Principles of AI Engineering

Exercise 03

Training a Random Forest Model for GitHub Issue Classification

Syed Muhammed Hassan Ali

Source Code

Training the Random Forest Model

Content

- **Objective:** Classify GitHub issues into categories like bugs, enhancements, and questions.
- **Dataset:** GitHub Issues dataset containing text descriptions.
- **Model:** Random Forest Classifier using scikit-learn.
- **Approach:**
    a. Data Preprocessing
    b. Feature Extraction with TF-IDF
    c. Model Training with Cross-Validation
    d. Evaluation and Serialization for Deployment

Model Pipeline Components

1. **Text Preprocessor:**
   - Functionality:
     - Removes noise (URLs, special characters)
     - Normalizes text (lowercase, whitespace)
     - Tokenizes and lemmatizes text
     - Removes stop words
   - Benefit: Ensures clean and consistent input for the model.
2. **TF-IDF Vectorizer:**
   - Purpose: Converts text into numerical features.
   - Parameters:
     - max_features=5000
   - Benefit: Highlights important words while minimizing common word impact.
3. **Random Forest Classifier:**
   - Configuration:
     - n_estimators=100
     - max_depth=None
     - random_state=42
   - Benefit: Handles large feature spaces and provides robust classification.

Training and Evaluation of the Model

- **Training Methodology:**
  - Used scikit-learn's RandomForestClassifier.
  - Applied cross-validation for reliable performance assessment.
  - Serialized model using joblib for future deployment.
- **Evaluation Metrics:**
  - Overall Accuracy: 73%
  - Bug Classification:
    - Precision: 76%, Recall: 79%
  - Enhancement Classification:
    - Precision: 70%, Recall: 80%
  - Question Classification:
    - Precision: 61%, Recall: 9% (due to class imbalance)

Drawbacks of Random Forest with Concept Drift

1. **Inflexibility to New Data:**
   - Static after initial training—requires complete retraining for updates.
   - Unable to learn from streaming data or new patterns.
2. **Memory and Computational Overhead:**
   - Stores multiple decision trees, increasing memory usage.
   - High computational cost for retraining.
3. **Feature Space Limitations:**
   - Fixed vocabulary—fails to recognize new or evolving terms.
   - Sensitive to distributional changes in input data.

Addressing Concept Drift – Alternative Models

Alternative: SGDClassifier (Stochastic Gradient Descent)

- **Advantages:**
  - Incremental Learning with partial_fit—adapts to new data.
  - Reduced Memory Usage—only stores model parameters.
  - Efficient for streaming and real-time data updates.
- **Implementation Strategy:**
  - Regular updates with new data.
  - Monitoring for significant performance drops.
  - Applying a sliding window approach for recent data patterns.

Conclusion and Future Work

- **Summary:**
  - Successfully trained a Random Forest model for issue classification.
  - Modularized components for integration into a Flask app.
  - Identified Random Forest limitations under concept drift.
- **Future Directions:**
  - Experiment with adaptive models like SGDClassifier.
  - Enhance dataset with balanced classes.
  - Deploy model in Flask for real-time predictions.