evolved to testing the model against the new dataset [see 2.2] while using the same metrics of accuracy, precision, recall and F1-score. The metrics were adequate, as the results would vary significantly against the new dataset, allowing us to accurately access the performance of our models against each other.

6 Progress

We followed through 50% of our plan from the progress report. We brought in new data [see 2.2] to further test our model's accuracy. This helped the team identify model performance, as our earlier evaluation strategy was proving irrelevant, as all models scored over 98% accuracy from the test set [see 2.1]. We ended up not implementing k-fold cross validation as it would be too computationally expensive for relatively limited improvements as we had an adequate dataset size. Moving to the model itself, we ended up not implementing a Neural Network. Although we did test using a Neural Network, we used Multi-layer Perceptron (MLP) instead of Long Short Term Memory (LSTM). The main cause of that was due to our TF-IDF vectorization, LSTM requires a sequence input, however our current model preprocessing converts that into a un-sequenced vector. This distinction makes our current structure incompatible with LSTM, we would've had to completely rebuild our model's preprocessing, pipeline and feature sets. However, even with MLP, the training time was extremely long, and the resulting accuracy was very similar to the model we ended up going with the Linear Support Vector Classification (SVC) model since we were using TF-IDF and the accuracy was almost identical to MLP while being much faster, which is what we ended up with.

7 Error Analysis

We systematically examined the model's errors in two stages, using accuracy, precision, recall, F1-score and Confusion Matrix. Initially, the Linear SVC model was tested on its original training distribution (DAIGT V2), where it achieved 99.7% accuracy [see Table 4 and Table 5]. This result signaled overfitting. To expose its true limitations, a critical second stage involved testing on a new external dataset, designed to simulate Data Drift, featuring essays from the advanced ChatGPT 5.1 (GPT-5) and authentic human essays. This external test caused the model's performance to plummet to 61.8% [see Table 6], highlighting the profound difference between its in-distribution and real-world generalization capabilities.

The model's strength lies in its specialized ability to detect the specific statistical signatures and stylistic artifacts of older AI texts that were present

in the DAIGT V2 training data. It excelled at separating those initial AI generations from the original human essays. However, its profound weakness is a complete inability to generalize to newer, more sophisticated LLM output.

The core error pattern identified was a catastrophic bias towards classifying all text as AI-generated [see Figure 3]. This pattern is supported by key figures from the external evaluation: the model produced 13 False Positives (FP) and 0 True Negatives (TN). This means the model incorrectly flagged every single human-written essay in the new dataset as being AI-generated. This occurred because the decision boundary, relying on fragile TF-IDF and punctuation features, was pushed so far by the highly human-like GPT-5 text that authentic human essays now fell on the misclassified AI side of the boundary.

To specifically address the fragility of the TF-IDF and punctuation features and the model's poor generalization, future development should focus on two key areas:

1. Update Database: Incorporating text generated by the latest, most sophisticated LLMs (GPT-5) to teach the detector their new statistical and stylistic signatures.

2. Implement Deep Neural Networks with Dropout Regularization: Given the complexity of distinguishing advanced AI from human text, we must move beyond linear models. Implementing a deep Neural Network architecture (such as an LSTM or Transformer) combined with Dropout is essential. This prevents complex co-adaptations on the training data, effectively mitigating the severe overfitting observed in the current model and promoting generalization across diverse writing styles.

8 Team Contributions

Fei: Restructured the model to allow users to pass in new datasets and predict using trained model. Created the new dataset used in the new evaluation strategy. Wrote sections Evaluation and Progress. Updated sections Introduction and Dataset.
Ghena: Implemented MLPClassifier and tried to implement and tried to implement KerasClassifier with dropout, however there wasn't enough CPU space and couldn't run. Wrote the Error Analysis section.
Haniye: Explored SBERT and LSTM extensions to the model and wrote the Features and Inputs section as well as the Implementation section of the report.

References

Argumentative essay examples. https://blog.prepscholar.com/argumentative-essay-examples. Accessed: 2025-12-01.

Argumentative essays: A step-by-step guide — skyline college library. https://guides.skylinecollege.edu/c.php?g=279231&p=2778200. Accessed: 2025-12-01.

