VeritasVigil: Final Report

1 Introduction

VeritasVigil: The Truth Watchman is a fake news detection engine built from the ground up using a custom NLP pipeline. Our objective was to classify news (from a Kaggle FAKE+TRUE News Dataset: https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset) into real or fake news using handcrafted linguistic features, without relying on pre-built NLP libraries(like NLTK or SpaCy) for core preprocessing.

2 Custom Pipeline Design

2.1 Tokenizer Design

The tokenizer was designed to break raw news headlines into meaningful units while addressing:

- **Informal language** (e.g., "gonna", "lemme").
- **Contractions** using pattern matching (e.g., 'can't' → 'can','not').
- **Emoticons and symbols** (for example, :) : (: P), extracted via RegEx.
- **Punctuation splitting** while preserving sentence structure.
- **Repeated-character normalization**: Words like 'soooooo scary' are reduced to "so <REPEAT:5>scary".

This normalization process helps reduce vocabulary sparsity while retaining expressive intent.

2.2 Mini POS Tagger

A minimalistic part-of-speech tagger was constructed using:

- **Suffix-based rules** (e.g., words ending in '-ing' or '-ed' tagged as verbs).
- **Digit detection** for numerals.
- **Regex-based classification** for common patterns (e.g., punctuations, emojis).
- **Fallback to 'X'** for unknown or ambiguous words.

This allowed the lemmatizer to operate with greater contextual awareness.

2.3 Custom Lemmatizer Design

Our lemmatizer used POS tags to perform rule-based reductions:

- **Verb lemmatization**: Regex and rules convert forms like 'running', 'played', 'goes' into base verbs ('run', 'play', 'go').
- **Irregular verb dictionary**: Including mappings such as 'went' → 'go', 'bought' → 'buy'.
- **Pronouns, punctuation, and numbers** are retained without changes.
- **Fallback**: If no rule matches, the original token is returned.

This lemmatizer improves semantic consistency across syntactic variations in news headlines.

3 Feature Engineering

We extracted the features using the following:

- TF-IDF Vectorization (Scikit-learn)
- Stop word removal using a custom list of high-frequency non-informative tokens.

