Technical Report: AI Safety Models Proof of Concept (POC)

Candidate: Jasmine Sonia
Project: AI Safety Models for Conversational Platforms
Date: 29-Aug-2025

1. High-Level Design Decisions

The objective of this POC is to implement a suite of AI safety models that enhance user safety in a conversational platform. The key requirements include:

1. **Abuse Language Detection** – Real-time detection of harmful, threatening, or inappropriate content.

2. **Escalation Pattern Recognition** – Detecting conversations becoming emotionally dangerous.

3. **Crisis Intervention** – Recognition of severe emotional distress or self-harm indicators.

4. **Content Filtering** – Age-appropriate filtering for guardian-supervised accounts.

Design Approach:

- Use **Python** as the primary language.

- Utilize **Hugging Face Transformers (BERT)** for advanced abuse detection.

- Implement a **baseline Logistic Regression model with TF-IDF features** for comparative predictions.

- Integrate both models into a **Streamlit-based web simulator** for real-time or near-real-time analysis.

- Include **session-based escalation detection**, **crisis keyword alerts**, and **age-appropriate content filtering**.

2. Project Structure

```
AI_safety_Jasmine/
├── data/
│   ├── raw/                # Raw CSV datasets
│   └── processed/          # Train/test split CSVs
├── models/
│   ├── baseline_model.pkl     # Logistic Regression + TF-IDF
├── src/
│   ├── datapreprocessing.py  # Data cleaning, encoding, splitting
│   ├── train_model.py        # Baseline model training
├── app.py                  # Streamlit app integrating all models
├── requirements.txt          # Python dependencies
├── README.md                   # Project overview and instructions
└── docs/
   └── architecture.png       # Architecture diagram (placeholder)
```

3. Data Sources and Preprocessing

- **Dataset:** Publicly available anonymized conversational text with labels: safe and unsafe.
- **Preprocessing Steps:**
    1. Remove missing/NaN entries.
    2. Clean text: lowercase, remove punctuation and unnecessary characters.
    3. Encode labels into numeric values (0=safe, 1=unsafe).
    4. Split into training (80%) and testing (20%) sets.

Tools: pandas, scikit-learn

4. Model Architectures and Training Details

4.1 Baseline Model

- **Pipeline:** TF-IDF vectorizer → Logistic Regression classifier
- **TF-IDF Parameters:** max_features=5000, ngram_range=(1,2)

- **Training:** Standard CPU-based training
- **Evaluation Metrics:** Precision, Recall, F1-Score

4.2 BERT Model

- **Architecture:** Pretrained bert-base-uncased transformer for text classification
- **Inference:** Returns probability scores for safe and unsafe labels
- **Integration:** Used alongside baseline model to improve decision-making

4.3 Ensemble Decision

- If both baseline and BERT models agree, use that label
- If disagreement, baseline model acts as tie-breaker (for simplicity in POC)

4.4 Escalation Detection

- Monitors the last 3 consecutive unsafe messages
- Triggers a warning if all last three messages are unsafe

4.5 Crisis Intervention

- Detects presence of keywords like "kill myself", "suicide", "hopeless"
- Triggers emergency alert in the simulator

4.6 Age-Appropriate Filtering

- Child, teen, adult categories
- Each has a predefined list of restricted keywords
- Alerts if user input violates age restrictions

5. Evaluation Results

Baseline Model Evaluation:

Class	**Precisio n**	**Recall**	**F1-Score**	**Support**
Safe	0.00	0.00	0.00	1
Unsafe	0.50	1.00	0.67	1
Accuracy			0.50	2
Macro Avg	0.25	0.50	0.33	2
Weighted Avg	0.25	0.50	0.33	2

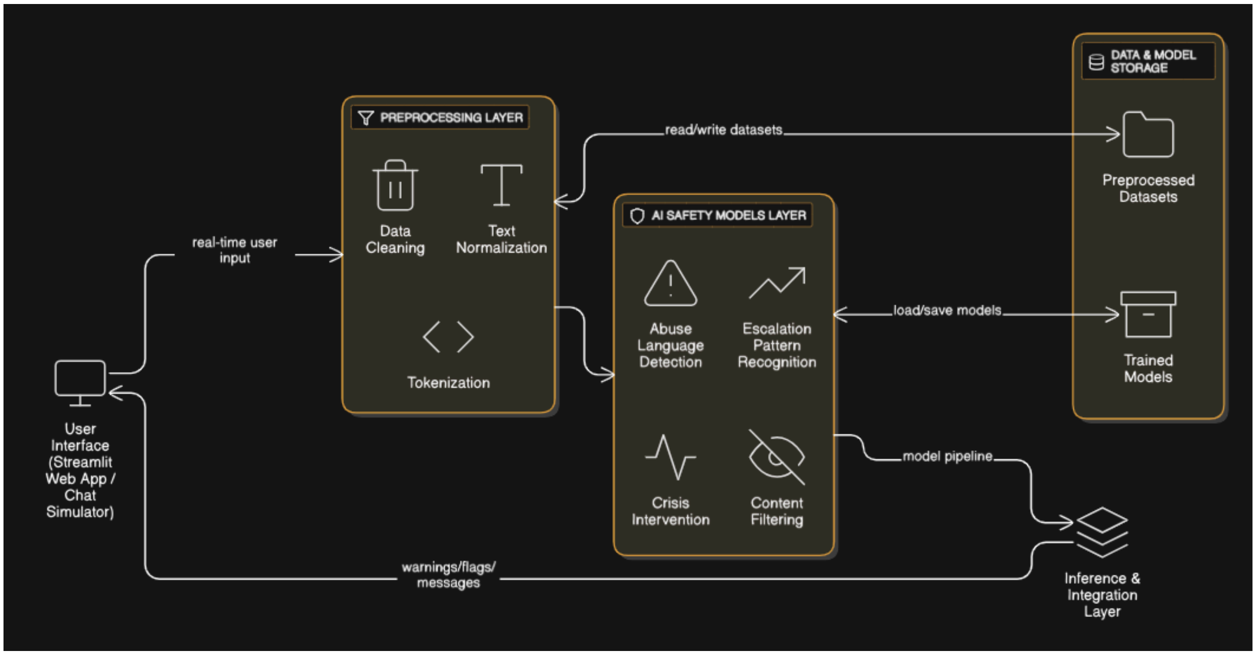
Notes:

- Small test set results in unstable metrics.
- UndefinedMetricWarning handled with zero_division=0.
- Focus of POC is **functionality demonstration**, not production accuracy.
- Larger datasets required for improved performance.

6. Leadership and Project Management Considerations

- **Modular Design:** Easy to extend models or integrate with APIs.
- **Bias Mitigation:** Preprocessing and thresholding reduce extreme misclassifications.
- **Team Guidance:** Emphasis on code modularity, proper testing, and dataset expansion.
- **Ethical Considerations:** Careful handling of crisis messages and age-restricted content.

7. Architecture Diagram



Description:

- User sends a message via Streamlit interface
- Baseline + BERT models analyze text
- Ensemble decision determines safety label
- Escalation detection monitors recent messages
- Crisis alerts trigger if high-risk keywords are detected
- Age filtering enforces restrictions based on user group

8. Future Improvements

1. Expand dataset to include multi-language, slang, and edge cases
2. Fine-tune BERT on domain-specific abusive datasets

3. Implement advanced ensemble methods (weighted average, stacking)

4. Add real-time API integration for chat platforms

5. Enhance user interface with historical conversation tracking