Real-Time Phone Call Fraud Detection Using NLP and Interpretable Machine Learning

Dang Quang Nguyen
Applied Modeling and Quantitative Methods (Big Data Analytics)
Trent University
Peterborough, Canada
dangquangnguyen@trentu.ca

Abayomi Olaoye
Applied Modeling and Quantitative Methods (Big Data Analytics)
Trent University
Peterborough, Canada
abayomiolaoye@trentu.ca

Siyuan Zhao
Applied Modeling and Quantitative Methods (Big Data Analytics)
Trent University
Peterborough, Canada
barryzhao@trentu.ca

***Abstract*— This paper addresses the growing threat of real-time phone scams that exploit urgency, impersonation, and caller deception. It proposes an interpretable, real-time fraud detection system combining PySpark MLlib, Kafka streaming, and LLM-based explanation. The system includes: (1) a TF-IDF-driven PySpark pipeline using Decision Tree, Random Forest, and XGBoost classifiers; (2) a Kafka-based architecture for streaming transcribed dialogues; and (3) a DeepSeek module that generates natural language justifications for predictions. Trained on the BothBosu multi-agent dataset, the Decision Tree was selected for deployment due to its balance of accuracy (F1 = 0.9834) and transparency. While ensemble models achieved marginally higher metrics (AUC > 0.999), they risked overfitting. Results show the system can detect scam dialogues with high precision and deliver real-time, interpretable outputs through a Streamlit interface. This study demonstrates that lightweight, explainable ML pipelines can offer scalable, trustworthy solutions for mitigating phone-based fraud in dynamic environments.**

Keywords*— *Real-time scam detection, phone fraud detection, PySpark MLlib, Decision Tree classifier, TF-IDF feature extraction, Apache Kafka, natural language processing (NLP), explainable AI (XAI), DeepSeek, Streamlit interface

I. Introduction

With the rapid expansion of the information society, phone scams have become a pervasive threat targeting individuals, financial institutions, and businesses. Fraudsters increasingly exploit caller ID spoofing, urgency tactics, and social engineering to deceive victims in real time. According to the U.S. Federal Trade Commission, over 2.6 million fraud cases were reported in 2024, totaling $12.5 billion in losses, with nearly $3 billion attributed to imposter scams [1].

Despite active efforts from academia, industry, and government, existing fraud detection systems fall short—particularly in **real-time detection**. Traditional rule-based and post-call analysis tools often lack adaptability, generate high false positives, and struggle with unstructured conversational data. While Natural Language Processing (NLP) has shown promise in analyzing call content, many existing models are either opaque, too slow for live use, or lack sufficient generalization [2, 3].

To address this gap, we propose a real-time, NLP-based phone scam detection system designed for **interpretability and deployability**. Our key contributions include:

1. A scalable PySpark-based ML pipeline trained on a synthetic agent-customer dataset using TF-IDF and Decision Trees.
2. A Kafka-powered real-time inference system with a Streamlit frontend for live dialogue classification.
3. Integration with DeepSeek, a large language model (LLM), to generate **human-readable justifications** for each prediction.

This paper is structured as follows: Section 2 surveys existing approaches to phone fraud detection. Section 3 details the system design and implementation. Section 4 presents results and comparative analysis, and Section 5 concludes with findings and future directions.

II. Literature Review

This section examines recent advancements in real-time phone call fraud detection, focusing on machine learning (ML) and natural language processing (NLP) techniques. It traces the evolution from traditional rule-based systems to modern NLP-driven models, highlighting their strengths, limitations, and the research gaps that this study aims to address.

A. Traditional Approaches: Rule-Based and Behavior-Based Systems

Early fraud detection methods primarily relied on static rule-based systems that flagged anomalies based on predefined thresholds, such as unusual call durations or frequencies. For instance, [4] integrated rule-based logic with neural networks to monitor mobile traffic, achieving a

false positive rate below 3%, though at a significant computational cost. Similarly, [5] explored behavior-based intrusion detection methods that monitored user activity for deviations, achieving notable detection accuracies. However, these traditional approaches often suffered from high false alarm rates, limited scalability, and inadequate handling of real-time or unstructured data [7].

B. NLP-Based Approaches to Fraud Detection

The integration of NLP techniques has markedly enhanced real-time fraud detection by enabling the analysis of textual and speech content to uncover deceptive patterns. [8] introduced ScamDetector, a fine-tuned language model tailored for scam detection in Singaporean call centers, achieving an accuracy of 99.6% with fast inference times. However, its reliance on known scam phrases raised concerns about adaptability to evolving fraud tactics [9]. Similarly, [10] developed Call-E, an NLP-driven system for blocking fraudulent calls and SMS in real-time, achieving 98% accuracy while balancing security and user privacy. Nevertheless, its dependence on background permissions was identified as a potential trust barrier.