4 Visualizations (Custom Pipeline-VeritasVigil)

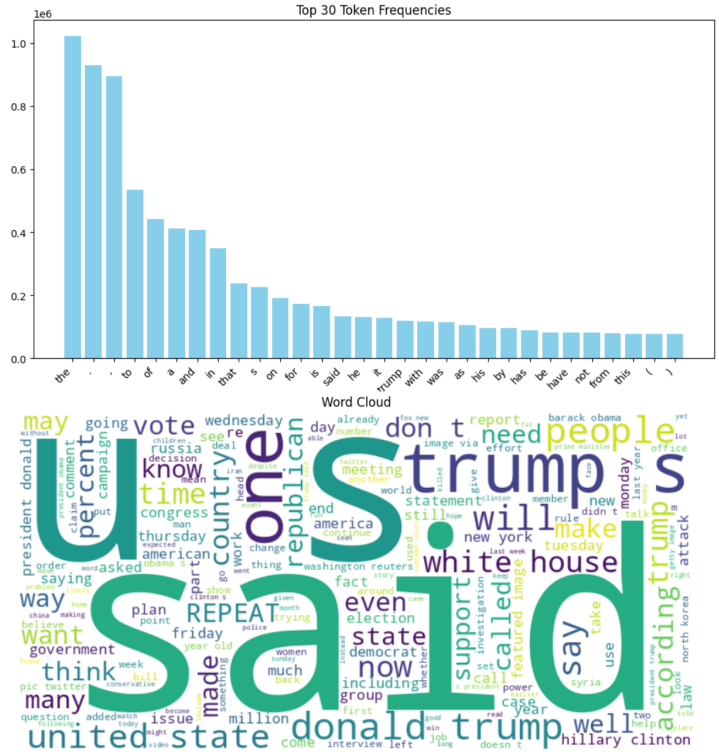

Figure 1: Token Frequency for top 30 tokens and Word Cloud for Custom Pipeline

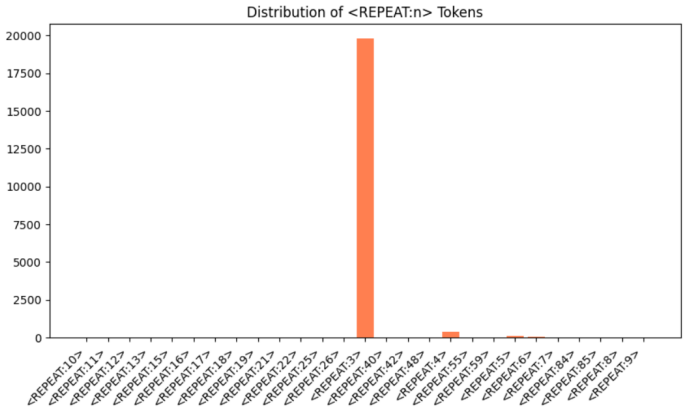

Figure 2: <REPEAT:n>Token Distribution for Custom Pipeline

5 Classification Results

5.1 Custom Pipeline Results

Naive Bayes (Custom):

```
Accuracy/F1 Score: 0.97
Precision/Recall: 0.96 to 0.98
```

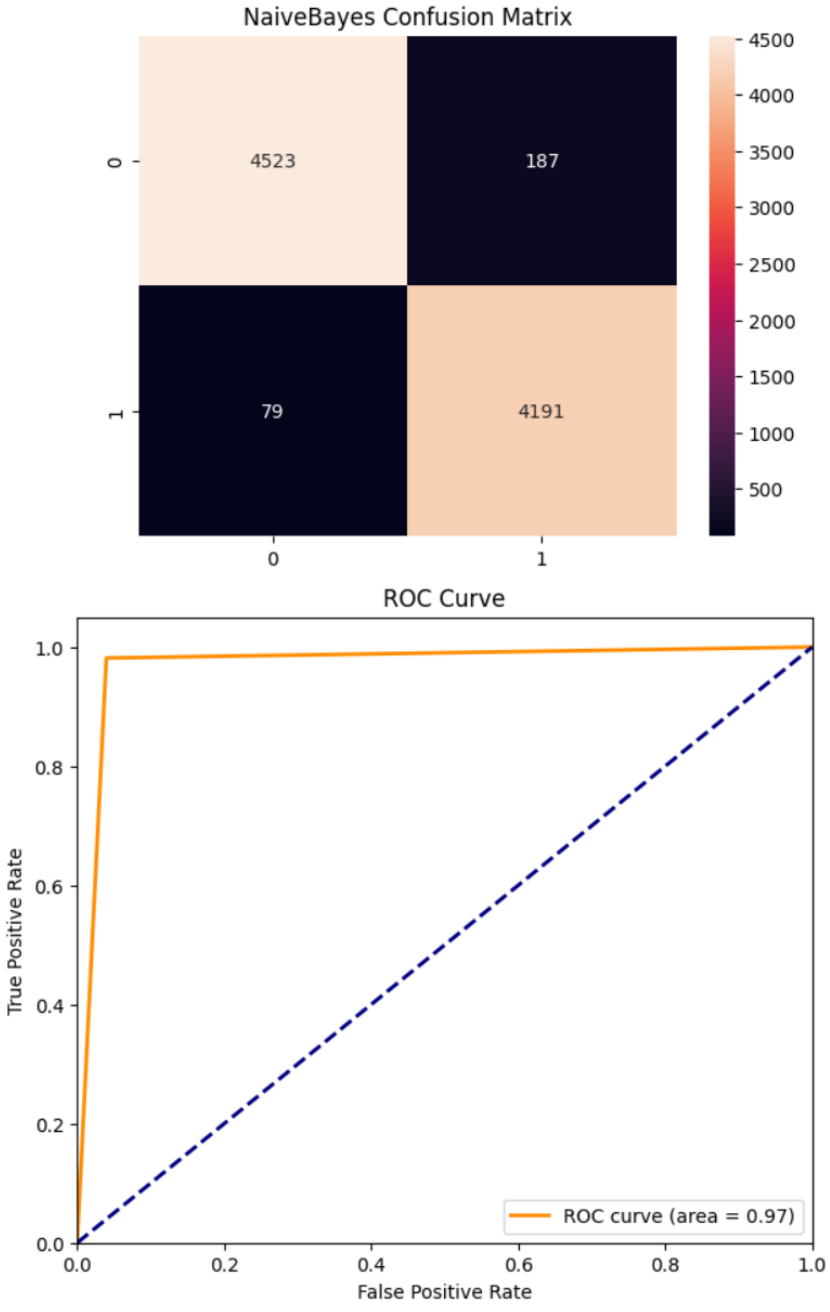

Figure 3: Confusion Matrix and ROC Curve for Custom Pipeline (NaiveBayes)

