COMP 3608 - Intelligent Systems Project

Anees Wahid	816030569
Saleem Wahid	816030374
Jaheem Edwards	816035606

Table of Contents

Identification of the Problem

Fake news has emerged as a major concern globally, particularly in critical domains such as politics, health, and finance. With the rapid spread of misinformation across social media and digital platforms, the ability of individuals to discern credible news from falsehoods has become increasingly compromised. For instance, a study in 2023 revealed that over 40% of news consumers in the United States encountered false information about COVID-19 within a randomly selected week. Similarly, 45% of Slovakian participants reported exposure to misleading news about the war in Ukraine during the same period.

Detecting fake news can be straightforward when the headline is blatantly misleading; however, in many cases, careful analysis of the full article is required to determine its authenticity. Given this challenge, our group aims to develop an intelligent system capable of automatically analyzing news articles and classifying them as either true or false. Such a system has the potential to support media consumers, regulators, and platforms in the ongoing effort to combat misinformation effectively and reliably.

Problem Classification and Selection of Intelligent Algorithms

The task of identifying whether a news article is real or fake is a classification problem. In most cases, this fits naturally into a binary classification setup, where each article is labeled as either *true* or *false*. However, one of the datasets used, the LIAR dataset, includes six labels ranging from "pants on fire" to "true", allowing us to also explore the problem as a multiclass (multinomial) classification task.

This gave us the opportunity to compare both binary and multiclass approaches during our experimentation. While our main focus remained on binary classification, the multiclass version provided insight into how fake news can vary in severity and truthfulness, making it a valuable angle to investigate.

To approach this problem, we selected three different machine learning models:

- **Logistic Regression with TF-IDF**: A simple, interpretable model that works well for text classification when paired with TF-IDF for feature extraction. It served as a strong baseline.

- **XGBoost**: A tree-based ensemble method known for its accuracy and ability to capture complex relationships in data. It's particularly effective for structured features and classification tasks.

- **BERT**: A deep learning model based on transformers, pre-trained on large text corpora. Unlike traditional models, BERT understands word context and meaning, which makes it well-suited for handling nuanced language in news articles.

These models were chosen to reflect a mix of traditional and modern approaches, each bringing different strengths to the table. Exploring both binary and multiclass setups gave us a broader view of how classification strategies can be applied to the fake news problem.

Modeling as an Optimization Problem

In the context of fake news detection, the task can be framed as an optimization problem. The objective is to find a model that minimizes the classification error, ensuring that news articles are accurately labeled as either true or fake.

Let the objective function Z represent the total classification error across all samples. Specifically, for a dataset of news articles with true labels y_i and model-predicted probabilities y'_i, the goal is to minimize the discrepancy between y_i and y'_i.

$$Z = \frac{1}{N}\sum_{i=1}^{N} I(y'_i \neq y_i)$$

Where:

- N is the number of samples
- y'_i is the predicted label for the i-th sample
- y_i is the actual label for the i-th sample
- $I(\cdot)$ is the indicator function, which returns 1 if the condition is true and 0 otherwise

In Logistic Regression, this is typically achieved by minimizing the cross-entropy loss function:

$$Z = -\sum_{i=1}^{N} [y_i log(y'_i) + (1 - y_i) log(1 - y'_i)]$$

This loss penalizes incorrect predictions more severely the more confident the model is in them, encouraging the model to produce well-calibrated probability estimates.

For XGBoost, the optimization objective includes both a differentiable loss function (such as logistic loss for binary classification) and a regularization term to control model complexity. The objective function for XGBoost is:

$$Z = \sum_{i=1}^{n} L(y_i, y'^{(t)}_i) + \sum_{k=1}^{t} \Omega(f_k)$$

Where:

- L is the loss function (eg logistic loss)
- $\Omega(f_k)$ is the regularization term for tree f_k
- t it the number of boosting rounds (trees)

The algorithm minimizes this composite objective using gradient boosting, where each new tree corrects errors made by the previous ensemble.

For BERT, the optimization problem can be framed as minimizing the cross-entropy loss function, which is commonly used for classification tasks. During fine-tuning on a specific task, the model's parameters (weights) are updated to minimize the difference between the predicted and actual labels, effectively learning task-specific patterns.