[11] proposed a telecom fraud detection model based on RoBERTa, enhanced with multi-head attention and dual loss functions. Using the CCL2023 dataset, the model achieved an F1 score of 98.10, demonstrating improved classification reliability over conventional models. Additionally, [12] compared various ML models—including k-NN, SVM, GNB, and RF—on a dataset of call transcripts processed via NLP. Both k-NN and SVM achieved 99% accuracy, outperforming RF (97%) and GNB (94%). However, the study's limited dataset size impacted the generalizability of these findings.

C. TF-IDF and Hybrid ML Techniques

Term Frequency-Inverse Document Frequency (TF-IDF) remains a widely used method for feature extraction in textual fraud detection tasks. [13] combined TF-IDF with SVM, Naive Bayes, and Decision Trees to classify texts as suspicious or benign, with the SVM model achieving 84.57% accuracy, validating the relevance of linear classifiers in text classification problems. [14] employed TF-IDF alongside Logistic Regression for sentiment analysis of LinkedIn reviews, achieving an accuracy of 91.86%, demonstrating the effectiveness of this combination in sentiment classification tasks. [15] utilized logistic regression and Random Forest to detect fraud using structured call metadata, achieving accuracies of 93.5% and 96.1%, respectively, demonstrating strong potential in real-time fraud detection with proper feature engineering and dimensionality reduction.

Table I summarizes the key results and limitations of previous studies:

TABLE I. SUMMARY OF PREVIOUS RESEARCH RESULTS AND LIMITATIONS

Ref.	Techniques	Application	Accuracy/F1 Score	Limitations
[4]	Rule-based & Neural Network	Mobile traffic monitoring	FP < 3%	High computational cost
[8]	Fine-tuned LLM (ScamDetector)	Fraudulent call detection	99.6%	Low inference time; limited adaptability
[10]	RNN (Call-E)	Fraudulent calls and SMS blocking	98%	Low inference time; trust and permission issues
[11]	RoBERTa & Multi-head Attention	Telecom fraud detection	F1 = 98.10	Low inference time; limited dataset size
[12]	k-NN, SVM, RF, GNB	NLP-based call transcript analysis	k-NN & SVM: 99%	Limited dataset size
[13]	TF-IDF & SVM	Suspicious text classification	84.57%	Moderate accuracy indicating limited complexity
[14]	TF-IDF & Logistic Regression	Sentiment analysis	91.86%	Dependence on feature quality
[15]	Logistic Regression & RandomForest	Fraud detection from metadata	Random Forest: 96.1%	Requires structured and detailed metadata

D. Identified Gaps and Implications

While the reviewed works demonstrate the growing efficacy of ML and NLP techniques in real-time fraud detection, several critical gaps persist:

- Adaptability to Evolving Scams: Many NLP models rely heavily on known scam phrases, limiting their effectiveness against novel fraud tactics [9].
- Scalability Challenges: Traditional systems often struggle with high computational costs and poor scalability, hindering their deployment in large-scale, real-time environments [7].

Addressing these gaps, this study proposes a scalable, interpretable real-time detection system that integrates TF-IDF for transparent feature extraction, a Decision Tree classifier for interpretability, and a Large Language Model

(LLM) explanation module to enhance user trust and usability.

The following section outlines the methodology employed in developing this proposed system.

III. Methodology

This section outlines the systematic approach used to develop a real-time scam detection system designed to identify fraudulent conversations from transcribed dialogues. The methodology is structured to ensure scalability, interpretability, and real-time responsiveness—critical factors for practical fraud mitigation systems. It encompasses data acquisition, preprocessing, feature engineering, model development, evaluation, and system architecture. The implementation leverages Apache Spark MLlib for scalable machine learning tasks and integrates a Kafka-driven pipeline with an LLM-based explanation module for real-time interpretability.

A. Data Acquisition and Description

The dataset used in this study is "agent_conversation_all.csv," published by BothBosu [16], comprising 1,600 synthetic agent-customer dialogues annotated as scam (1) or non-scam (0). Each record comprises the full conversation text and an associated binary label. The dataset is synthetically generated yet linguistically rich, representing key fraud indicators such as urgency, impersonation, and social engineering.

To ensure balanced learning, the dataset is evenly split between scam and non-scam classes (800 each). Data diversity includes conversations with varied agent personas and communication tones, which enhances model generalization.

B. Programming Environment and Libraries

The implementation was conducted in Python 3.10 using Apache Spark MLlib 3.5.5, chosen for its scalability and native support for distributed data pipelines [17]. Supplementary libraries include **Pandas**, **NumPy**, **Matplotlib**, and **Seaborn** for data manipulation and visualization. For real-time data streaming, **Apache Kafka** was employed alongside the **confluent_kafka** client library (v2.9.0) to enable efficient message handling. The **Streamlit** library (v1.44.1) was used to build the frontend interface for user interaction. Finally, **DeepSeek**, a large language model, was integrated to generate natural language explanations for classification outcomes, enhancing model transparency and interpretability.