SVM (Custom):

```
Accuracy: 1.00
Precision/Recall/F1: 1.00
```

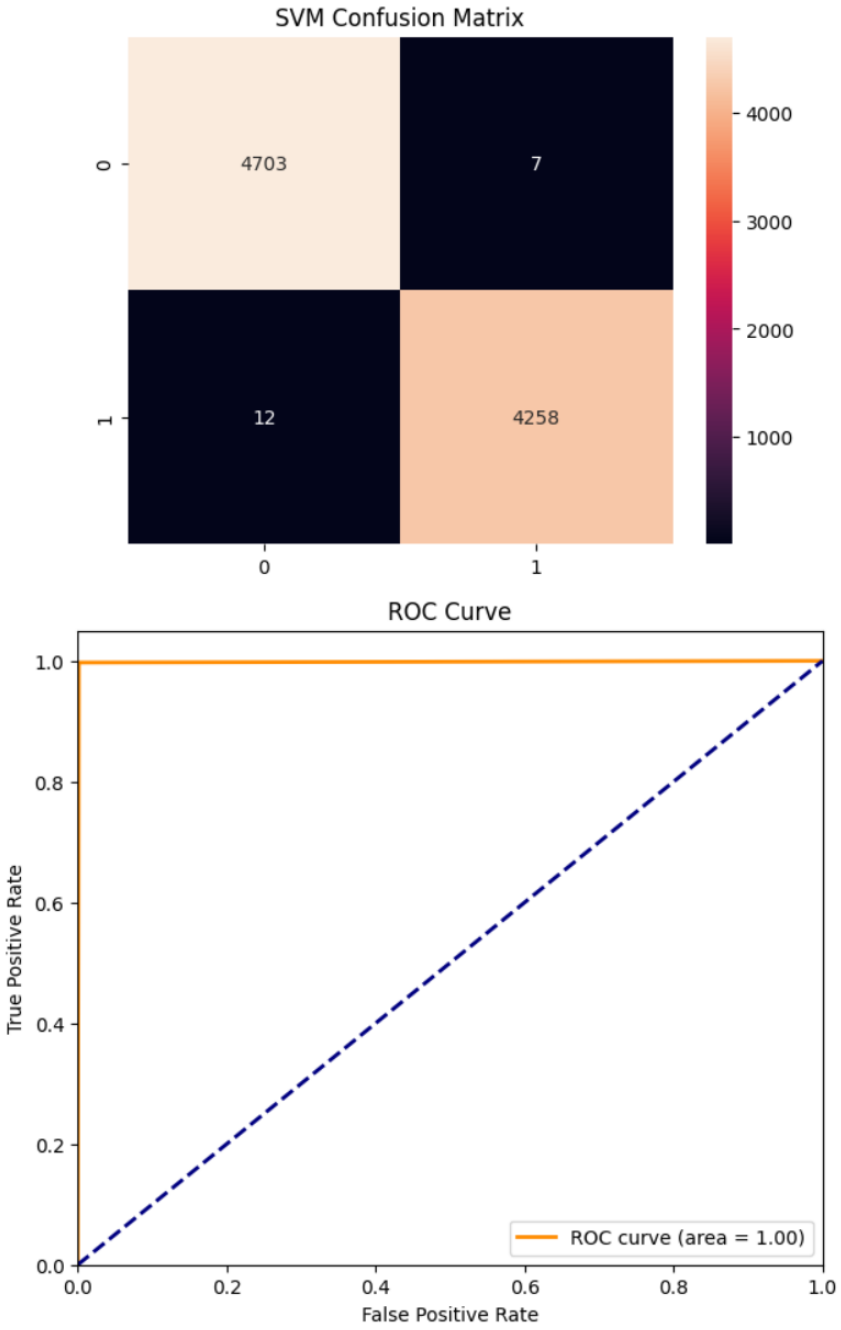

Figure 4: Confusion Matrix and ROC Curve for Custom Pipeline (SVM)

