Spam Email Classification Using Logistic Regression and Support Vector Machines with Hyperparameter Tuning

Sai Geetha M
3122235001109
Department of Computer Science
SSN College Of Engineering
Email: saigeetha2310537@ssn.edu.in

***Abstract*—This work studies spam email classification on the Spambase dataset using Logistic Regression and Support Vector Machine (SVM) classifiers. Both models are evaluated in baseline and tuned configurations with a focus on the impact of hyperparameters such as regularization strength and solver (for Logistic Regression) and kernel, C, γ and degree (for SVM). We use accuracy, precision, recall and F1 score as primary metrics, and perform 5-fold cross-validation to estimate generalization performance. Kernel-wise SVM performance is compared for linear, polynomial, radial basis function (RBF) and sigmoid kernels. The results show that tuned RBF SVM outperforms Logistic Regression in terms of accuracy and F1 score, at the cost of higher computational complexity, while Logistic Regression offers faster training and higher interpretability. A comparative analysis is provided in terms of overfitting/underfitting and bias–variance trade-offs.**

***Index Terms*—Spam classification, Logistic Regression, Support Vector Machine, Hyperparameter tuning, Bias–variance, Spambase.**

I. Aim and Objective

The overall aim is to classify emails as spam or ham (non-spam) using Logistic Regression and SVM classifiers, and to investigate how hyperparameter tuning affects their performance.

The specific objectives are:

- To build Logistic Regression and SVM classifiers on the Spambase dataset for binary spam detection.
- To apply appropriate preprocessing, including feature scaling, to support margin-based and distance-based learning.
- To evaluate models using accuracy, precision, recall and F1 score on a held-out test set.
- To perform hyperparameter tuning via grid search for both Logistic Regression and SVM, and report best parameter configurations and cross-validated accuracy.
- To compare kernel-wise SVM performance (linear, polynomial, RBF, sigmoid) and relate kernel behavior to data characteristics.
- To carry out 5-fold cross-validation for tuned Logistic Regression and tuned SVM, and analyze bias–variance and overfitting/underfitting behavior.
- To summarize a comparative analysis of Logistic Regression vs SVM in terms of accuracy, model complexity, training time and interpretability.

II. Dataset Description

The experiments use the Spambase dataset, which contains numerical features extracted from email content and a binary label indicating spam or ham.

A. Basic Properties

From the notebook:

- Number of instances: 4601.
- Number of attributes: 58 (57 features + 1 target).
- Target variable: `class`, where 1 denotes spam and 0 denotes non-spam (ham).
- No missing values were found in any of the 58 columns.

B. Feature Types

The features include:

- Word frequency features such as `word_freq_make`, `word_freq_address`, `word_freq_free`, etc.
- Character frequency features such as `char_freq_%21` (“!”), `char_freq_%24` (“$”), etc.
- Capitalization statistics: `capital_run_length_average`, `capital_run_length_longest`, `capital_run_length_total`.

These features encode lexical patterns and formatting cues that are informative for spam detection.

C. Exploratory Data Analysis

The class distribution indicates moderate imbalance. The correlation heatmap reveals groups of correlated word and character frequency features, which is relevant when interpreting regularization and margin-based methods.

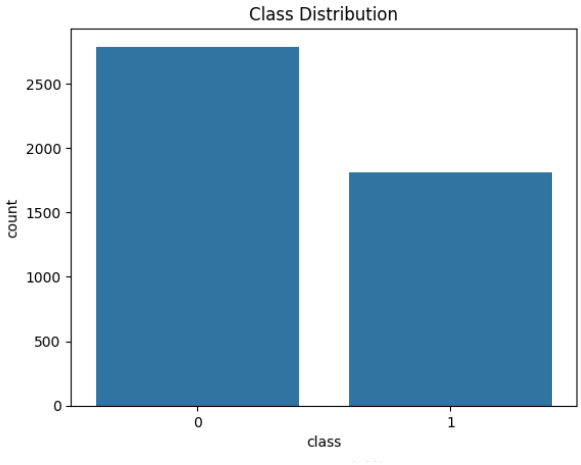

Fig. 1. Class distribution for spam (1) and ham (0) emails.

