Hackathon Problem Statement:2

AI Powered Churn Reasoning and Retention Insights for Auto Insurance

Context/Background: Customer churn is a major challenge for auto insurance companies, leading to significant revenue loss and high acquisition costs. By leveraging historical data such as demographics, income, location, policy tenure, and credit behaviour, machine learning models can identify customers likely to discontinue their policies. Explainable AI (XAI) provides the reasoning behind model outputs, enabling targeted, data-driven retention strategies.

The Challenge:

Develop a model that analyses historical customer, demographic, and policy data to predict the likelihood of churn, using explainability techniques such as SHAP, LIME, or Grad-CAM to highlight which factors most influenced each prediction. Design an interactive dashboard showing customer churn probabilities, reasons for churn, and regional trends that allows non-technical users to explore predictions and insights.

Understanding the Problem and Scope

Use Cases: Participants are expected to build a prototype that can:

- Analyses historical customer, demographic, and policy data to predict the likelihood of churn.
- Output a churn probability score for each customer, helping identify high-risk individuals before they leave.
- Develop an interactive dashboard showing customer churn probabilities, reasons for churn, and regional trends that allows non-technical users to explore predictions and insights.

Recommended Tools/Technologies but not limited to

- Explainability techniques(e.g. SHAP, LIME, or Grad-CAM)
- Interactive web app (e.g., Streamlit, Gradio, Flask, Angular)
- Python ML Libraries

Guidelines

Dataset: Use publicly available datasets such as:

- No internal company data will be shared.
- Kaggle: Auto Insurance churn analysis dataset
- Participants may use publicly available pretrained models or libraries for explainability and visualization.
- Think about Data Imbalance that will introduce the bias.

Technical and Implementation Details

- Train an ML model to compute a churn risk score for each customer.
- Ensure the solution is broken into distinct components (e.g., prediction, explainability, UI).
- Add simple explainability (e.g., feature importance, SHAP, or reason codes).
- The solution should handle large datasets efficiently.
- Ensure model inference runs within seconds.

Explainability:

- Encourage the use of SHAP/LIME to explain churn predictions and show which Features (e.g., Premium to Income ratio, Credit score, Policy Tenure, Area, claim history) most influenced each decision.

Guidelines contd…

Output	A working prototype (web app, API, or notebook demo).
	A model should output a churn probability score for each customer, helping identify high-risk individuals before they leave.
	An interactive dashboard showing customer churn probabilities, reasons for churn, and regional trends that allows non-technical users to explore predictions and insights.
Edge Cases to Test	Handling noisy, incomplete, or imbalanced data.
	Addressing edge cases, such as customers with very short or very long policy tenures.
	Ensure explanation exists even when model abstains or has low confidence. It should work consistently across diverse customer profiles.

Expectations and Deliverables

Format	• Both a working prototype (minimum: web app or notebook demo) + a short presentation (pitch deck).
Prototype Deliverable	• Web app (e.g., Streamlit/Gradio/Flask) • API endpoint or notebook for backend logic
Minimum Deliverable	• Standalone Jupyter Notebook with inference pipeline using public or synthetic data
Turnaround Expectation	• The system should return results within seconds per customer record
Accuracy Expectation	• Use Confusion Matrix, Precision, Recall scores, ROC-AUC etc.

Evaluation Criteria

Key criteria:

- Innovation & Creativity – how novel is the approach?
- Technical Implementation – how well is AI/ML applied (model choice, training, reasoning)?
- Accuracy & Reliability – correctness of churn prediction results, ability to avoid overfitting or bias and handling of edge cases.
- Usability & Explainability – clear UI, ability to explain predictions (not just “black box”).
- Scalability & Realism – can the solution scale, and would it make sense in real-world claims?

Judging Rubric:

- Innovation and Explainability (30%)
- Technical Approach (25%)
- Accuracy and Robustness (20%)
- User Experience (15%)
- Presentation and Impact(10%)