Student samples – english 12 composition. https://www2.gov.bc.ca/assets/gov/education/administration/kindergarten-to-grade-12/exams/student-samples/en12-comp-sp.pdf. Accessed: 2025-12-01.

Faris Al-Ahmadi. Ai text detection competition. https://www.kaggle.com/code/farisalahmdi/ai-text-detection-competition. Accessed: 2025-08-03.

Alexia Audevart. Kerasnlp starter notebook llm - detect ai generate. https://www.kaggle.com/code/alexia/kerasnlp-starter-notebook-llm-detect-ai-generate. Accessed: 2025-08-03.

Nicholas Broad. 2023a. Daigt data - llama 70b + gpt4 + falcon180b dataset. https://www.kaggle.com/datasets/nbroad/daigt-data-llama-70b-and-falcon180b. Accessed: 2025-11-08.

Nicholas Broad. 2023b. Persuade corpus 2.0. https://www.kaggle.com/datasets/nbroad/persaude-corpus-2/. Accessed: 2025-11-08.

Darragh Hanley. 2023. 1000 essays from anthropic claude. https://www.kaggle.com/datasets/darraghdog/hello-claude-1000-essays-from-anthropic. Accessed: 2025-11-08.

Siti Khotijah. Using lstm for nlp: Text classification. https://www.kaggle.com/code/khotijahs1/using-lstm-for-nlp-text-classification. Accessed: 2025-08-03.

Darek Kleczek. Daigt v2 train dataset. https://www.kaggle.com/datasets/thedrcat/daigt-v2-train-dataset/data. Accessed: 2025-08-03.

Qian Li, Hao Peng, Jianxin Li, Congying Xia, Renyu Yang, Lichao Sun, Philip S. Yu, and Lifang He. 2022. A survey on text classification: From traditional to deep learning. *ACM Trans. Intell. Syst. Technol.*, 13(2).

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. *Preprint*, arXiv:2301.11305.

Jasmine Mohammad. Detecting ai-text(svm, nn) + full ml guide. https://www.kaggle.com/code/jasminemohamed2545/detecting-ai-text-svm-nn-full-ml-guide. Accessed: 2025-08-03.

Radek Osmulski. 2023. Llm generated essays. https://www.kaggle.com/datasets/radek1/llm-generated-essays. Accessed: 2025-11-08.

Alejo Paullier. 2023. Daigt external dataset. https://www.kaggle.com/datasets/alejopaullier/daigt-external-dataset. Accessed: 2025-11-08.

Muhammad Rizqi. 2023. Llm-generated essays using palm. https://www.kaggle.com/datasets/kingki19/llm-generated-essay-using-palm-from-google-gen-ai. Accessed: 2025-11-08.

Tables and figures

List of Tables

List of Figures

Equations

$$TF(w,d) = \frac{\text{count}(w,d)}{\text{total words in } d} \quad (1)$$

$$IDF(w,C) = \log\left(\frac{|C|}{|\{\, d \in C : w \in d \,\}|}\right) \quad (2)$$

Dataset Name	Description
Persuade Corpus 2.0 (Broad, 2023b)	Provides argumentative essays produced by 6 to 12 grade students. It was created by The Learning Agency and Vanderbilt University, originally pulled from the following GitHub repository.
ChatGPT (Paullier, 2023)	Contains 2.5k student written texts sourced from the FeedBack Prize 3 Competition, and 2.5k AI-generated texts using ChatGPT. The compiled dataset includes only AI-generated texts and prompts.
Llama 70b + GPT-4 (Broad, 2023a)	Contains 9k essays generated by Llama 70b and Falcon 180b. Prompts come from the Persuade Corpus and GPT4, using a total of 35 prompts for generation.
LLM Generated Essays (Osmulski, 2023)	Contains 700 essays generated by LLMs 500 from GPT3.5Turbo and 200 from GPT4.
Claude Essays (Hanley, 2023)	Contains 1000 essays generated by Claude-Instant-1 using 15 prompts from the Persuade Corpus. Prompts were sourced from the competition discussion.
PaLM Essays (Rizqi, 2023)	Contains 1384 essays generated by PaLM. Prompts were sourced from a Kaggle competition notebook.

