Anti-Scam Job Fraud Detection Report

The approach aims to employ a multi-modal fraud detection system that leverages both structured tabular data and unstructured text data, ensuring a comprehensive understanding of job posting's characteristics. This dual strategy is vital because fraudulent job ads often exhibit telltale signs in both their numerical and textual features. By combining these two data modalities, we can capture a wider array of patterns that differentiate legitimate job postings from scams.

Specifically, fraudulent job listings often reveal their deceptive nature through identifiable characteristics. For example, the absence of a company logo is a common indicator, as scammers typically refrain from providing such identifying visuals. A lack of company profile or industry details further contributes to the profile of a fraudulent listing.

Text-based clues also provide additional insights. Overly generic job descriptions, devoid of specific responsibilities raise concerns. The presence of scam-related phrases, such as "no experience required" can serve as an indicator of red flags, to a certain extent. The goal is to assess how well these combined clues distinguish fraudulent listings from legitimate ones.

Methodology

EDA for structural data

The preprocessing of the data involves several crucial steps aimed at handling missing values, transforming categorical variables, and cleaning the textual data.

For tabular data, apply transformation techniques to prepare the features for modeling. First, numerical fields such as salary range were standardized to ensure that differences in scale would not negatively impact the learning process. Binary and categorical variables were transformed into a numerical format using one-hot encoding.

Subsequently, uni-, bi- and multivariate analyses were conducted to explore the unique feature interactions across modalities. Techniques like bar plots were employed to easily compare the frequency distributions of the various features and highlight which ones may have a stronger association with fraudulent listings. For example, fraud rates by salary range across the 5 quantiles represented by bars of similar height implied fraudulent rates may be independent of salary ranges. A correlation matrix is used to examine the relationship between binary variables change and fraud, which revealed a moderate tendency for jobs with company logos to be less likely fraudulent.

Missing values are an important aspect of data preprocessing and should not be discarded without a thorough investigation. In the case of fraudulent job listings, it is critical to explore whether the absence of certain values correlates with fraudulent listings. From a pairwise chi-square test, it was observed that fraudulent job listings do exhibit higher instances of missing data across selected columns. This suggests that poorly constructed job postings are more likely to lack key information. For instance, when a job posting lacks details about employment type, it is often accompanied by missing information in other features such as industry (0.71) or required experience (0.54).

From the table showing the proportion of missing values for each of the specified columns, separated by fraudulent and non-fraudulent job listings, it highlights that features like employment type, company_profile, required_experience, show significantly more missing data in fraudulent listings, this might be a red flag for detecting fraud, as scammers often omit important details to avoid detection.

Further validate these correlations through machine learning models like XGBoost to confirm the predictive power of missing features in identifying fraud.

Data Preprocessing and Feature Engineering for Textual data
Textual features such as description, company_profile, requirements, and benefits, provide crucial information for identifying fraudulent job postings. Preprocessing these fields involves several steps to ensure that the data is clean, consistent, and in a suitable format for machine learning models.

Firstly, text cleaning involves removing unnecessary elements and standardizing the text for a more uniform text representation for further analysis. This includes,

- De-contraction: Expanding contractions to normalize text and ensure consistency.
- Character Removal: Removing special characters, hyperlinks, punctuation etc.
- Tag Removal: Removed using regular expressions to avoid interfering with the tokenization.

Next is tokenization using RoBERTa's Byte-Pair Encoding (BPE), a subword tokenization technique. This approach is particularly effective in handling rare or out-of-vocabulary words, as it splits the words into subword units based on frequency.

Subsequently, job descriptions are either padded (if too short) or truncated (if too long) to a fixed length, since input sequences for machine learning models must have a consistent length. For this pipeline, a maximum sequence length of 256 tokens is used, ensuring that the model processes all inputs consistently. Also, perform label encoding on the fraudulent column (indicating whether a job posting is fraudulent), convert into numerical values.