C. Data Preprocessing

Preprocessing included text normalization to eliminate high-frequency, low-information tokens (e.g., punctuation, numbers, and common stopwords). All entries were lowercased and cleaned using regular expressions. Spark's Tokenizer and StopWordsRemover were employed to tokenize and filter the text. Dialogues were also manually reviewed to verify the integrity of the labels and ensure no leakage. These steps reduced noise and emphasized semantically meaningful content that could influence classification decisions.

D. Feature Engineering

After preprocessing, the textual data was transformed into numerical representations using a two-stage vectorization pipeline. First, CountVectorizer was applied to convert tokenized dialogues into sparse term frequency vectors. Then, TF-IDF (Term Frequency–Inverse Document Frequency) weighting was used to scale these vectors, emphasizing rare but informative terms while downplaying common ones [18]. This technique was selected for its interpretability and strong alignment with transparent models like Decision Trees. To ensure computational efficiency, the vocabulary size was limited to the top 10,000 tokens. This approach effectively captured class-distinctive linguistic patterns crucial for accurate and explainable fraud detection.

E. Experimental Setup

Data was split using stratified sampling into training (70%), validation (10%), and test (20%) sets to maintain class balance. A fixed random seed (42) ensured reproducibility. Each classifier was encapsulated in a Spark ML pipeline, combining preprocessing, vectorization, transformation, and model stages. This modular approach ensured consistent data flow and allowed seamless switching between models. Hyperparameter tuning (e.g., depth of tree) was guided by literature and validation set performance [19].

F. Model Development

Multiple classifiers were explored, including Logistic Regression, Decision Trees, Random Forest, and XGBoost. The pipeline included tokenization, stopword removal, TF-IDF feature extraction, and classification. Random Forest

and XGBoost models, also trained on the same pipeline. Three classifiers were trained and evaluated:

1. **Decision Tree** (primary model)
2. **Random Forest** (benchmark ensemble model)
3. **XGBoost** (gradient-boosted benchmark model)

TABLE II. MODEL COMPARISON

Model	Accuracy	F1 Score	AUC	Inference Time	Interpretability
Decision Tree	0.9834	0.9834	0.9894	Fast	High
Random Forest	0.9934	0.9934	0.9998	Medium	Medium
XGBoost	0.9934	0.9934	0.9999	Slow	Low

While ensemble models slightly outperformed the Decision Tree in raw metrics, they incurred higher latency and reduced interpretability (**Table II)**. The Decision Tree, with a maximum depth of 5 and Gini impurity as the criterion, was selected for deployment due to its ability to provide transparent decision boundaries and fast predictions suitable for real-time systems [20].

The Decision Tree classifier was selected for its strong balance between interpretability and performance [21]. Given the synthetic nature of the dataset and near-perfect ensemble results, academic supervision recommended deploying the Decision Tree to validate behavior and reduce the risk of overfitting or hidden data leakage.

G. Evaluation Metrics

The models were evaluated using the following metrics:

- **Accuracy**: Overall correctness of classification
- **Precision**: Proportion of true positives among predicted positives
- **Recall**: Proportion of true positives among actual positives
- **F1 Score**: Harmonic mean of precision and recall
- **AUC (Area Under Curve)**: Sensitivity across thresholds

All metrics were calculated as weighted averages to account for future use cases where class imbalance may exist. Although the dataset is balanced, weighted metrics were used to prepare for deployment scenarios where imbalance may occur [22].

H. Real-Time Interface and System Architecture

The end-to-end real-time fraud detection system integrates Apache Kafka, Spark MLlib, DeepSeek, and a Streamlit-based frontend. Apache Kafka handles the ingestion of transcribed dialogues from call data and streams them into Spark for real-time processing. Within Spark, the trained Decision Tree classifier executes preprocessing and prediction tasks using the TF-IDF vectorized pipeline.

The frontend, developed in Streamlit, allows users to input individual or batch dialogues. Upon submission, each input is classified in real time, with results returned in sub-second latency. This includes the scam/non-scam prediction, model confidence score, and a human-readable explanation. This interactive interface enhances usability for fraud investigators and call center analysts.

I. Justification for DeepSeek Integration

DeepSeek was selected over general-purpose large language models (LLMs) such as GPT-3.5 and LLaMA due to its superior performance across several critical dimensions. First, its efficiency enables faster inference speeds, making it well-suited for real-time applications where low latency is essential. Second, DeepSeek demonstrates exceptional precision, delivering concise and well-structured responses that closely adhere to user prompts and follow instructions accurately. Third, it exhibits high reliability by minimizing hallucinations and generating factually grounded outputs, a crucial feature for high-stakes decision-making.