Here is the formula for cross-entropy loss, which is the objective function typically minimized during BERT's fine-tuning:

$$Z = L(y', y) = -\frac{1}{N}\sum_{i=1}^{N}[y_i log(y'_i) + (1 - y_i) log(1 - y'_i)]$$

Where:

- N is the number of samples
- y_i is the true label for the i-th sample (0 or 1)
- y'_i is the predicted probability of the sample being in the positive class (label 1)
- $L(y', y)$ is the loss function (cross-entropy loss)

Thus, the optimization goal for each model is to fine-tune its parameters to minimize the classification error, ensuring the most accurate fake news detection possible.

Datasets and Experimental Design

We used three Kaggle datasets for this project, each focused on fake news detection but with slightly different formats:

- **Fake News Dataset –** full news articles and corresponding headlines for fake news.
- **Real News Dataset –** full news articles and corresponding headlines for real news
- **LIAR Dataset –** consists of short political statements with six different truthfulness labels ranging from "pants on fire" to "true." This was the only dataset used for multiclass classification, while the others were used in binary classification.

Preprocessing

To prepare the data for modeling, we first concatenated the headline and article body (for the Fake and Real datasets) into a single string, separated by a hyphen ("headline - content") and added label 1 for all rows in the real dataset and label 0 for all rows in the fake dataset. The LIAR dataset already came as a single short statement, so no concatenation was needed there.

Next, we applied text cleaning tailored to the model being used:

- **Logistic Regression and XGBoost**
    - Removed URLs
    - Stripped punctuation
    - Converted text to lowercase

- **BERT**:
    - Removed URLs
    - Preserved punctuation and capitalization, since they are important for BERT's contextual understanding

Four datasets were derived as follows from merging all 3 datasets into one unified dataset:

- **Binary classification** preprocessed for **Logistic Regression and XGBoost** (multinomial labels classified as either true or false, classification labels are 0 - 1)
- **Multinomial classification** preprocessed for **Logistic Regression and XGBoost** (multinomial labels preserved, classification labels are 0 - 5)
- **Binary classification** preprocessed for **BERT** (multinomial labels classified as either true or false, classification labels are 0 - 1)
- **Multinomial classification** preprocessed for **BERT** (multinomial labels preserved, classification labels are 0 -5)

Experimental Strategy

Our experiments were designed to fairly compare the performance of different models across consistent input and output formats:

- All models were trained on the **concatenated text inputs with a 80 / 20 split**
- We tested three algorithms: **TF-IDF + Logistic Regression**, **XGBoost**, and **BERT**
- Each model was evaluated using **accuracy**, **F1-score**, and **confusion matrices**

This setup allowed us to focus on how different modeling approaches handle the fake news detection task, using the same cleaned and prepared data throughout.

Implementation Layout

Our project followed a structured file organization to keep all components modular and easy to manage. Below is a breakdown of the main folders and their purpose:

- **`/datasets`**
  Contains the original, unmodified datasets exactly as downloaded from Kaggle. These were kept untouched for reference and reproducibility.

- **`/cleaned_datasets`**
  Holds all cleaned and preprocessed datasets used for training and testing the models. This separation helped ensure we didn't overwrite or mix raw and processed data.

- **`/notebooks`**
  Includes Jupyter notebooks used for data analysis, model training, evaluation, and experimentation. Each notebook is named based on its task for clarity (e.g., `bert.ipynb`, `xgboost.ipynb`, `data_cleaning.ipynb`).

- **`/models`**
  Stores the trained models exported in .**pkl** format. These models are used directly in the Streamlit app for predictions without needing to retrain.

- **`/src`**
  Contains the Streamlit app that allows users to interact with the classifier through a simple UI. The app loads a trained model from the `/models` folder and processes user input for real-time prediction.

This structure kept the project organized and made collaboration and debugging much more manageable throughout development.