Table 1: Summary of Datasets Used in the DAIGT V2 Train Dataset

Model	Prompt
GPT 5.1	Write an argumentative essay about cars and transportation, in the style of a 6th–12th grader, using the following essay prompt: “Today the majority of humans own and operate cell phones on a daily basis. In essay form, explain if drivers should or should not be able to use cell phones in any capacity while operating a vehicle.”
GPT 5.1	Write five argumentive essay about cars and transportation, in the style of a 6th-12th grader with the following essay prompt: "Some schools require students to complete summer projects to assure they continue learning during their break. Should these summer projects be teacher-designed or student-designed? Take a position on this question. Support your response with reasons and specific examples. "
GPT 5.1	Write five argumentive essay about cars and transportation, in the style of a 6th-12th grader with the following essay prompt: "Your principal has decided that all students must participate in at least one extracurricular activity. For example, students could participate in sports, work on the yearbook, or serve on the student council. Do you agree or disagree with this decision? Use specific details and examples to convince others to support your position. "

Table 2: Prompts Used to Generate Argumentative Essays, 5 essays were generated from each

Source	Essays Used
PrepScholar (pre)	3
Skyline College (sky)	1
Government of British Columbia (bce)	9

Table 3: Sources of Human written essays

Class	Precision	Recall	F1-Score	Support
0 (Human)	1.00	1.00	1.00	2735
1 (AI)	1.00	0.99	1.00	1748
Accuracy: 0.996877091233549				
F1 Score: 0.995983935742972				
Macro Avg	1.00	1.00	1.00	4483
Weighted Avg	1.00	1.00	1.00	4483

Table 4: Validation Evaluation Metrics

Class	Precision	Recall	F1-Score	Support
0 (Human)	1.00	1.00	1.00	2737
1 (AI)	1.00	0.99	1.00	1750
Test Accuracy: 0.9973256073100066				
Test F1 Score: 0.9965616045845271				
Macro Avg	1.00	1.00	1.00	4487
Weighted Avg	1.00	1.00	1.00	4487

Table 5: Test Evaluation Metrics (In-Distribution)

Class	Precision	Recall	F1-Score	Support
0 (Human)	0.00	0.00	0.00	13
1 (AI)	0.72	0.81	0.76	42
Test Accuracy: 0.6181818181818182				
Test F1 Score: 0.7640449438202246				
Macro Avg	0.36	0.40	0.38	55
Weighted Avg	0.55	0.62	0.58	55

Table 6: External Data Drift Evaluation Metrics (Critical Failure)