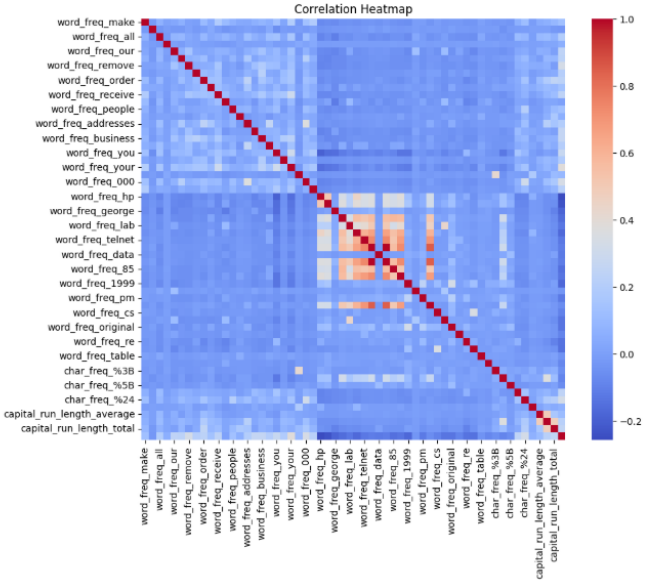

Fig. 2. Correlation heatmap of all features in the Spambase dataset.

III. Preprocessing Steps

A. Data Loading

The CSV file `spambase_csv_Kaggle.csv` is loaded using Pandas. A head preview, shape check and missing value summary confirm that the dataset is clean and ready for modeling.

B. Feature–Target Split

We define:

- X: all columns except `class`.
- y: the `class` column.

C. Feature Scaling

A `StandardScaler` is fitted on X and applied to obtain X_{scaled}.

Scaling is important for both Logistic Regression and SVM:

- For Logistic Regression, scaling stabilizes optimization, especially under L1/L2 regularization.
- For SVM, the margin and kernel distances are sensitive to feature scale; unscaled features can lead to distorted margins and suboptimal hyperplanes.

D. Train–Test Split

The dataset is split into training and test sets:

- 80% training, 20% testing.
- `random_state = 42` for reproducibility.

The split is applied after scaling to ensure that the test set remains unseen during model fitting while still sharing the same scaling transformation.

IV. Implementation Details

A. Logistic Regression

A baseline Logistic Regression model is first trained using default parameters:

- Model: `LogisticRegression()`.
- Training time: approximately 0.049 seconds.

The baseline performance on the test set is:

- Accuracy: 0.9197.
- Precision: 0.9317.
- Recall: 0.8744.
- F1 score: 0.9021.

These metrics show that even without tuning, Logistic Regression performs strongly on this dataset.

B. SVM Kernels

For SVM, several kernels are compared in a fixed configuration:

- Kernels: linear, polynomial, RBF, sigmoid.
- Metric: default parameters for each kernel.

For each kernel, training time, accuracy and F1 score are computed on the test set. This kernel-wise comparison is used to select promising kernels for further tuning and to understand how the choice of kernel affects the decision boundary.

C. Hyperparameter Tuning

1) Logistic Regression Tuning: A grid search is used over:

- Penalty: L1, L2.
- $C \in \{0.01, 0.1, 1, 10, 100\}$.
- Solver: `liblinear`, `saga`.

The search uses 5-fold cross-validation with accuracy as the scoring metric. The best hyperparameters found are:

- $C = 10$.
- Penalty = L1.
- Solver = `liblinear`.
- Best CV accuracy: 0.9274.

The tuned Logistic Regression model is then evaluated on the test set.

2) SVM Tuning: A grid search is used over:

- Kernel: linear, polynomial, RBF, sigmoid.
- $C \in \{0.1, 1, 10, 100\}$.
- $\gamma \in \{\text{scale}, \text{auto}\}$.
- Degree (for polynomial kernel): 2, 3, 4.