In practical applications, DeepSeek excels at identifying scam-indicative phrases—such as “Your Social Security number has been suspended”—and provides actionable recommendations, including “Terminate the call” or “Verify the request with the bank.” This capability not only enhances real-time scam detection but also ensures explainability and decision support, allowing users to understand and act upon potential threats with confidence. Such performance underscores DeepSeek’s suitability for security-sensitive environments where accuracy, speed, and trustworthiness are paramount [23].

IV. RESULTS AND ANALYSIS

This section presents the empirical results of the proposed scam detection system, highlighting its performance across multiple datasets and comparing different classifiers. The analysis includes classification metrics, confusion matrices, feature importance insights, and interpretability considerations in both static and real-time deployment settings.

A. Model Evaluation Across Data Splits

The Decision Tree classifier was assessed across training, validation, and test sets using accuracy, F1-score, and AUC. On the training set, the model achieved perfect performance (accuracy = 1.0000, F1 = 1.0000), confirming its ability to learn training data patterns. However, generalization was verified using validation and test sets to mitigate overfitting concerns.

TABLE III. DECISION TREE PERFORMANCE ACROSS TRAINING, VALIDATION, AND TEST SPLIT

Dataset	Accuracy	Weighted Precision	Weighted Recall	F1 Score	AUC
Training	1.0000	1.0000	1.0000	1.0000	1.0000
Validation	0.9809	0.9810	0.9809	0.9809	0.9932
Test	0.9834	0.9835	0.9834	0.9834	0.9894

These results from **Table III** demonstrate the Decision Tree's high accuracy and generalization ability, with minimal overfitting. In contrast, ensemble models like XGBoost and Random Forest achieved slightly higher scores (AUC = 0.9999 and 0.9998, respectively). However, these gains were offset by longer inference times and lower transparency. **Table III** presents a summary comparison across all models.

B. Comparative Analysis with Previous Research

The proposed system demonstrates competitive performance while maintaining a strong focus on interpretability—an essential feature often lacking in prior work. For instance, [8] employed ScamDetector, a fine-tuned language model specifically designed for detecting fraudulent calls in Singaporean call centers, achieving an accuracy of 99.6% but facing limitations such as low inference speed and limited adaptability. Similarly, [11] proposed a RoBERTa-based telecom fraud detection model enhanced with multi-head attention and dual loss functions, achieving an F1 score of 98.10, yet also experiencing low inference speeds and dataset limitations (more in Table I).

While these studies demonstrate strong predictive capabilities, they typically sacrifice model transparency. Transformer-based and ensemble methods often function as black boxes, presenting significant interpretability challenges in high-stakes, real-time environments where decisions require clear justification. In contrast, our proposed Decision Tree-based approach prioritizes interpretability without significantly compromising performance, thereby offering a more practical solution suitable for real-world deployment.

C. Confusion Matrix Interpretation

The confusion matrices (Appendix A) illustrate the Decision Tree classifier's predictive performance across both the validation and test datasets. On the validation set, the model achieved an accuracy of 98.09%, correctly identifying 80 scam cases while misclassifying only 3 non-scams. On the test set, it maintained strong generalization with an accuracy of 98.34%, correctly classifying 135 non-scams and 162 scams, with just 5 misclassifications overall. The model shows a slight conservative bias, favoring the detection of scams, which is appropriate for fraud detection scenarios where false negatives are more critical than false positives (more details in Appendix A).

D. Feature Importance and Lexical Insights

To enhance model transparency, key lexical features influencing the Decision Tree classifier were analyzed. As shown in Fig. 1, the left panel presents word frequency distributions, while the right plots scam ratio against feature importance.

Words such as "**process", "social", "device**", and "**prize**" were highly indicative of scam dialogues. Notably, "**process**" appeared 693 times in scam cases and had the highest feature importance (0.616), making it the most decisive term. Though "device" and "prize" exhibited near-perfect scam ratios, their influence was moderated by lower frequency.

Conversely, terms like "**insurance", "pm**", and "**scheduled**" were more common in non-scam dialogues and acted as strong negative indicators. These insights, expanded in **Appendix B**, validate the model's focus on clear, class-distinctive cues—supporting its interpretability and practical deployment.

Appendix C extends this analysis to the Random Forest model. While it shares overlap with the Decision Tree—highlighting "**process", "social",** and **"security**"—it also emphasizes contextually ambiguous terms like "legitimate" and "assure", often used to build false credibility. This reflects Random Forest's capacity to capture subtle patterns, albeit with reduced interpretability.