5.2 Off-the-Shelf Pipeline (NLTK + Scikit-learn)

Naive Bayes (Off-the-Shelf):

```
Accuracy: 0.93
F1 Score: 0.93
```

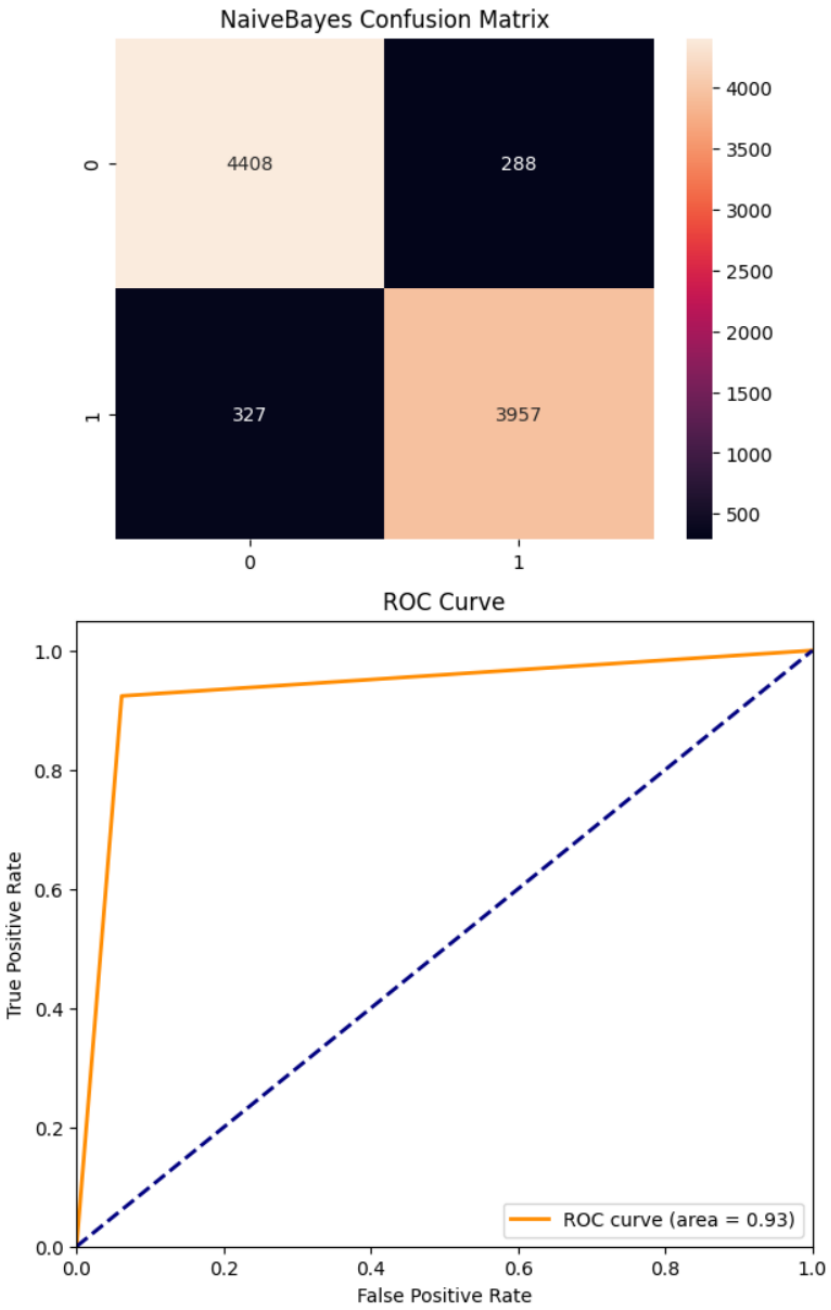

Figure 5: Confusion Matrix and ROC Curve for Off-the-shelf Pipeline (NaiveBayes)