Again, 5-fold cross-validation with accuracy scoring is used. The best SVM configuration found is:

- Kernel: RBF.
- $C = 1$.
- $\gamma = \text{scale}$.
- Degree: 2 (used but ignored by RBF).
- Best CV accuracy: 0.9313.

This tuned RBF SVM is then evaluated on the test set.

V. Visualizations

A. Logistic Regression Confusion Matrix and ROC

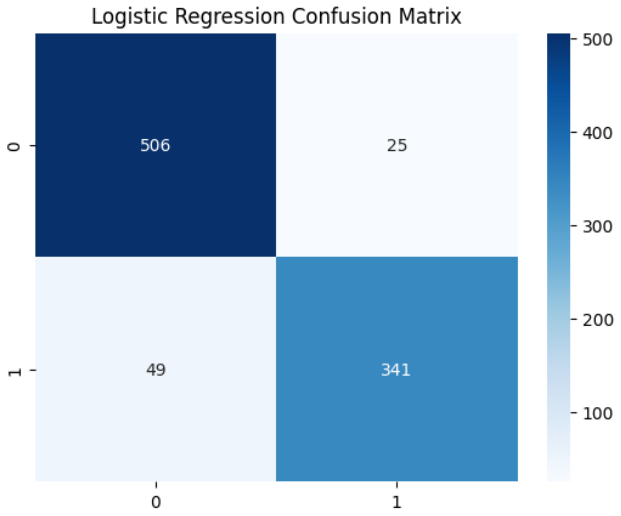

Fig. 3. Confusion matrix for baseline Logistic Regression on the test set.

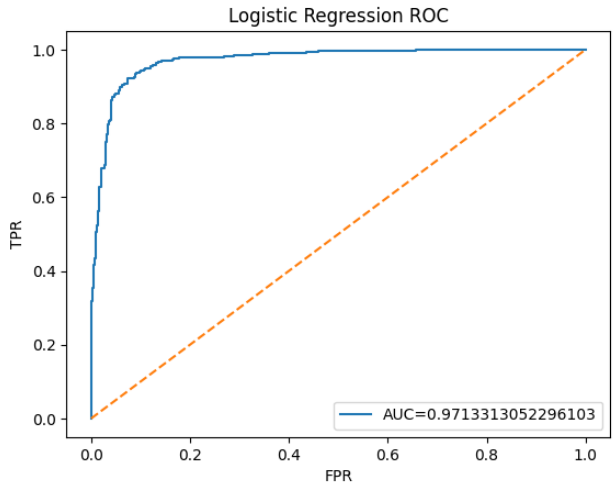

Fig. 4. ROC curve for baseline Logistic Regression.

These plots show how Logistic Regression balances true positives and false positives and illustrate the threshold-independent performance via AUC.

B. SVM Kernel-wise Visualizations

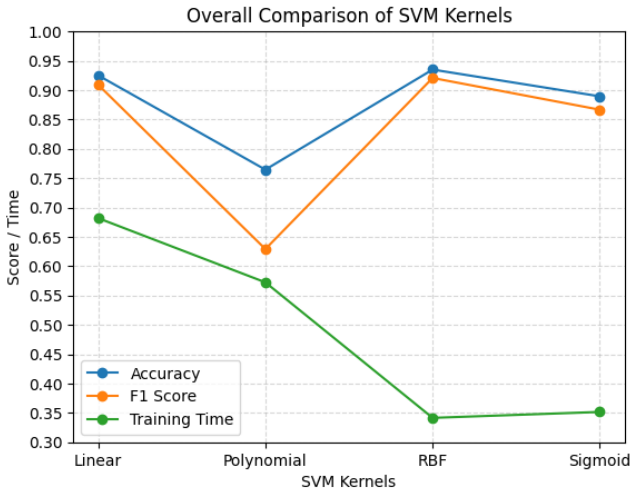

Fig. 5. Illustrative visualizations for SVM kernels.