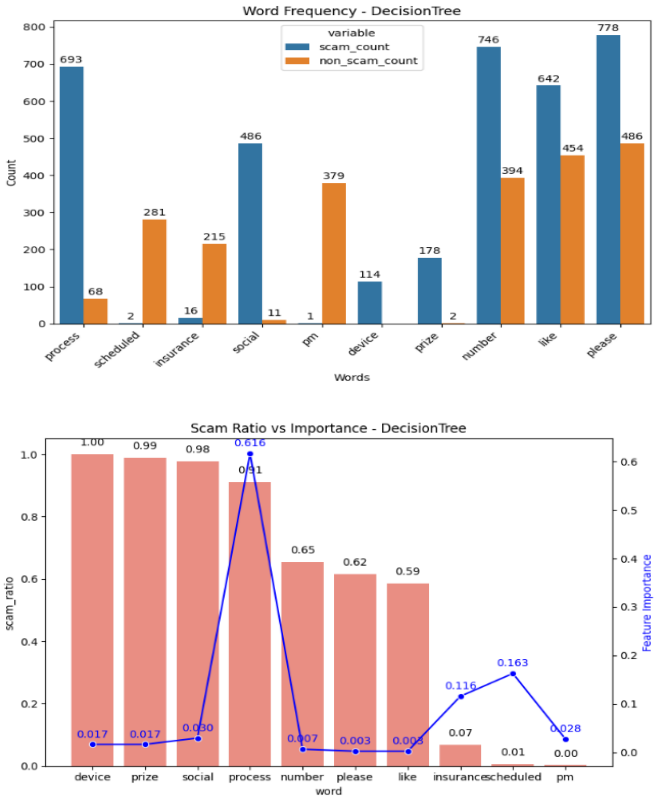

Fig. 1. Word Frequencies and Feature Importances in Scam Classification

Example of a figure caption.

E. Visual Interpretation of Decision Pathways

To promote transparency, the trained Decision Tree structure was visualized (**Appendix D**). Each decision node corresponds to a threshold condition on a specific lexical feature (e.g., "activity", "device"), enabling clear traceability of predictions. This reinforces the model's interpretability for deployment in high-stakes scenarios.

F. Real-Time Interface and LLM Explanation

As part of the real-time deployment, a Streamlit-powered frontend was deployed to classify incoming dialogue streams in real time. Upon classification, results and confidence scores are presented alongside natural language explanations generated by DeepSeek. The explanation panel, provides:

- Summaries of decision rationale (**Appendix E**)
- Highlighted scam indicators (e.g., "Your Social Security number has been suspended") (**Appendix E**)
- Suggested user actions (e.g., "Report to authorities") (**Appendix F**)

This integration promotes interpretability and trust in real-world applications.

V. Conclusion

This study presented the design, development, and evaluation of a real-time scam detection system capable of classifying agent-customer dialogues as scam or non-scam. Using the BothBosu multi-agent dialogue dataset, the system was trained using multiple classifiers within an Apache Spark MLlib pipeline, featuring robust preprocessing and interpretable TF-IDF-based feature engineering.

The Decision Tree classifier, selected for deployment based on interpretability and efficiency, achieved strong results across all data splits (test set F1-score = 0.9834, AUC = 0.9894), while ensemble models like XGBoost and Random Forest showed slightly higher metrics. However, those models raised concerns regarding overfitting due to perfect validation scores. On recommendation from academic supervision, the Decision Tree was chosen to verify model behavior under transparent, auditable conditions.

Key features contributing to accurate scam detection included terms such as "process," "social," and "device"—words frequently tied to urgency and impersonation tactics used in fraudulent schemes. The system also incorporated a Kafka-based streaming architecture and a DeepSeek-powered explanation module to support real-time analysis and user-facing interpretability. The frontend displayed classification results, confidence scores, and natural language justifications, promoting actionable decision-making in deployment settings.

Despite these strengths, limitations remain. The dataset was synthetic and domain-specific, which may constrain generalization to real-world or multilingual data. Additionally, while Decision Trees offer transparency, they may not capture the full complexity of nuanced language patterns compared to more advanced models.

Future work should focus on validating the model in production-like environments with noisy, multi-language ASR transcripts and expanding the explanation system to include token-level justifications. Integration of user feedback and adaptive learning mechanisms could further enhance long-term system performance and trust.

In summary, this research demonstrates that a hybrid approach combining traditional machine learning, real-time data streaming, and modern explainability tools offers a viable, scalable solution for conversational fraud detection.

VI. References

[1] Federal Trade Commission. "Consumer Sentinel Network Data Book," Federal Trade Commission. [Accessed: March. 24, 2025]. [Online]. Available: https://public.tableau.com/app/profile/federal.trade.commission/viz/ConsumerSentinel/Infographic

[2] A. Abdallah, M. A. Maarof, and A. Zainal, "Fraud detection system: A survey," Journal of Network and Computer Applications, vol. 68, pp. 90-113, June. 2016, https://doi.org/10.1016/j.jnca.2016.04.007

[3] A. Gandhar, K. Gupta, A. K. Pandey, and D. Raj, "Fraud detection using machine learning and deep learning," SN Computer Science, vol. 5, no. 5, pp. 453, April. 2024. https://doi.org/10.1007/s42979-024-02772-x