SVM (Off-the-Shelf):

```
Accuracy: 0.99
F1 Score: 0.99
```

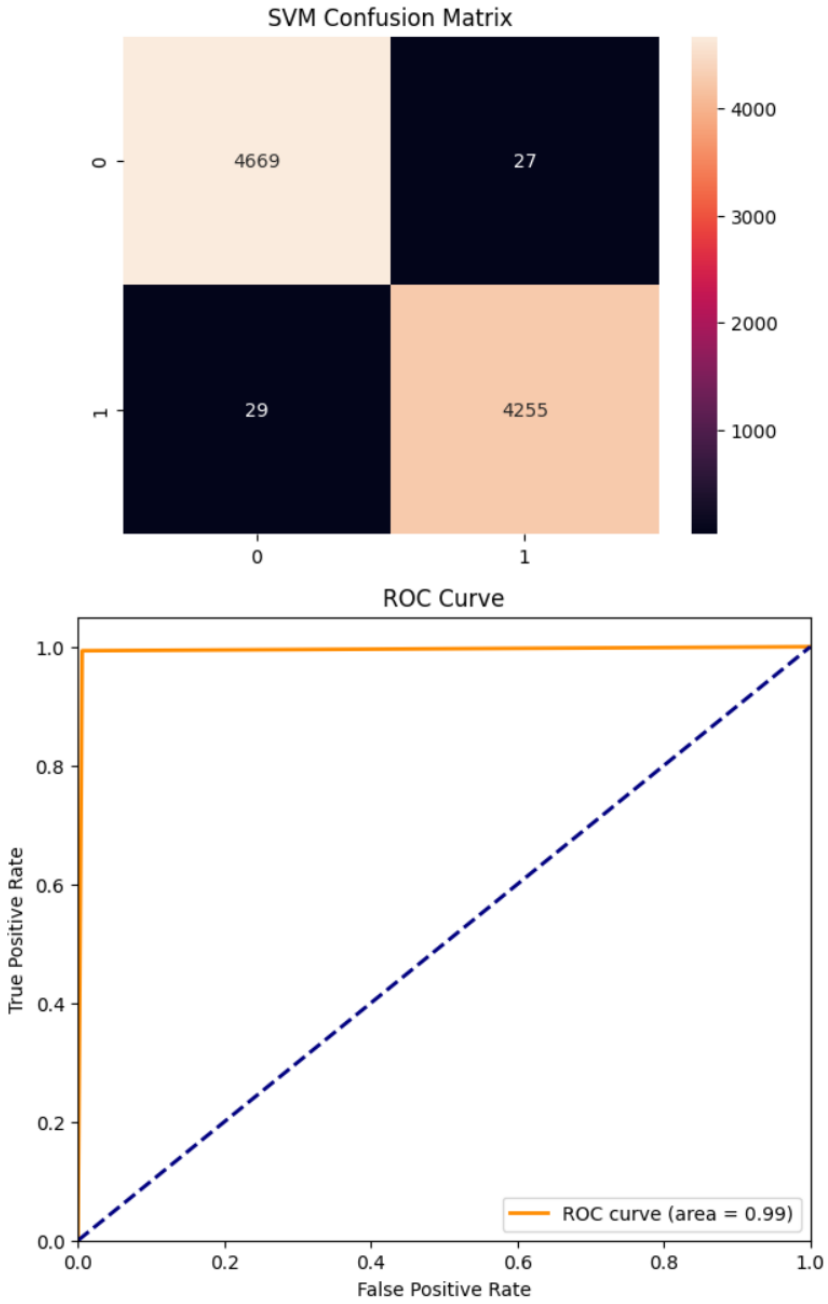

Figure 6: Confusion Matrix and ROC Curve for Off-the-shelf Pipeline (SVM)

6 Impact Analysis

6.1 Effect of Repeated-Character Normalization

- **Before normalization**: Elongated words like "soooo", "nooooo" created token sparsity and false sentiment signals.
- **After normalization**: Collapsing repeated letters improved frequency-based models by reducing redundant vocabulary entries, and the <REPEAT:n>tag preserved expressive intent.
- **Outcome**: Improved classification accuracy and faster convergence in both Naive Bayes and SVM models.

6.2 Effect of POS-Guided Lemmatization

- Lemmatization using coarse POS tags significantly reduced noisy lexical variants, such as “plays”, “playing”, “played” → “play”.
- Helped map headlines with syntactic variations to the same semantic meaning, thereby boosting TF-IDF overlap.
- **Outcome**: This reduced vocabulary size, enhanced feature density, and improved model generalizability.

7 Comparative Analysis

The custom pipeline, while less sophisticated than pretrained NLP tools derived from popular open source python libraries like NLTK, achieved high accuracy through domain-aware linguistic preprocessing. Repeated-character normalization and rule based POS-aware lemmatization improved generalization for informal and deceptive news texts.

The off-the-shelf models were slightly less accurate but far faster to implement. They serve as a solid benchmark and validate the core assumptions of the problem and this fascinating project-VeritasVigil.

8 Conclusion

VeritasVigil demonstrates the viability of handcrafted NLP pipelines in identifying fake news from a dataset of news headlines. While modern libraries streamline workflows, rule-based logic allows deeper control and interpretability—critical in sensitive domains like fake news detection.