Technical Report: AI-Powered Insurance Risk Assessment & Customer Insights System

---

1. Executive Summary

Overview of the Problem

The insurance industry faces numerous challenges, including predicting risk scores, estimating claim amounts, detecting fraudulent claims, analyzing customer sentiments, and making insurance policies more accessible. Manual processes are often slow, error-prone, and lack scalability.

Proposed AI Solution

The **AI-Powered Insurance Risk Assessment & Customer Insights System** leverages machine learning and natural language processing to automate and enhance key insurance processes:

- **Risk Score Prediction**: Assess customer and policy data to calculate risk levels.
- **Claim Amount Prediction**: Predict the potential amount of claims.
- **Fraud Detection**: Identify fraudulent claims through anomaly detection algorithms.
- **Sentiment Analysis**: Analyze customer feedback to derive insights.
- **Policy Translation & Summarization**: Translate insurance policies into multiple languages and provide concise summaries.
- **FAQ Chatbot**: Provide instant responses to common insurance-related queries.

The solution uses advanced models for accurate predictions and provides clear visual insights to aid decision-making.

---

2. Exploratory Data Analysis (EDA)

Insights from Visualizations & Statistics

The EDA phase of the project focused on understanding the insurance data and uncovering patterns that are crucial for the subsequent predictive modeling. The following visualizations and statistics were generated:

1. **Claim Amount Distribution**:

    - A **histogram** and **KDE plot** were used to visualize the distribution of claim amounts. It was found that most claims tend to fall within a moderate range, with some outliers indicating high-value claims.

2. **Fraudulent Claims Distribution**:

    - A **bar plot** demonstrated the distribution of fraudulent versus non-fraudulent claims. A small but significant percentage of claims were flagged as fraudulent, indicating a need for advanced fraud detection models.

3. **Gender Distribution**:

  - The **bar chart** displayed the gender distribution of policyholders. This insight could help identify any gender-based trends in insurance claims or risk.

4. **Policy Type Distribution**:
   - **Count plots** were generated to visualize the distribution of different policy types. Understanding the popularity of various policy types helps in predicting claim amounts and assessing risk.

5. **Claim Amount by Policy Type**:
   - A **bar plot** was used to show the average claim amount across different policy types. This provides insight into which policy types are associated with higher claims.

6. **Annual Income vs Claim Amount**:
   - **Scatter plots** were employed to analyze the relationship between annual income and claim amount. Higher-income policyholders tend to file larger claims, but there are exceptions.

7. **Correlation Heatmap**:
   - A **heatmap** displayed the correlation between various numeric features in the dataset. This helped identify highly correlated variables that can be used for feature engineering.

These insights were crucial for refining the data and improving model predictions.

---

3. Model Training & Evaluation

Performance Comparison of Different Models

After performing feature engineering and pre-processing, the following models were trained and evaluated:

Module 1: Risk Score Prediction

Objective

Classify policyholders into Low, Medium, or High risk categories to support underwriting decisions.

Models Used

- Logistic Regression
- Random Forest ✅ (Best)
- XGBoost
- LightGBM

Best Model: Random Forest

Metric	Value
Accuracy	92%

Metric	Value
F1-Score	0.89
Precision	0.90
Recall	0.88

Challenges & Solutions

Challenge	Solution
Imbalanced distribution across risk classes	SMOTE oversampling
Overfitting with complex models	Cross-validation, feature selection
Correlated features affecting prediction	Removed highly correlated features based on heatmap

Skills Used

- Data Preprocessing & Encoding
- Classification Algorithms
- Model Evaluation (confusion matrix, ROC-AUC)
- Feature Engineering & Selection
- Hyperparameter Tuning (GridSearchCV)

Module 2: Claim Amount Prediction

Objective

Predict the estimated claim amount filed by a policyholder.