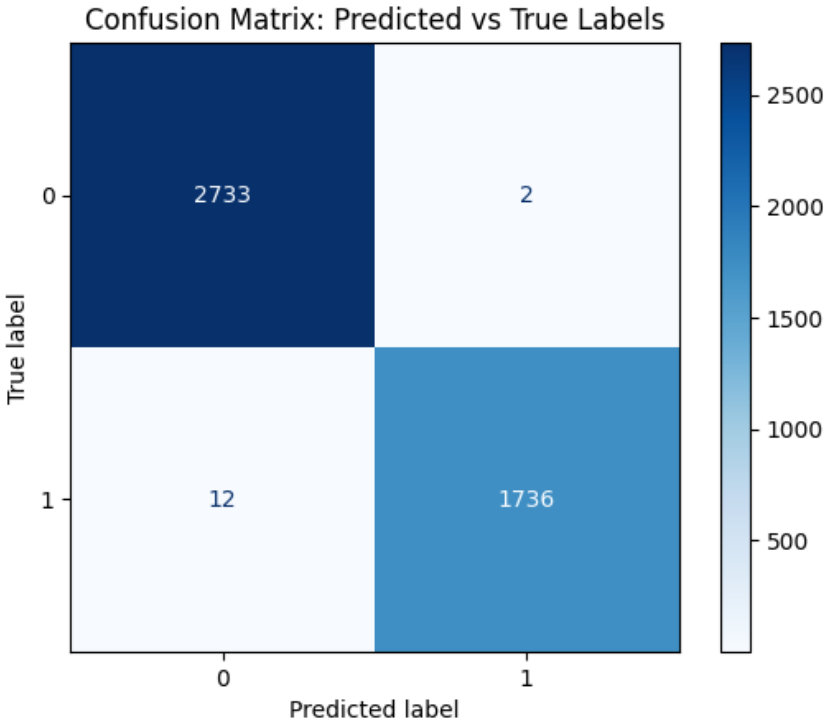

Figure 1: Confusion Matrix of the Validation Data

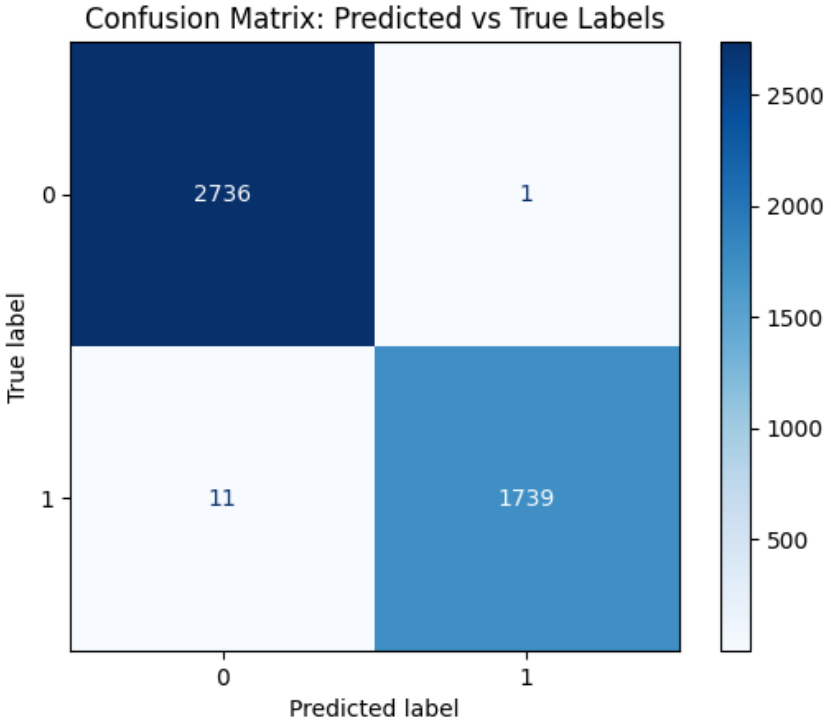

Figure 2: Confusion Matrix of the Test Data

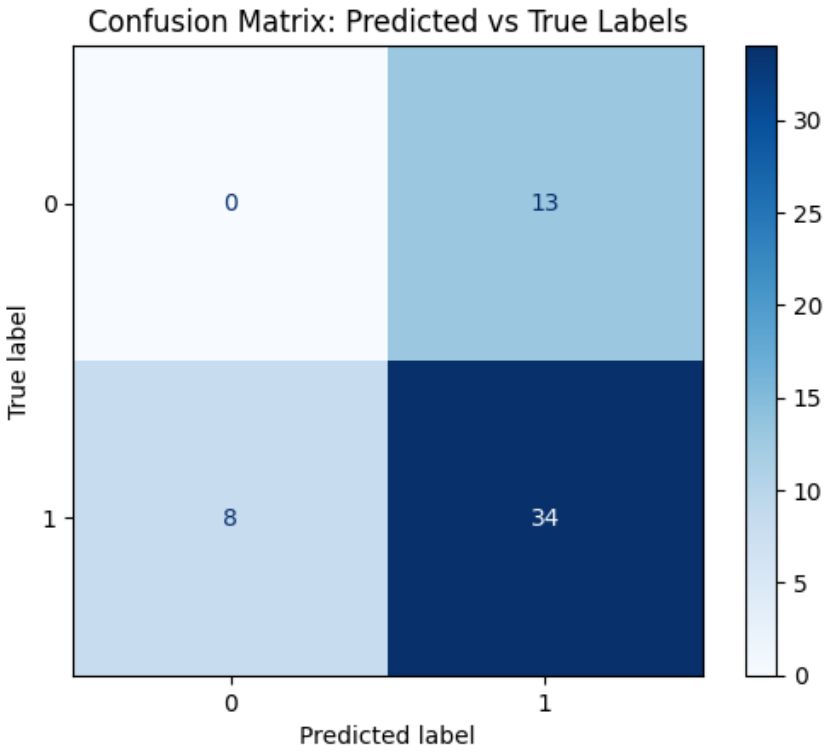

Figure 3: Confusion Matrix of the External Data Drift