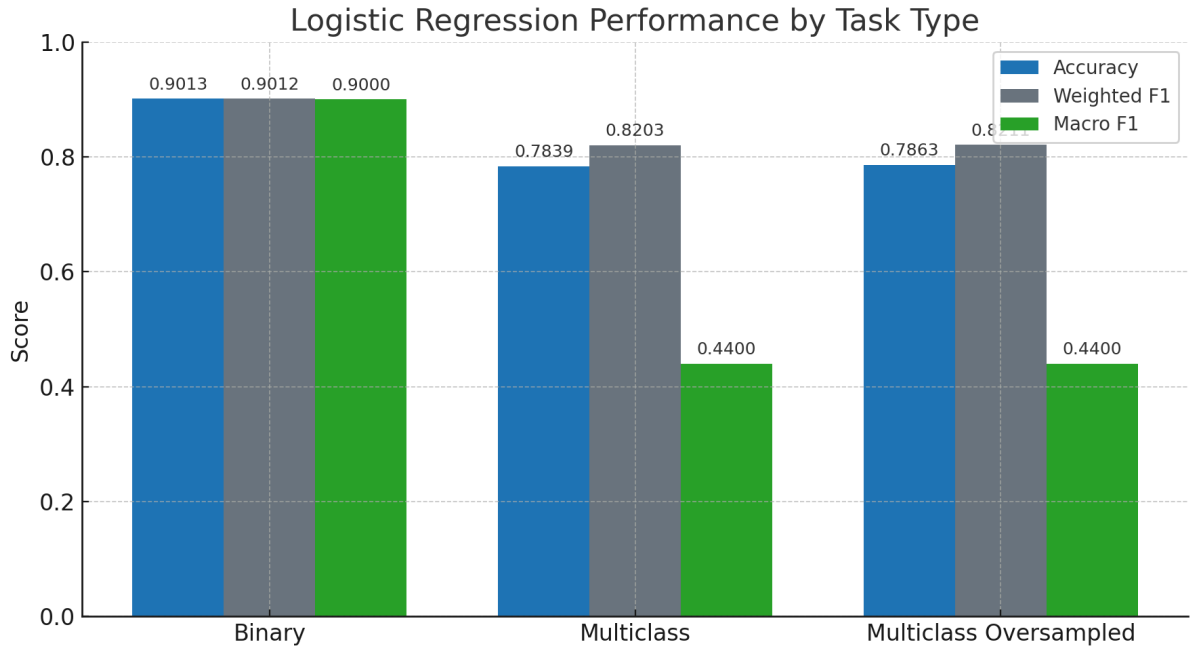
Links

GitHub Repository: https://github.com/Intelligent-Systems-Team-1/Fake-News-Detector

Trello:
https://trello.com/invite/b/67e2d0ce356f9f3629f3410d/ATTI76f75761dd57cb13172d3313f3465fb6695D42A4/intelligent-systems-project-team-1

Discussion and Interpretation of Results

Logistic Regression Model



Binary Classification

Logistic Regression performed surprisingly well for a baseline model. With an accuracy of **90.13%** and a weighted F1-score of **0.9012**, it consistently predicted both fake and real news accurately. Precision and recall were balanced across both classes, indicating good generalization and well-calibrated predictions.