Models Used

- Linear Regression
- Random Forest Regressor ✅
- XGBoost Regressor
- Gradient Boosting

Best Model: Random Forest Regressor

Metric	Value
MAE	₹145
RMSE	₹154
R^2 Score	0.70

Challenges & Solutions

Challenge	Solution
Wide range in claim values	Log transformation of target variable
Outliers skewing predictions	Tree-based models like RF/XGB are more robust to them
Linear regression underperformance	Ensemble models gave better generalization

Skills Used

- Regression Modeling
- Feature Importance Analysis
- Model Tuning (Cross-Validation)
- Evaluation Metrics (MAE, RMSE, R^2)

Module 3: Fraud Detection

Objective

Detect whether a claim is fraudulent using classification algorithms.

Models Used

- Logistic Regression
- Random Forest ✅ (Best)
- XGBoost
- LightGBM

Best Model: Random Forest

Metric	Value
Accuracy	91%
Precision	0.86
Recall	0.78
AUC Score	0.93

Challenges & Solutions

Challenge	Solution
Imbalanced data (few frauds)	SMOTE oversampling and threshold tuning
Low recall from logistic model	Ensemble methods yielded better class separation
Noise in categorical variables	Label encoding & one-hot encoding with feature selection

Skills Used

- Binary Classification
- SMOTE Balancing
- Tree-Based Ensemble Learning
- Evaluation with Precision, Recall, AUC

Module 4: Sentiment Analysis

Objective

Analyze customer reviews/feedback and classify them into Positive, Negative, or Neutral sentiments.

Tools & Techniques

- TextBlob for sentiment polarity scoring
- NLTK for NLP preprocessing
- TFIDF Vectorisation

Accuracy

- ~90% overall accuracy on validation data
- Real-time response processing with prediction

Challenges & Solutions

Challenge	Solution
Noisy and unstructured input	Cleaned via lemmatization, tokenization, stopword removal
Short, Unstructured Text Inputs	Preprocessed texts to improve accuracy.
Subjectivity in neutrality	Custom thresholds for better neutral handling

Skills Used

- Natural Language Processing (NLP)
- Sentiment Classification
- Text Preprocessing (NLTK)
- Streamlit Real-time Interface

Model Evaluation

- **Comparison Metrics**:Naive bayes performed best in their respective tasks.
- **Model Selection**: Due to the importance of accuracy and scalability, Naive Bayes were chosen as the final models for production use.

Module 5: Policy Translation & Summarization

Objective

Translate policy documents into Indian regional languages and generate summarized versions for user clarity.

Stack Used

- PDF Handling: PyPDF2
- Translation: Google Translate API (via `googletrans`)
- Summarization: LSA algorithm via `sumy`
- File Output: ReportLab for custom multilingual PDFs

Supported Languages

- Hindi, Tamil, Telugu, Gujarati, Kannada, Bengali, Malayalam, Urdu

Challenges & Solutions

Challenge	Solution
Poor formatting in extracted PDFs	Pre-cleaned text before processing
Unicode support for Indian scripts	Used compatible fonts with ReportLab
Semantic loss in summary	Combined abstractive + extractive summarization techniques

Skills Used

- PDF Parsing and Cleaning
- Multilingual Translation (API Integration)
- Text Summarization Algorithms
- Unicode and Font Management
- Automated PDF Report Generation

Module 6: Customer Segmentation

Objective

Segment customers based on behavioral and demographic patterns to tailor marketing and pricing strategies.

Techniques Used

- K-Means Clustering
- DBscan Algorithm
- StandardScaler for normalization
- PCA for 2D visualization

Cluster Insights

Cluster	Profile
-1	High risk Customers

Cluster	Profile
0	Low Risk with High profitable customers
1	High risk,frequent Claimers

Challenges & Solutions

Challenge	Solution
Choosing optimal K	Used Elbow Curve + Silhouette Score
Non-scaled input skewed clustering	Normalized using StandardScaler
Hard to interpret clusters	Used PCA to visualize and validate groupings

Skills Used

- Unsupervised Machine Learning
- Clustering Algorithms
- Feature Scaling & Transformation
- Dimensionality Reduction
- Customer Profiling

Module 7: FAQ Chatbot

Objective

Provide real-time, multilingual, AI-powered responses to frequently asked questions by policyholders, improving customer engagement and support automation.

