Support Vector Machine (SVM) Classifier

Parth Upadhyay
Roll No: 25CS60R35
Technical Report

1. Introduction

Support Vector Machines (SVMs) are a class of supervised learning models used for classification and regression. An SVM constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, SVMs seek the decision boundary that maximizes the margin between different classes, often leading to good generalization.

This report summarizes the theory, methodology, implementation, and experimental results of an SVM classifier.

2. Theory / Background

2.1. Linear SVM

Given labeled training data $(\mathbf{x}_i, y_i)$ where $y_i \in \{+1, -1\}$, a linear SVM seeks a hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ that separates the classes. The margin is defined as the perpendicular distance between the hyperplane and the closest points (support vectors). The optimal separating hyperplane is obtained by solving:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \ i = 1, \dots, n. \end{aligned} \tag{1}$$

2.2. Soft-margin SVM

For non-separable data a soft-margin formulation with slack variables $\xi_i \geq 0$ is used:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \ \xi_i \geq 0. \end{aligned} \tag{2}$$

The regularization parameter C controls the trade-off between maximizing the margin and minimizing classification error on the training data.

2.3. Kernels

The kernel trick allows SVMs to learn nonlinear decision boundaries by implicitly mapping input features to a higher-dimensional feature space. A kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ replaces dot products in the dual formulation. Common kernels include:

- Linear: $K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{z}$
- Polynomial: $K(\mathbf{x}, \mathbf{z}) = (\gamma \mathbf{x}^T \mathbf{z} + r)^d$
- RBF (Gaussian): $K(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma ||\mathbf{x} - \mathbf{z}||^2\right)$

3. Methodology

The notebook implements SVM classification in the following steps:

1. **Data preprocessing:** loading data, handling missing values, feature scaling (standardization), and train/test split.
2. **Model choices:** evaluating linear SVM and kernelized SVMs (RBF, polynomial).
3. **Hyperparameter tuning:** grid search or cross-validation over C, kernel-specific parameters (e.g., γ), and polynomial degree where applicable.
4. **Evaluation:** compute accuracy, precision, recall, F1-score, and confusion matrices. Visualize decision boundaries for 2D data or a reduced projection.

4. Results and Discussion

4.1. Model selection and hyperparameters

Table 1: Comparison of SVM Models and Performance Metrics

Model	**Parameters**	**Accuracy**	**Precision**	**Recall**	**F1**	**AUC**	**Time (s)**
Linear	$C = 10.0$	0.995	0.982	0.9827	0.982	0.9821	2.99
Poly (d=2)	$C = 10.0, \gamma = 1.0$	0.7714	0.7515	0.7515	0.7515	0.7484	62.18
Poly (d=3)	$C = 100.0, \gamma = 1.0$	0.997	1.000	1.000	1.000	1.000	1.80
RBF	$C = 10.0, \gamma = 0.1$	1.0000	1.0000	1.0000	1.0000	1.0000	3.49

4.2. Confusion Matrix and Metrics

	Pred +	Pred -
True +	188	0
True -	0	146

5. Discussion and Analysis

5.1. Kernel Performance

Best Kernel: RBF Kernel performed best as it achieved the highest cross-validation and test accuracy.

Decision Boundaries: The choice of kernel (linear, polynomial, RBF) affects the shape and complexity of the decision boundaries.

Computational Complexity: Compare computational complexity during training and prediction.

5.2. Regularization Effects

Impact of C: The parameter C controls the trade-off between maximizing the margin and minimizing error.

Low C (strong regularization): wider margin, possible misclassifications.
High C (weak regularization): narrower margin, possible overfitting.
Accuracy improved from low C (0.1) $\rightarrow$ medium C (1 or 10).
Low C $\rightarrow$ More support vectors $\rightarrow$ Underfitting.
High C $\rightarrow$ Fewer support vectors $\rightarrow$ Overfitting.

5.3. Dataset-Specific Insights

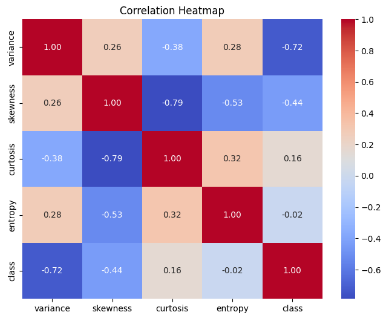
SVM Suitability: Works best for moderate-sized datasets with clear but possibly non-linear boundaries.
Class Imbalance: SVMs can be sensitive to class imbalance.
Feature Scaling: Essential for proper performance due to distance-based nature.

5.4. Recommendations

Best Model: SVM with RBF kernel ($C = 10$, $\gamma = 0.1$).

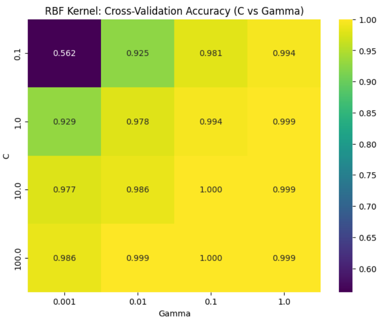
6. Support Vector Machine (SVM) Visualizations

This section presents figures generated from the `SVM_Tutorial.ipynb` notebook.