[4] O.F. Nonyelum, "Fraud Detection in Mobile Communications Using Rule-Based and Neural Network System," IUP Journal of Science & Technology, Vol. 6, no. 4, pp.35, December. 2010

[5] A. Boukerche and M. S. M. A. Notare, "Behavior-based intrusion detection in mobile phone systems," Journal of Parallel and Distributed Computing, vol. 62, no. 9, pp. 1476-1490, September. 2002, https://doi.org/10.1006/jpdc.2002.1857

[6] J. A. Iglesias, A. Ledezma, A. Sanchis, and P. Angelov, "Real-time recognition of calling pattern and behaviour of mobile phone users through anomaly detection and dynamically-evolving clustering," Applied Sciences, vol. 7, no. 8, p. 798, August. 2017, https://doi.org/10.3390/app7080798

[7] B. M. Muter and A. J. Mohammed, "The future of AI: Assessing the strengths and limitations of deep learning and machine learning," Wisdom Journal For Studies & Research, vol. 4, no. 6, pp. 348-374, November. 2024, https://doi.org/10.55165/wjfsar.v4i06.407

[8] P. Y. J. Nicholas and P. C. Ng, "ScamDetector: Leveraging fine-tuned language models for improved fraudulent call detection," in TENCON 2024-2024 IEEE Region 10 Conference (TENCON), 2024, pp. 422-425, https://doi.org/10.1109/TENCON61640.2024.10902894

[9] O. Sandor and M. K. Malinka, "Resilience of biometric authentication of voice assistants against deepfakes," B.S. thesis, Brno University of Technology, 2023.

[10] A. Gupta, "Detection of spam and fraudulent calls using natural language processing model," in 2024 Sixth International Conference on Computational Intelligence and Communication Technologies (CCICT), 2024, pp. 423-427, https://doi.org/10.1109/CCICT62777.2024.00075

[11] J. Li, C. Zhang, and L. Jiang, "Innovative telecom fraud detection: A new dataset and an advanced model with RoBERTa and dual loss functions," Applied Sciences, vol. 14, no. 24, p. 11628, December. 2024, https://doi.org/10.3390/app142411628

[12] P. K. Kumar, S. Ray, L. Kumarasankaralingam, A. Ramamoorthy, P. Kumar, and A. Dutta, "Detecting fraud calls vis-à-vis natural language processing," 2024 2nd International Conference on Advancement in Computation & Computer Technologies (InCACCT), pp. 457-462, May. 2024, https://doi.org/10.1109/InCACCT61598.2024.10551082

[13] O. Sharif, M. M. Hoque, A. S. M. Kayes, R. Nowrozy, and I. H. Sarker, "Detecting suspicious texts using machine learning techniques," Applied Sciences, vol. 10, no. 18, p. 6527, September. 2020, https://doi.org/10.3390/app10186527

[14] N. S. Wardana, F. P. Aditiawan, and A. P. Sari, "Penggunaan Ekstraksi Fitur Tf-Idf dan Fasttext pada Klasifikasi Sentimen Ulasan Linkedin Dengan Metode Logistic Regression," *VISA: Journal of Visions and Ideas*, vol. 4, no. 3, pp. 1359–1371, 2024, https://doi.org/10.47467/visa.v4i3.2835.

[15] S. Apostu, "Using machine learning algorithms to detect frauds in telephone networks, " The Annals of "Dunarea de Jos" University of Galati. Fascicle III, Electrotechnics, Electronics, Automatic Control, Informatics*, vol. 43, no. 3, pp. 16-20, January. 2020, https://doi.org/10.35219/eeaci.2020.3.03

[16] BothBosu, "Synthetic Multi-Turn Scam and Non-Scam Phone Dialogue Dataset with Agentic Personalities," Hugging Face, 2024. [Online]. Available: https://huggingface.co/datasets/BothBosu/multi-agent-scam-conversation

[17] Apache Spark Documentation, "DecisionTreeClassifier," [Online]. Available: https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.classification.DecisionTreeClassifier.html

[18] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," Information Processing & Management, vol. 24, no. 5, pp. 513–523, 1988. https://doi.org/10.1016/0306-4573(88)90021-0

[19] H. He and E. A. Garcia, "Learning from imbalanced data," IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 9, pp. 1263–1284, 2009. https://doi.org/10.1109/TKDE.2008.239

[20] Apache Spark Documentation, "ML Decision Trees," [Online]. Available: https://spark.apache.org/docs/latest/ml-classification-regression.html

[21] J. R. Quinlan, "Induction of decision trees," Machine Learning, vol. 1, no. 1, pp. 81–106, 1986. https://doi.org/10.1007/BF00116251

[22] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," Information Processing & Management, vol. 45, no. 4, pp. 427–437, 2009. https://doi.org/10.1016/j.ipm.2009.03.002