Stack & Techniques Used

- Streamlit for front-end UI
- OpenAI/GPT-based NLP (or local NLP pipeline)
- Intent classification (fallbacks via keyword matching)
- Multi-turn dialogue capability
- Context management with session state
- Multilingual support via Google Translate API

Capabilities

- Understand customer queries in English + regional languages
- Respond to FAQs related to policy, claims, and processes
- Clarify policy features, renewal dates, premiums, documents, etc.
- Translate user inputs/output when needed

Performance

- Accuracy (intent classification): ~88%

- Response time: Instantaneous (<1s with caching)

Challenges & Solutions

Challenge	Solution
Diverse phrasing of insurance queries	Used semantic similarity with transformers or fallback keyword match
Handling out-of-scope queries	Added graceful fallback and default help responses
Maintaining conversation state in Streamlit	Used `st.session_state` to preserve user dialogue context
Multilingual understanding	Integrated `googletrans` to auto-translate input/output dynamically

Skills Used

- Conversational AI / Chatbot Design
- Natural Language Understanding (NLU)
- Multilingual NLP (Google Translate, Unicode handling)
- Streamlit session management
- API Integration (translation + model)
- User Interaction Design

5. Future Enhancements

Potential Improvements or Extensions

- **Integration of Deep Learning**: Moving from traditional machine learning models to deep learning approaches such as neural networks could improve prediction accuracy, especially for complex tasks like sentiment analysis and fraud detection.

- **Real-time Fraud Detection**: Deploying a real-time fraud detection system using streaming data could provide immediate alerts for suspicious claims and reduce response times.

- **Enhanced NLP for Policy Translation**: Using transformer models like GPT-3 or T5 could further improve the translation and summarization of insurance policies, making them more accurate and contextually appropriate.

- **User Experience**: Enhancing the chatbot's ability to provide personalized recommendations based on past user queries would improve customer service and engagement.

- **Data Integration**: Integrating external datasets, such as market trends, could provide a more holistic view of customer behavior and improve predictions.

Conclusion

The AI-powered insurance system offers powerful tools to transform traditional insurance operations. By leveraging machine learning for risk prediction, claim estimation, fraud detection, sentiment analysis, and policy translation, the system enhances decision-making, customer satisfaction, and operational efficiency. Future enhancements, such as real-time fraud detection and deep learning integration, will further boost the capabilities of this system.

Appendices

A. Data Overview

The dataset used in this project contains various features related to insurance policies, claims, and customer information. Below is a summary of key features:

Feature	Description	Data Type
Policy_ID	Unique identifier for each insurance policy	String
Gender	Gender of the policyholder	Categorical
Age	Age of the policyholder	Integer
Annual_Income	Annual income of the policyholder	Integer
Policy_Type	Type of the insurance policy (e.g., Health, Auto, Life)	Categorical
Claim_Amount	Amount claimed by the policyholder	Float
Fraudulent_Claim	Indicates whether the claim was fraudulent (1) or not (0)	Binary (0/1)
Premium_Amount	The premium paid for the insurance policy	Float

---

B. Model Code

Below are the high-level code snippets for training models used in this project. Each model was trained and evaluated using its respective algorithm.

Random Forest Classifier (Example)

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split

# Load data and split into features and target
X = df.drop(['Claim_Amount', 'Fraudulent_Claim'], axis=1)
y = df['Fraudulent_Claim']

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Initialize the model
```

```python
rf_model = RandomForestClassifier(n_estimators=100, max_depth=10, random_state=42)

# Fit the model
rf_model.fit(X_train, y_train)

# Evaluate the model
accuracy = rf_model.score(X_test, y_test)
print(f'Accuracy: {accuracy}')
```