

Automated Risk Insight System (ARIS)

README

Project Overview

The Automated Risk Insight System (ARIS) is an AI-powered system designed to analyze IT incidents, predict risk levels, and estimate resolution times using machine learning models. It also includes an interactive chatbot that allows employees to query incident data and receive intelligent recommendations while maintaining security rules for data access.

This application is built with FastAPI, Scikit-learn, XGBoost, and Ollama LLM for natural language processing. Google Gemini Pro and OpenAI ChatGPT 4o, o1 and o3, helped with debugging, making and optimizing the models.

Purpose & Key Features

ARIS provides automated assistance for IT risk management and operational decision-making. The system:

- ✓ **Predicts risk categories** (`HIGH_RISK` / `LOW_RISK`) based on incident attributes.
- ✓ **Estimates incident resolution times** using a machine learning regression model.
- ✓ **Allows employees to interact with incident data** via a chatbot that enforces access control rules.
- ✓ **Provides references to the original data source** for employees to inspect the raw information.
- ✓ **Works without requiring manual input of resolution time** – it first predicts resolution time before using it in risk classification.



How It Works

① **Input:** The application takes incident reports in JSON format, excluding `time_to_resolution`.

② **Step 1 - Predict Resolution Time:**

- Uses the `train_model_new_res.py` regression model to predict resolution time.

③ **Step 2 - Predict Risk Level:**

- Uses the `train_model_orig_risk.py` classification model, now including the predicted `time_to_resolution`.

④ **Step 3 - LLM Chat Interaction:**

- Employees can ask questions via the chatbot to get AI-powered insights.
- Chatbot enforces security rules based on the employee's credentials.
- Employees can get a reference to the original data source.



Data Preprocessing & Feature Engineering

Source: Kaggle IT Incident Log dataset

Link: <https://www.kaggle.com/datasets/shamiulislamshifat/it-incident-log-dataset>

① **Data Cleaning:**

- Dropped unnecessary columns (`number`, `sys_created_by`, `sys_updated_by`, etc.)
- Converted categorical features to numerical using Label Encoding
- Saved encoders for consistency in predictions

② **Feature Engineering:**

- Time to Resolution Calculation: Derived from `opened_at` and `resolved_at` timestamps
- Incident Complexity Score: Based on `reassignment_count`, `reopen_count`, and `priority`
- Escalation Flag: Assigned if `reassignment_count > 2`
- SLA Breach Indicator: Marked incidents with `time_to_resolution > 24 hours`

③ **Final Processed Dataset Saved:** `engineered_incident_data.csv`

Technical Stack

- **Backend:** FastAPI
- **Machine Learning:** Scikit-learn, XGBoost, SMOTE
- **Data Processing:** Pandas, NumPy, Pickle
- **LLM Chatbot:** Ollama (Llama 3.1 8B model)
- **UI & API Requests:** Streamlit, Requests
- **Database:** JSON file-based storage (can be extended to SQL/NoSQL databases)
- **Knowledge base idea:** Tested on the side in Open WebUI with Ollama server

Challenges & Blockages Encountered

1 Feature Mismatch in Models

- The initial design of the models required `time_to_resolution`, but the JSON input lacked this field.
- Solution: The system was redesigned to first predict `time_to_resolution` before using it in risk classification.

2 Scaling & Data Transformation Issues

- Incorrect transformations led to extremely high or negative resolution times.
- Solution: Switched to MinMaxScaler for improved numerical stability.

3 LLM API Connectivity Issues

- The chatbot initially failed to respond due to missing or incorrect model references.
- Solution: Ensured Ollama LLM was running, verified API endpoints, and fixed formatting issues in responses.

4 Limited Time for Development & Testing

- Due to time constraints, further optimizations (e.g., fine-tuning models, database integration) were not implemented.

Future Development & Usage

ARIS has significant potential for expansion and adaptation to different use cases:

♦ **Enhanced Security & Role-Based Access**

- Improve the chatbot to enforce access rules dynamically based on roles.
- Implement OAuth authentication for employee login.

♦ **Integration with External Data Sources**

- Connect to SQL databases or enterprise ticketing systems (e.g., ServiceNow, Jira).
- Allow direct queries into structured databases instead of JSON storage.

♦ **Multi-Language LLM Capabilities**

- Expand chatbot beyond English, supporting multilingual environments.
- Train the LLM to handle specialized IT incident-related knowledge.

♦ **Automated Incident Escalation & Workflows**

- Trigger automatic escalations based on predicted HIGH_RISK incidents.
- Integrate with email alerts, Slack, or Microsoft Teams notifications.

Conclusion

Despite encountering various challenges, the Automated Risk Insight System (ARIS) successfully demonstrates the potential of AI-powered risk analysis and decision-making in IT incident management. The system functions correctly within its current scope, predicting resolution times and risk levels, and providing a chatbot for intelligent data interactions.

With further development, ARIS can become a fully operational AI-driven IT assistant, improving efficiency, reducing manual workload, and ensuring better risk handling in enterprise environments.



Installation & Running the Project

First install Ollama server and download Llama3.1:8b (GPU > 2GB required) see the websites documentation for more information.

<https://ollama.com/>

<https://ollama.com/library/llama3.1>

1 Setup Environment

```
python -m venv venv
source venv/bin/activate    # On Windows use: venv\Scripts\activate
pip install -r requirements.txt
```

2 Preprocess the data

```
python preprocess.py
```

3 Run FastAPI Backend

```
uvicorn api:app --reload
```


4 Run Main Script - CLI (LLM & Prediction Pipeline)

```
python main.py
```

5 Run Streamlit UI (Optional for web app)

```
streamlit run app.py
```

Author: Alexandru Ciobanu

 **Last Updated:** February 2025