[23] DeepSeek Documentation, "DeepSeek Large Language Model," [Online]. Available: https://deepseek.com (if a specific citation is needed, update accordingly)

VII. Appendix

A. Confusion Matrices – Decision Tree

TABLE IV. Train Set confusion matrix

Actual / Predicted	**0.0**	**1.0**
0.0	586	0
1.0	0	555

TABLE V. Validation Set confusion matrix

Actual / Predicted	**0.0**	**1.0**
0.0	74	1
1.0	2	80

TABLE VI. Test Set Confusion Matrix

Actual / Predicted	**0.0**	**1.0**
0.0	135	2
1.0	3	162

B. Top Predictive Features and Class Associations – Decision Tree

TABLE VII. Decision Tree – Top Words and Associations

Word	**Scam Count**	**Non-Scam Count**	**Scam Ratio**	**Importance**
process	693	68	0.911	0.616
scheduled	2	281	0.007	0.163
insurance	16	215	0.069	0.116
social	486	11	0.978	0.030
pm	1	379	0.003	0.028
prize	178	2	0.989	0.017
device	114	0	1.000	0.017
number	746	394	0.654	0.007
like	642	454	0.586	0.003
please	778	486	0.616	0.003

C. Top Predictive Features and Class Associations Random Forest

TABLE VIII. Random Forest – Top Words and Associations

Word	**Scam Count**	**Non-Scam Count**	**Scam Ratio**	**Importance**
legitimate	524	39	0.931	0.044
process	693	68	0.911	0.041
security	630	55	0.920	0.036
social	486	11	0.978	0.030
verify	685	115	0.856	0.030
identity	543	60	0.900	0.025
information	732	304	0.707	0.023
assure	615	99	0.861	0.021

