■ Spam Email Classification

Advanced ML Pipeline with OpenSpec Workflow

Report Generated: 2025■11■14■

Executive Summary

This report documents a complete end-to-end machine learning project for spam email classification. The project implements a logistic regression model trained on 5,574 SMS messages with 96.95% test accuracy. The implementation includes data preprocessing, model training, evaluation, and an interactive Streamlit web application for real-time classification and analysis.

Project Overview

Aspect	Details
Dataset Size	5,574 SMS messages
Spam Ratio	13.4% (747 spam, 4,827 ham)
Model Type	Logistic Regression
Test Accuracy	96.95%
Precision (Spam)	100%
Recall (Spam)	77.18%
F1 Score	0.871
Vectorization	TF-IDF (max 5,000 features)
N-grams	Unigrams and Bigrams (1-2)

Threshold Sweep Analysis

The following table shows model performance metrics across different decision thresholds, allowing optimization for specific use cases (prioritize precision or recall):

Threshold	Precision	Recall	F1 Score
0.1	0.6558	0.9920	0.7896
0.2	0.9283	0.9705	0.9490
0.3	0.9744	0.9170	0.9448
0.4	0.9893	0.8648	0.9229

0.5	0.9915	0.7831	0.8751
0.6	0.9910	0.5877	0.7378
0.7	1.0000	0.3548	0.5237
0.8	1.0000	0.1299	0.2299
0.9	1.0000	0.0147	0.0290

Data Preprocessing Pipeline

The project implements a comprehensive 7-stage text preprocessing pipeline:

Stage	Operation	Purpose
1. Raw	Original text	Baseline reference
2. Lowercase	Convert to lowercase	Normalization
3. Contact Masking	Mask emails/phones	Remove PII
4. Number Replacement	Replace digits with <NUM>	Generalization
5. Punctuation Removal	Remove special characters	Simplification
6. Whitespace Normalization	Normalize spaces	Formatting
7. Stopword Removal	Remove common words	Feature reduction

Key Features

1. Multi-Format CSV Support: The application supports both simple 2-column CSV format and advanced 9-column preprocessing pipeline format for detailed text transformation analysis.

2. Interactive Dashboard: Streamlit-based web application with real-time classification, token analysis, and model performance visualization.

3. Advanced Analytics: Threshold sweep analysis, ROC curves, confusion matrices, and precision-recall curves for comprehensive model evaluation.

4. CLI Tools: Command-line utilities for batch prediction, visualization generation, and model training with custom parameters.

5. Professional Documentation: Comprehensive README, quick-start guides, and delivery summaries with usage examples and technical details.

Technology Stack

Component	Technology	Purpose
Language	Python 3.12+	Core implementation
ML Framework	Scikit-learn	Model training & evaluation
Data Processing	Pandas, NumPy	Data manipulation
Visualization	Plotly, Matplotlib, Seaborn	Interactive & publication charts
Web Framework	Streamlit	Interactive dashboard
Serialization	joblib	Model & vectorizer storage
Deployment	Streamlit Cloud	Public web application
Version Control	Git, GitHub	Code management
Workflow	OpenSpec	Specification-driven development

Project Structure

```
.
■■■ app.py # Streamlit web application
■■■ train.py # Model training script
■■■ requirements.txt # Python dependencies
■■■ README.md # Project documentation
■■■ src/
■ ■■■ data_loader.py # Data loading & preprocessing
■ ■■■ model_trainer.py # Model training & evaluation
■■■ scripts/
■ ■■■ predict_spam.py # CLI prediction tool
■ ■■■ visualize_spam.py # Visualization toolkit
■ ■■■ generate_report.py # PDF report generation
■■■ data/
■ ■■■ sms_spam_clean.csv # Clean 2-column format
■ ■■■ sms_spam_preprocessing.csv# 9-column preprocessing pipeline
■ ■■■ sms_spam_no_header.csv # Original format
■■■ models/
■ ■■■ logistic_regression.pkl # Trained model (joblib)
■ ■■■ vectorizer.pkl # TF-IDF vectorizer
■ ■■■ label_mapping.json # Label mappings
■ ■■■ metrics_logistic_regression.json # Performance metrics
■ ■■■ threshold_sweep.json # Threshold analysis
■ ■■■ test_predictions.json # Test predictions for ROC
■■■ docs/
■■■ PREPROCESSING.md # Preprocessing documentation
```

Model Performance Results

The Logistic Regression model achieved excellent performance on the spam classification task:

Metric	Value	Description
Test Accuracy	96.95%	Overall correctness of predictions
Precision (Spam)	100%	All spam predictions were correct
Recall (Spam)	77.18%	77% of actual spam was detected
F1 Score	0.871	Harmonic mean of precision & recall
ROC-AUC	~0.98	Excellent discriminative ability
Specificity	100%	No false positive rate
True Negative Rate	100%	All legitimate emails correctly classified

How to Use

1. Running the Web Application:
```
streamlit run app.py
```
Open your browser to http://localhost:8501

2. Making Predictions (CLI):
```
python scripts/predict_spam.py --text "Your message here"
```

3. Batch Predictions:
```
python scripts/predict_spam.py --input data.csv --output predictions.csv
```

4. Generating Visualizations:
```
python scripts/visualize_spam.py --input data.csv --dist --tokens
```

5. Training Model:
```
python train.py --model logistic_regression
```

Conclusions & Future Work

Achievements:
• Successfully built a high-accuracy spam classification model (96.95% accuracy)
• Implemented comprehensive data preprocessing pipeline with 7 stages
• Created professional interactive dashboard with real-time classification
• Developed CLI tools for batch processing and automation
• Demonstrated OpenSpec specification-driven development workflow

Future Enhancements:
• Support for multiple languages and character sets
• Ensemble models combining multiple algorithms
• Active learning with user feedback integration
• Advanced NLP techniques (BERT, transformers)
• Deployment to cloud platforms (AWS, GCP, Azure)
• Real-time model retraining with new data

Generated on 2025■11■14■ | GitHub: github.com/Brain0927/HW3_Fulong_5114056035