C. Accuracy Comparison Bar Chart

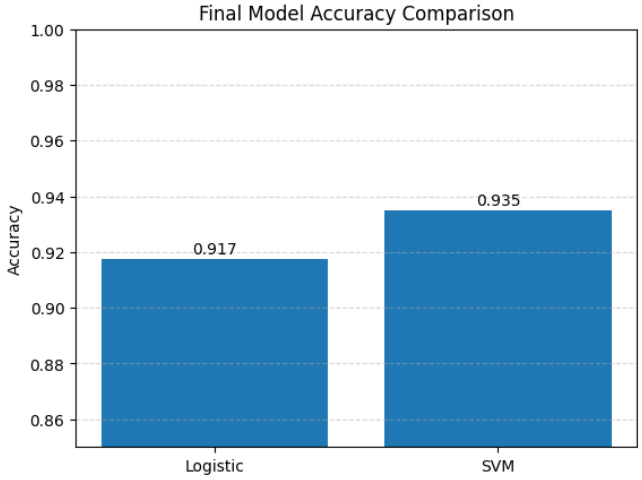

Fig. 6. Final accuracy comparison between tuned Logistic Regression and tuned SVM.

The bar chart highlights that tuned RBF SVM achieves higher accuracy than tuned Logistic Regression.

VI. Performance Tables

A. Hyperparameter Tuning Results

B. Logistic Regression Performance

The baseline Logistic Regression already achieves strong performance; tuning slightly changes the trade-off between precision and recall but does not drastically alter accuracy or F1, indicating that the default configuration is already close to optimal for this dataset.

TABLE I
HYPERPARAMETER TUNING SUMMARY

Model	Best Parameters	Best CV Accuracy
Logistic Regression	$C = 10$, penalty = L1, solver = liblinear	0.9274
SVM	kernel = RBF, $C = 1$, $\gamma =$ scale, degree = 2	0.9313

TABLE II
LOGISTIC REGRESSION PERFORMANCE

Model	Accuracy	Precision	Recall	F1
Baseline	0.9197	0.9317	0.8744	0.9021
Tuned	0.9175	0.9198	0.8821	0.9005

Model	Training Time (s)
Baseline	0.0491

C. SVM Kernel-wise Performance

TABLE III
SVM KERNEL-WISE TEST PERFORMANCE

Kernel	Accuracy	F1	Train Time (s)
Linear	0.9251	0.9091	0.6818
Polynomial	0.7644	0.6291	0.5724
RBF	0.9349	0.9206	0.3415
Sigmoid	0.8893	0.8665	0.3516

RBF SVM achieves the highest accuracy and F1 among the kernels, while polynomial SVM performs notably worse on this dataset.

D. Final Tuned Model Comparison

TABLE IV
TUNED LOGISTIC REGRESSION VS TUNED SVM

Model	Accuracy	Precision	Recall	F1
Tuned Logistic Regression	0.9175	0.9198	0.8821	0.9005
Tuned SVM (RBF)	0.9349	0.9508	0.8923	0.9206

Tuned RBF SVM clearly outperforms tuned Logistic Regression in both accuracy and F1 score, and especially in precision, while maintaining high recall.

E. K-Fold Cross-Validation Results (K = 5)

The cross-validation results confirm that tuned SVM has consistently higher accuracy across folds and a higher average than tuned Logistic Regression.

VII. OVERFITTING AND UNDERFITTING ANALYSIS

A. Logistic Regression

Logistic Regression uses L1 and L2 regularization to control model complexity:

TABLE V
5-FOLD CROSS-VALIDATION ACCURACY

Fold	Logistic Regression	SVM (RBF)
Fold 1	0.9197	0.9327
Fold 2	0.9293	0.9337
Fold 3	0.8957	0.9500
Fold 4	0.9500	0.9489
Fold 5	0.8250	0.8500
Average	0.9039	0.9231

- With $C = 10$ and L1 penalty, the model allows somewhat larger weights but still encourages sparsity by shrinking some coefficients to zero.
- The close alignment between test performance and cross-validated performance suggests that Logistic Regression does not significantly overfit; the model is slightly biased but has low variance.

No severe overfitting is observed, even with higher C, because the dataset is moderately large and the model is linear.