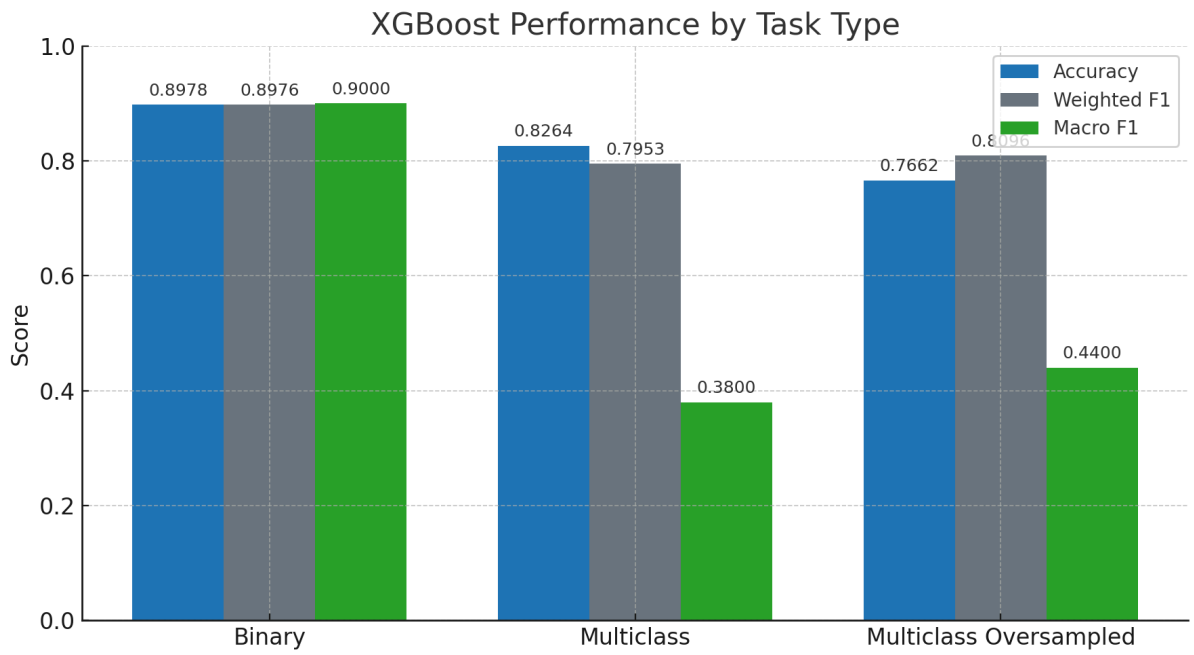
Multiclass Classification

When extended to the 6-class setup from the LIAR dataset, Logistic Regression's performance declined. Accuracy dropped to **78.39%** and the weighted F1-score fell to **0.8203**. Most of this performance was carried by the two dominant classes (labels 0 and 1). Minority classes (2–5) received much lower F1-scores, revealing a clear issue with class imbalance and limited model expressiveness.

Multiclass with Oversampling

After applying random oversampling to balance the class distribution, accuracy increased slightly to **78.63%** and the weighted F1-score reached **0.8211**. Although the overall metric shift was minimal, this slight improvement suggests better recognition of minority classes. However, macro F1 remained low, implying the model still couldn't learn distinct patterns for less frequent labels. The lack of contextual understanding in TF-IDF likely contributed to this plateau.

XGBoost Model



Binary Classification

XGBoost performed well in the binary task, achieving **89.78% accuracy** and a **weighted F1-score of 0.8976**. Its strength lies in capturing non-linear relationships and patterns in word distributions, and it handled both classes with solid balance. Although slightly behind BERT and Logistic Regression in F1, it remained competitive.