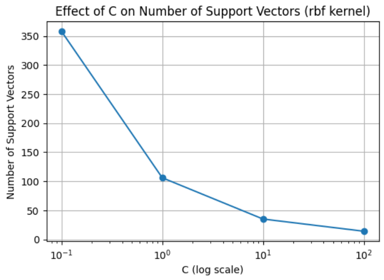
(a) Correlation Matrix

(b) Feature Histogram

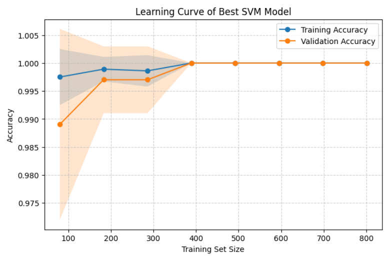


(c) RBF Kernel vs Gamma

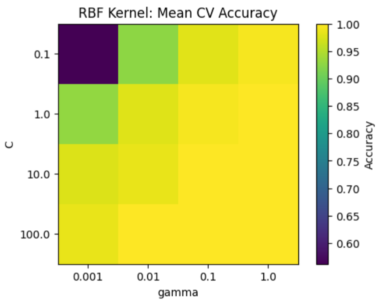
Figure 1: Data and kernel-related visualizations from `SVM_Tutorial.ipynb`.



(a) C vs Support Vectors

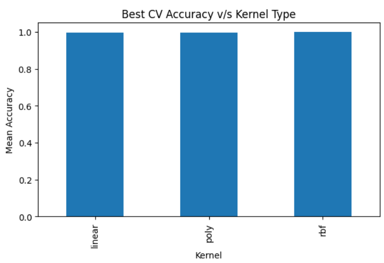


(b) Learning Curve

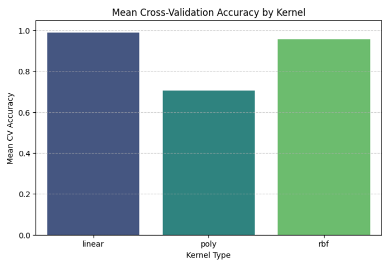


(c) RBF vs Gamma

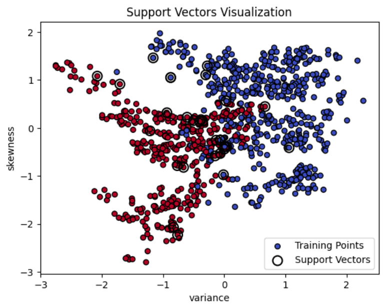
Figure 2: Effect of hyperparameters on SVM performance.



(a) Cross-validation vs Accuracy

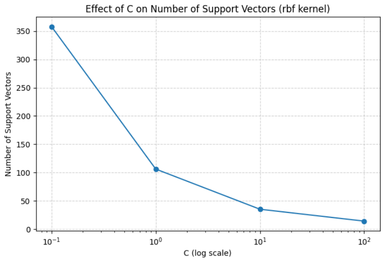


(b) Mean Accuracy vs Kernel Type

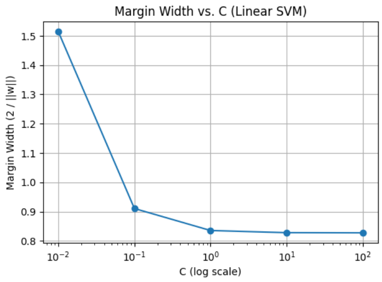


(c) Support Vectors Distribution

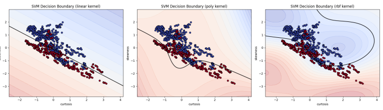
Figure 3: Model accuracy and kernel performance visualizations.



(a) Effect of C on Support Vector Count

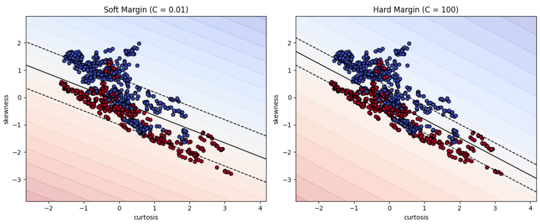


(b) Margin Width vs C

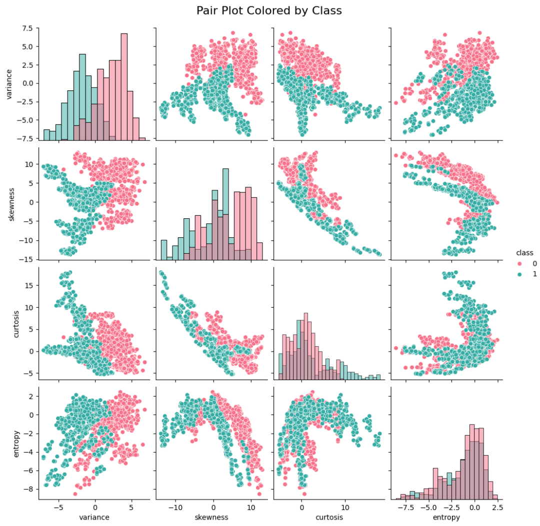


(c) Decision Boundary vs Attributes

Figure 4: Influence of regularization and attributes on the SVM model.



(a) Hard Margin vs Soft Margin



(b) Pair Plot

Figure 5: Margin comparison and pairwise feature relations.

7. Conclusion

This report presented a concise overview of Support Vector Machines and a practical implementation pipeline: preprocessing, model selection, hyperparameter tuning, and evaluation. Kernel selection (RBF vs linear) and the regularization parameter C are the most influential choices. Proper feature scaling and careful cross-validation are essential for reliable performance. The implementation in the associated notebook provides reproducible code for training and evaluating SVMs on tabular datasets.