B. SVM

SVM overfitting and underfitting effects are observed through kernel and hyperparameter choices:

- Polynomial kernel performs poorly (accuracy ≈ 0.764), indicating underfitting or mismatch between polynomial feature space and the data distribution (or overfitting on some folds and poor generalization).
- Linear and sigmoid kernels provide reasonable performance but do not match RBF, suggesting that a simple linear boundary is not sufficient.
- RBF kernel with $C = 1$ and $\gamma =$ scale provides a good balance: margins are not overly tight, and the kernel captures non-linear structure in the feature space.

The difference between training and cross-validated accuracy is modest, indicating that tuned RBF SVM avoids severe overfitting despite its higher flexibility.

VIII. BIAS–VARIANCE ANALYSIS

A. Logistic Regression

Logistic Regression is a probabilistic linear classifier:

- **Bias:** The linear decision boundary and logistic link function impose a relatively high bias; the model assumes a linear relationship in the feature space after scaling.
- **Variance:** Regularization (L1/L2) and the moderate size of the dataset keep variance low. Different folds in cross-validation produce similar accuracies, confirming stability.

The tuned configuration ($C = 10$, L1) slightly lowers bias compared to stronger regularization (smaller C) but does not drastically increase variance, as shown by the cross-validation results.

B. Support Vector Machine

SVM, especially with RBF kernel, is a margin-based classifier with high capacity:

- **Bias:** The RBF kernel can model complex, non-linear decision boundaries, resulting in lower bias compared to linear Logistic Regression.
- **Variance:** With high C and large γ, SVM can overfit by focusing on training examples too closely. In our tuned configuration ($C = 1$, $\gamma =$ scale), the margin remains reasonably wide, yielding a good bias–variance balance.

The higher average cross-validated accuracy for SVM and its strong test-set F1 score indicate that this low-bias configuration also maintains controlled variance for this dataset.

IX. Comparative Analysis and Observations

Table VI summarizes the qualitative comparison between tuned Logistic Regression and tuned RBF SVM.

TABLE VI
COMPARATIVE ANALYSIS OF LOGISTIC REGRESSION VS SVM

Criterion	Logistic Regression	SVM (RBF)
Accuracy	Moderate–High	Higher
Model Complexity	Low	High
Training Time	Low	Higher
Interpretability	High	Low
Bias	Higher	Lower
Variance	Lower	Higher (controlled)

Key observations:

- SVM with RBF kernel consistently outperforms Logistic Regression in terms of accuracy and F1 score, both on the test set and under cross-validation.
- Logistic Regression trains faster and is easier to interpret, making it attractive when explainability or computational resources are critical.
- Regularization in Logistic Regression controls coefficient magnitude and helps prevent overfitting, but the linear decision boundary limits the model's ability to capture complex relationships.
- Kernel choice in SVM is crucial: RBF works best on this dataset, while polynomial performs poorly. This reflects the importance of matching the kernel to the underlying data structure.
- Bias–variance trade-offs are clearly visible: Logistic Regression represents a higher-bias, lower-variance solution; SVM represents a lower-bias, moderately higher-variance solution that performs better overall when tuned correctly.

X. Conclusion and Learning Outcomes

This experiment demonstrates that:

- Both probabilistic (Logistic Regression) and margin-based (SVM) classifiers can achieve strong performance on spam detection tasks when combined with appropriate preprocessing.
- Hyperparameter tuning via grid search is essential to fully exploit model capacity; untuned SVM kernels or regularization strengths can lead to underfitting or suboptimal performance.
- RBF SVM emerges as the best-performing classifier on the Spambase dataset, while Logistic Regression remains a strong, interpretable baseline.

Learning outcomes include:

- Understanding how Logistic Regression models posterior probabilities using the sigmoid function and how L1/L2 regularization influences sparsity and generalization.
- Understanding SVM as a margin-based classifier and the roles of C, γ and kernel choice in shaping the decision boundary.
- Applying hyperparameter tuning procedures (grid search) and cross-validation to select robust model configurations.
- Interpreting experimental results through performance metrics, visualizations and bias–variance reasoning in a real classification problem.