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.

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 table below shows model performance metrics across different decision thresholds, enabling optimization for specific use cases.

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: Supports simple 2-column and 9-column preprocessing pipeline formats.

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

3. Advanced Analytics: Threshold sweep, ROC curves, confusion matrices, and precision-recall curves.

4. CLI Tools: Command-line utilities for batch prediction and visualization generation.

5. Professional Documentation: README, quick-start guides, and technical delivery summaries.

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

```
. (root)
■■■ app.py - Streamlit web application
■■■ train.py - Model training script
■■■ requirements.txt - Python dependencies
■■■ src/
■ ■■■ data_loader.py - Data loading
■ ■■■ model_trainer.py - Model training
■■■ scripts/
■ ■■■ predict_spam.py - CLI prediction
■ ■■■ visualize_spam.py - Visualizations
■ ■■■ generate_report.py - PDF report
■■■ data/
■ ■■■ sms_spam_clean.csv - 2-column format
■ ■■■ sms_spam_preprocessing.csv - 9-column pipeline
■ ■■■ sms_spam_no_header.csv - Original format
■■■ models/
■ ■■■ logistic_regression.pkl - Trained model
■ ■■■ vectorizer.pkl - TF-IDF vectorizer
■ ■■■ metrics_logistic_regression.json - Metrics
■ ■■■ threshold_sweep.json - Threshold analysis
■ ■■■ test_predictions.json - Test predictions
■■■ docs/ - Documentation files
```

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
```

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

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

4. Training Model:
```
python train.py
```

Conclusions & Future Work

Achievements:
• Built high-accuracy spam classification model (96.95% accuracy)
• Implemented comprehensive 7-stage preprocessing pipeline
• Created professional interactive dashboard with Streamlit
• Developed CLI tools for batch processing
• Demonstrated OpenSpec specification-driven workflow

Future Enhancements:
• Support for multiple languages
• Ensemble models combining multiple algorithms
• Active learning with user feedback
• Advanced NLP techniques (BERT, transformers)
• Cloud platform deployment