Health Insurance Lead Prediction

**Q1.A brief on the approach, which you have used to solve the problem?**

**Answer:** The problem was tackled using a structured pipeline, ensuring a systematic approach to data handling, model training, and evaluation. The workflow included:

* **Data Loading & Inspection:** The dataset was loaded using pandas, followed by an initial examination (head (), info) to understand the data structure and check for missing values.
* **Handling Missing Data:** Numerical missing values were imputed with the median (robust to outliers), while categorical missing values were replaced with the mode (most frequent category). This ensured consistency between train and test sets.
* **Feature Encoding & Scaling:** Label encoding was applied to categorical variables, converting them into numerical form. Standardization (Standard Scalar()) was used to normalize numerical features, improving model convergence.
* **Data Splitting:** The dataset was split into training and validation sets (80-20 split, stratified by the target variable) to prevent data leakage and ensure balanced class distribution.
* **Model Training & Evaluation:** Five models were trained—Random Forest,

Gradient Boosting, XG Boost, Logistic Regression, and SVM. Probabilistic outputs were used for comparison, and the best model was selected based on ROC-AUC score (a robust metric for imbalanced classification).

* **Feature Importance Analysis:** Importance scores were extracted from tree-based models and coefficients from linear models to understand which features

influenced predictions.

* **Performance Comparison & Selection:** ROC curves, AUC scores, and classification reports were used to select the best-performing model. The final model was used to generate predictions on the test set.

**Q2. What data-preprocessing / feature engineering ideas really worked? How did you discover them?**

**Answer:** The following steps were included as follows

* **Handling Missing Data Thoughtfully:** Instead of dropping missing values, numerical features were imputed with the median and categorical features with the mode. This prevented loss of data and retained consistency.
* **Encoding Categorical Variables:** Label encoding worked well for categorical features like "Accomodation\_Type" and "Reco\_Insurance\_Type", as they had a small number of unique values.
* **Standardization of Numerical Features:** Models like Logistic Regression and SVM benefited from Standard Scaler, ensuring better model performance and faster convergence.
* **Feature Selection via Importance Scores:** Visualizing feature importance helped identify the most relevant variables, leading to better interpretability and potential feature selection in future iterations.
* **Stratified Train-Test Split:** Using stratify=y ensured that class distributions remained balanced across training and validation sets, improving generalization.

# How were these discovered?

 Handling Missing Values: Identified through msno. matrix(train\_data) addressed by filling with a median and mode.

 Encoding Categorical Features: Identified by inspecting train\_data.info () addressed using Label Encoder

# Q3. What does your final model look like? How did you reach it?

**Answer**: The following steps were included as follows

* **Multiple Models Trained:** The pipeline trained Random Forest, Gradient Boosting, XG Boost, Logistic Regression, and SVM. Their performance was assessed using ROC-AUC scores, classification reports, and feature importance plots.
* **Model Selection via AUC Score:** The model with the highest ROC-AUC score was chosen. The winning model was either XG Boost, Gradient Boosting, or Random Forest (since tree-based models typically perform well on structured data).
* **Final Predictions on Test Data:** The best model was used to generate

probability-based predictions on the test dataset, which were converted to binary outputs using a 0.5 threshold. The final results were saved as CSV files and packaged into a ZIP file for submission.

**How it was reached**: The final model was reached through empirical evaluation. The code trains and evaluates five different models and selects the one with the best performance (highest ROC AUC score) on the validation dataset.