Georgia State University
Machine Learning
Andres Cuervo

11/28/25

Project Report

Classification Study Report

This report summarizes the machine learning classification experiments conducted across four distinct classification tasks.

Each task varied significantly in sample size, feature dimensionality, and class distribution, which strongly influenced model performance, feature selection strategies, and final model choice.

For all four classification tasks, the workflow followed a consistent pipeline-based approach. Training and test datasets were imported into Pandas DataFrames, missing values were replaced with nulls, and a standardized preprocessing pipeline was applied.

The pipeline included a median imputer, StandardScaler, and variance/covariance threshold filtering. Six base models were trained for each task: Logistic Regression, Support Vector Machine (SVM), Random Forest, Multilayer Perceptron (MLP) classifier, k-Nearest Neighbors (KNN), and Gaussian Naïve Bayes.

CLASSIFICATION TASK 1 – Features: 3312 – Samples: 150

Multiple models performed similarly in terms of accuracy, precision, recall, and F1 score. Logistic Regression achieved 100% training accuracy, indicating possible overfitting. However, after running a 3-fold cross-validation, it still achieved the best average accuracy among all models. Initial attempts at aggressive feature reduction reduced validation accuracy, suggesting that the model benefited from the high-dimensional feature space. Thus, Logistic Regression was selected as the final model.

CLASSIFICATION TASK 2 – Features: 9182 – Samples: 100

Due to the extremely high feature-to-sample ratio, model selection focused on approaches that handle high-dimensional data well.

Base Random Forest and Logistic Regression models performed the best, with accuracies of approximately 85% and 90%, respectively.

Logistic Regression was selected because it maintained stronger performance in cross-validation and is more stable with limited samples. Automatic feature selection was applied, which preserved accuracy while improving F1 score, indicating more stable classification across classes.

CLASSIFICATION TASK 3 – Features: 112 – Samples: 2547

This dataset was the most balanced in terms of dimensionality and sample size. Several models produced mediocre results, with MLP being the only model to exceed 80% accuracy. Manual feature

selection was applied, along with hyperparameter tuning of the MLP classifier, including adjustments to hidden layer sizes and activation functions. Cross-validation F1 score improved from 81% to 84%, demonstrating that the model benefited from targeted tuning. The optimized MLP was selected as the final model.

CLASSIFICATION TASK 4 – Features: 1119 – Samples: 11

This dataset suffered from an extremely small sample size and severe class imbalance. No model exceeded 70% base accuracy. Random Forest was chosen due to its ability to handle high-dimensional input with minimal assumptions and robustness with very small datasets. PCA was applied to reduce feature dimensionality, and GridSearchCV was used to fine-tune model parameters. Accuracy increased from 65% to 67%. However, precision and recall for multiple underrepresented classes remained at 0 due to lack of sufficient data. Improving these metrics would significantly reduce overall accuracy, and because the relative importance of precision/recall versus accuracy is unknown for this application, the highest accuracy model was chosen.

Overall, the four classification tasks required different modeling strategies due to the large variations in dataset shape and quality. In high-dimensional, small-sample settings (Tasks 1, 2, and 4), linear models and random forests performed reliably, while Task 3 benefited most from a neural network-based approach. Cross-validation, feature selection, PCA, and grid search techniques were essential in selecting the strongest-performing model for each task.

Ham vs Spam section

1) **Datasets:**

We are given two CSVs for training data (spam_train_1.csv and spam_train_2.csv), and their metrics are as follows:

Training Dataset 1: Total samples: 2,228, ham emails: 1,927 (86.5%), and spam emails: 301 (13.5%). **Training Dataset 2:** total samples: 2,068, ham emails: 1,440 (69.6%), spam emails: 628 (30.4%). **Combined Training Data:** total samples: 4,296, class distribution shows an imbalance favoring ham emails. We are also given another CSV for testing the data, which is unlabeled (spam_test.csv): **Total samples:** 6,447 unlabeled emails for final predictions

2) **Methodology:**

Pre-processing

- **Data Cleaning,** loaded datasets using pandas, standardized column names across datasets, and mapped labels to numeric values (ham=0, spam=1)
- **Handling Missing Values,** dropped rows with null or empty text fields, and removed duplicate entries
- **Text Normalization,** replaced carriage returns and line breaks (`\r\n`) with spaces, removed backslash characters (`\`), and reset indices after data cleaning.

- **Dataset Combination:** merged both training datasets into a single combined dataset, ensured consistent schema and data types, final combined dataset was used for all model training.

TF-IDF Vectorization

We used TF-IDF Vectorization, which transforms text into numerical representations based on how many times they pop up in an email.

- **Stop words removal**: English stop words filtered out
- **Maximum features**: 5,000 (top informative words retained)
- **N-gram range**: (1, 2) - capturing both single words and word pairs
- **Vocabulary**: Fitted on training data only to prevent data leakage

Train vs Validation Split

We split our data with an 80% Training and 20% Validation, which means we used 3436 samples for training and 860 for validating in total.

3) Models:

We chose 3 classification models that are great for their proven effectiveness for text classification, like **Logistic Regression**, **Multinomial Naive Bayes**, and **Linear SVM.** The three of them are made for binary text classification, which is exactly the Ham vs Spam exercise. If we combine them with TF-IDF Vectorization, it makes them more accurate and quicker.

4) Results:

After training and testing, the models were evaluated using multiple metrics to capture different aspects of performance:

- Accuracy: Overall correctness of predictions, precision: Of predicted spam, how many are actually spam (minimizes false positives), recall: Of actual spam, how many were detected (minimizes false negatives), f1 Score: Harmonic mean of precision and recall (balanced measure)

These are our final results, with Linear SVM being the most accurate in detecting Spam emails:

Model	Accuracy	Precision (Spam)	Recall (Spam)	F1 (Spam)
Logistic Regression	96.10%	89.89%	92.35%	91.11%
Naive Bayes	95.99%	94.61%	86.34%	90.29%
Linear SVM	**96.34%**	**90.43%**	**92.90%**	**91.64%**