detected	261	0	1.000	0.016
onetime	451	21	0.956	0.015

D. Decision Tree Visualization

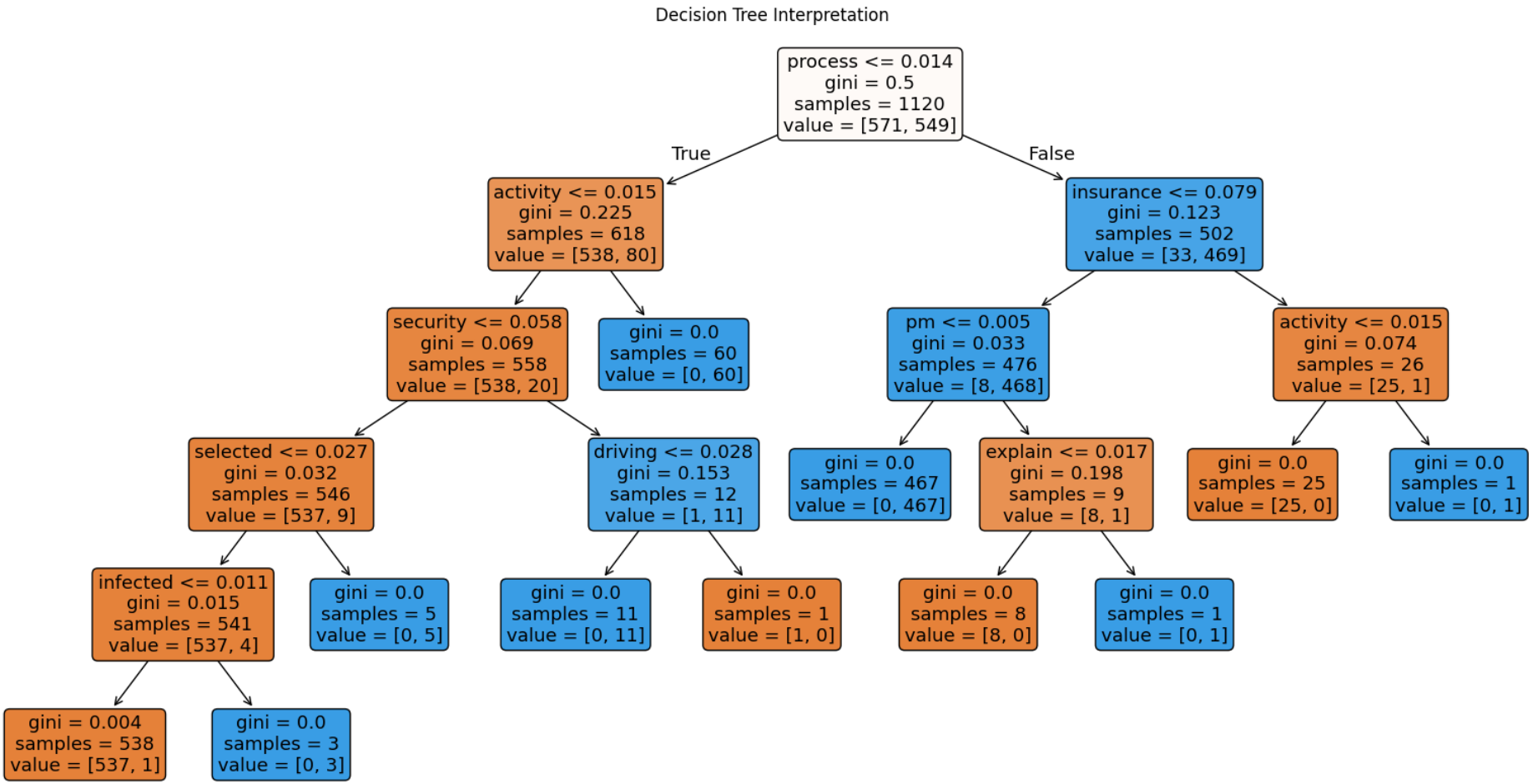

Fig. 2. Decision Tree Structure.

E. User Interface for Single Dialogue Prediction

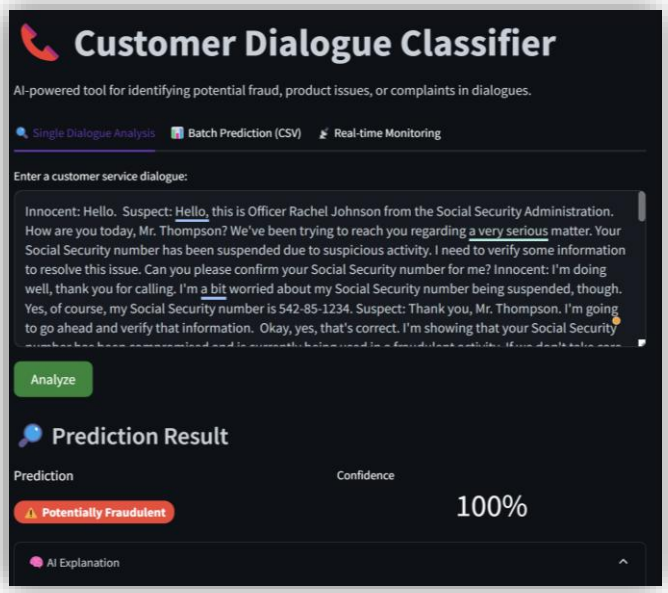

Fig. 3. Single Dialogue Prediction.

F. Explanation & Recommended Actions Panel

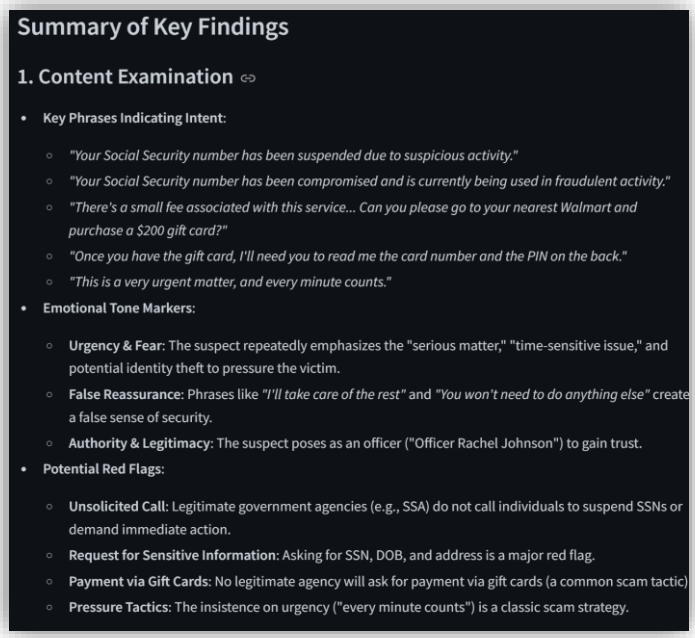

Fig. 4. AI Explanation Panel.

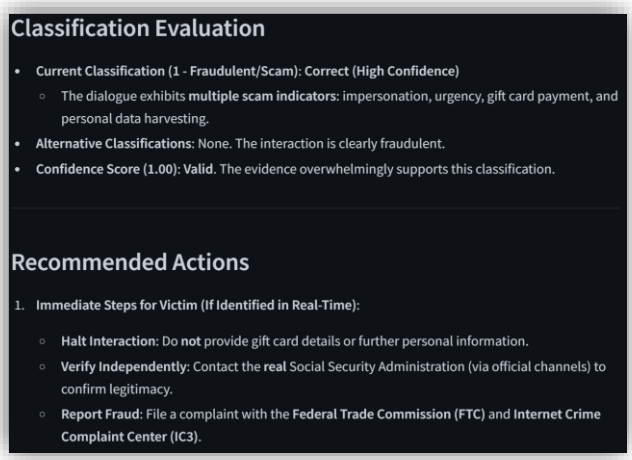

Fig. 5. AI Recommended Actions Panel.