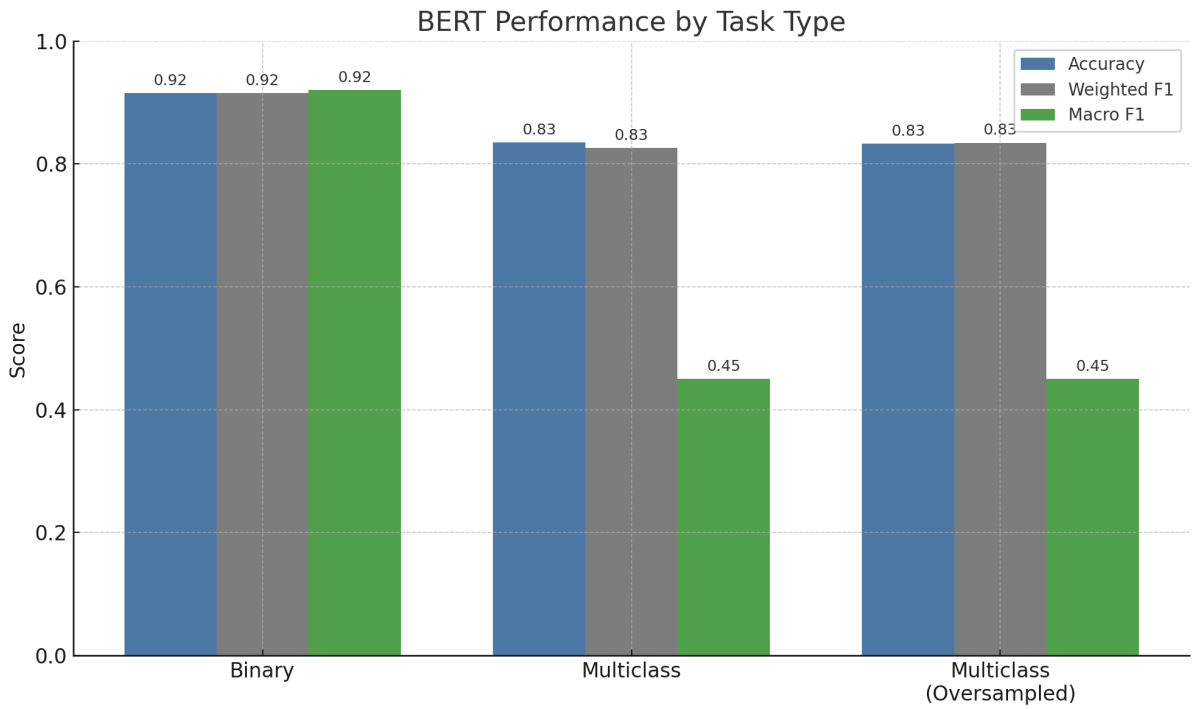
Multiclass Classification

In the 6-class setup using the LIAR dataset, XGBoost reached **82.64% accuracy**, the highest among the three models tested so far. However, its **weighted F1-score dropped to 0.7953**, and **macro F1 to just 0.38**, reflecting poor performance on the minority classes. This suggests the model made strong predictions for the majority classes but frequently misclassified the rest.

Multiclass with Oversampling

After oversampling all classes equally, **accuracy decreased to 76.62%**, but **weighted F1 improved to 0.8096**, and **macro F1 rose to 0.44**. This indicates that oversampling helped XGBoost be more inclusive of minority classes—at the cost of some overall correctness. The model handled the tradeoff better than Logistic Regression, suggesting tree-based methods are more adaptable to imbalanced label distributions.

BERT Model



Binary Classification

BERT delivered the strongest performance across all models in the binary classification task. It achieved an accuracy of **91.52%** and a weighted F1-score of **0.915**, demonstrating excellent balance between precision and recall for both fake and real news. Its deep contextual understanding gave it an edge over TF-IDF-based models, especially in cases where language was subtle, sarcastic, or ambiguous.

Multiclass Classification

Even when tasked with predicting six different truthfulness labels from the LIAR dataset, BERT maintained strong performance:

- **Accuracy**: **83.49%**
- **Weighted F1-score**: **0.826**
- **Macro F1-score**: **0.45**

It continued to perform well on dominant classes (labels 0 and 1 with F1 ≈ 0.92), but performance dropped for lower-support classes 2 through 5), reflecting difficulty in distinguishing between nuanced truth categories. Still, BERT outperformed both Logistic Regression and XGBoost in this more complex setting.

Multiclass with Oversampling

After applying random oversampling to rebalance the dataset, BERT achieved:

- **Accuracy**: **83.30%**
- **Weighted F1-score**: **0.835**
- **Macro F1-score**: **0.45**

These results show that oversampling helped improve F1-scores for some minority classes (like 2 and 4), but the overall macro F1 remained limited. This indicates that while BERT benefits from more balanced training data, its ability to distinguish truthfulness beyond binary still depends heavily on label clarity and semantic separability—something oversampling alone cannot solve.

Conclusion

Throughout this project, we explored how well different machine learning models can detect fake news, comparing Logistic Regression, XGBoost, and BERT across both binary and multiclass setups.

Logistic Regression turned out to be a solid baseline for binary classification—quick, simple, and surprisingly accurate. However, it struggled once we moved to the more detailed six-class problem, where understanding subtle differences in truthfulness mattered more. XGBoost handled class imbalance a bit better, especially when we used oversampling, but still had trouble with minority classes.

BERT stood out as the top performer. It consistently gave us the best results across all setups, thanks to its ability to understand context and meaning, not just keyword frequency. Even though BERT also found the multiclass problem challenging—especially with less common labels—it still outperformed the other models by a noticeable margin.

In the end, while all three models had their strengths, BERT was the most reliable and flexible overall. For real-world fake news detection, especially when the task involves understanding subtle phrasing or truthfulness levels, deep learning models like BERT are clearly the way forward.