To address the potential class imbalance in the dataset, undersampling is employed to create a more balanced distribution of fraudulent and non-fraudulent job listings. By sampling 1500 non-fraudulent job listings (the majority class) and combining them with all fraudulent job listings, the dataset achieves a split of approximately 2:1. This avoids biases toward predicting the majority class.

Once the dataset is balanced, it is split into training, validation, and test sets. We further split the training data into training and validation sets to allow for model tuning and hyperparameter optimization. The final datasets are formatted as lists of tokenized texts and their labels.

Before feeding the data into the model, we map tokens to IDs using RoBERTa's tokenizer. This involves converting each tokenized sentence into numerical IDs and creating attention masks, which indicate which tokens are important for the model to focus on. These attention masks are particularly useful for handling padded tokens, ensuring that the model only processes valid tokens.

Finally, the processed text data is structured into PyTorch TensorDataset objects, to store the tokenized input, attention masks, and labels for model training and evaluation.

Model Training & Evaluation
A fine-tuned RoBERTa model for binary sequence classification was used. The model was initialized with the pre-trained `roberta-base` architecture, tailored for the binary classification task of detecting fraudulent job postings. To optimize the model's performance, we used the AdamW optimizer with a learning rate of 2e-5, as found that a lower learning rate is necessary to make BERT overcome the catastrophic forgetting problem (Sun et al., 2019).

The batch size was set to 16, and the model was trained for 10 epochs. To enhance the efficiency of training, the learning rate was adjusted dynamically using a linear learning rate scheduler, which ramped up during the initial phase and then decayed gradually.

The dataset was split into training, validation, and testing. The training set was used to update model weights, while the validation set was used to monitor the model's performance and prevent overfitting.

The training loop consisted of forward passes where the model predicted the probability of a job posting being fraudulent. The loss (cross-entropy function) was back-propagated to adjust model weights. The training accuracy was calculated by comparing the predicted labels with the true labels, and the model's performance was optimized accordingly. After 10 epochs, the best model, which showed the highest validation accuracy, was saved. The performance of the model on the validation set was evaluated after each epoch, and the model weights were updated only if the validation accuracy improved, ensuring that the model was not overfitting.

Analysis

To further assess the model's generalization, we evaluated it on the test set, computing the accuracy, F1 score, and classification report.

To visualize the model's training progress, the training and validation accuracy and loss were plotted across the epochs. These plots confirmed that the model was learning effectively over time, with a steady improvement in accuracy and a reduction in loss for both the training and validation sets. The model showed significant improvement across the 10 epochs. Starting with a training loss of 0.470 and a validation accuracy of 0.844 in epoch 1, the performance improved rapidly. By epoch 5, the validation accuracy reached 0.996 with a training loss of 0.0476. From epochs 6 to 10, the model continued to optimize, achieving near-perfect validation accuracy of 0.999 and a minimal training loss of 0.00209 by epoch 10. These results demonstrate the model's effective learning and strong generalization on the validation set.

In addition, a confusion matrix was generated to visualize the model's performance, it displayed high true positives and negatives, and revealed a high accuracy of 0.924.

The final trained model and evaluation results underscore the model's robustness and potential for real-world deployment in fraud detection applications.

Discussions and Limitations

Some fraud indicators might be overlooked, and the dataset may not fully represent all fraudulent job types. Undersampling, while addressing class imbalance, could discard valuable non-fraudulent data. The use of RoBERTa’s tokenization might also miss domain-specific terms, affecting detection accuracy. Lastly, though the model showed high accuracy, continuous updates will be needed to adapt to evolving fraud tactics.

Conclusion

This study shows that combining structured and unstructured data improves fraud detection in job postings, with the model achieving near-perfect accuracy.

Next steps

Design and implement a robust architecture that seamlessly integrates both structured tabular data and unstructured text data. This will involve combining machine learning models capable of processing numerical features (e.g., logistic regression, decision trees) with natural language processing (NLP) models (e.g., transformers, RNNs) for text analysis.

References

Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019, October). How to fine-tune bert for text classification?. In *China national conference on Chinese computational linguistics* (pp. 194-206). Cham: Springer International Publishing.